# Discovering and Leveraging Deep Multimodal Structure for Reliable Robot Perception and Localization

Abhinav Valada

Technische Fakultät
Albert-Ludwigs-Universität Freiburg

Dissertation zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften

Betreuer: Prof. Dr. Wolfram Burgard

# Discovering and Leveraging Deep Multimodal Structure for Reliable Robot Perception and Localization

Abhinav Valada

Dissertation zur Erlangung des akademischen Grades Doktor der Naturwissenschaften
Technische Fakultät, Albert-Ludwigs-Universität Freiburg

| | |
|---|---|
| Dekanin | Prof. Dr. Hannah Bast |
| Erstgutachter | Prof. Dr. Wolfram Burgard |
| | Albert-Ludwigs-Universität Freiburg |
| Zweitgutachter | Prof. Dr. Dieter Fox |
| | University of Washington |
| Tag der Disputation | 7. Februar 2019 |

# Zusammenfassung

Fortschritte in der Robotik und im maschinellen Lernen führen zurzeit zu beispiellosen Veränderungen in verschiedenen gesellschaftlichen Bereichen, einschließlich der Dienstleistungsbranche, des Bergbaus, der Hausarbeit und des Transportwesens. Während wir uns tiefer in das Zeitalter intelligenter Roboter, die nicht mehr nur auf Szenarien in vereinfachten, kontrollierten Umgebungen beschränkt sind, begeben, stehen wir vor der kritischen Herausforderung, sie mit der Fähigkeit auszustatten, unsere sich ständig weiterentwickelnde komplexe Welt wahrzunehmen und zu verstehen. Dies beinhaltet unter anderem, sie mit der Möglichkeit auszustatten, Objekte aus einer Vielzahl von Variationen zu erkennen, hochdynamische Umgebungen zu modellieren, Informationen aus unterschiedlichen Sensormodalitäten zu gewinnen und Entscheidungen daraus abzuleiten und sich an Umstände anzupassen, die das Erscheinungsbild von Orten verändern, wie z. B. unterschiedliche Wetterbedingungen, Tageszeiten, Lichtverhältnisse und strukturelle Veränderungen. Diese Faktoren führen dazu, dass aktuelle Methoden, die von Hand für bestimmte Szenarien entwickelt wurden, fehlschlagen wenn sie mit der Komplexität und dem Reichtums unseres riesigen Wahrnehmungsraums konfrontiert werden.

In dieser Arbeit stellen wir Techniken vor, die es einem Roboter ermöglichen, die Struktur der Umgebung aus früheren Erfahrungen zu lernen, indem robuste Repräsentationen von Modalitäten genutzt werden, die über einfache Bilder hinausgehen, und indem gemeinsame Strukturen zwischen unterschiedlichen Aufgaben entdeckt und ausgenutzt werden. Wenn wir uns davon inspirieren lassen, wie Menschen in einem Gelände ohne visuelle Wahrnehmung spüren und navigieren, schlagen wir zuerst eine Netzwerkarchitektur vor, die Fahrzeug-Boden-Interaktionsgeräusche als eine propriozeptive Modalität nutzt, um ein breites Spektrum von Innen- und Außengeländen zu klassifizieren. Dieses Modell ermöglicht es Robotern, Bodenverhältnisse ohne visuelle Information und auch unter ungünstigen akustischen Bedingungen unter Verwendung kostengünstiger Mikrofone genau zu klassifizieren. Als Zweites stellen wir kompakte Netzwerkarchitekturen für das Szenenverständnis vor, die kontextabhängige Multiskaleninformationen zusammenfassen und neuartige Techniken für die semantische Segmentierung mit hoher Auflösung integrieren. Unsere Modelle können effizient auf Robotern mit begrenzten Rechenressourcen eingesetzt werden, und die daraus resultierende semantische Klassifizierung auf Pixelebene lässt sich effektiv auf Objekte anwenden, die in natürlichen Umgebungen häufig in unterschiedlichen Größenordnungen vorkommen, von Szenarien des autonomen Fahrens im Stadtverkehr bis hin zu Szenarien in Gebäuden und unstrukturierten Waldgebieten. Als Schritt in Richtung neuronaler Modelle, die sich entsprechend der jeweiligen Situation neu konfigurieren, schlagen wir drittens eine selbstüberwachte multimodale Architektur vor, die Informationen aus komplementären Modalitäten wie visuellen Bildern, Tiefe und Infrarot dynamisch verschmilzt, um Objekteigenschaften wie Aussehen, Geometrie und Reflexionsgrad für ein ganzheitliches semantisches Szenenverständnis zu nutzen. Diese Techniken ermöglichen es Robotern, lokale Unklarheiten in der Klassifizierung aufzulösen und die verschiedenen Elemente der Szenerie unter schwierigen Wahrnehmungsbedingungen, die auf wechselnde Wetter- und Jahreszeiten zurückzuführen sind, zuverlässig zu identifizieren. Viertens schlagen wir das erste Ende-zu-Ende-Lernnetzwerk für eine gemeinsame semantische Bewegungssegmentierung vor, welches sowohl Bewegungshinweise zur Verbesserung des Semantiklernens als auch entsprechend semantische Hinweise zur Verbesserung der Bewegungsschätzung nutzt. Mit unseren Modellen können Roboter die Dynamik der Umgebung verstehen und gleichzeitig die Semantik der Szene zeitlich kohärent erkennen. Zum Schluss stellen wir eine Multitask-Lernstrategie vor, mit der automatisch Synergien genutzt werden können, um sich visuell zu lokalisieren, die Szenensemantik vorherzusagen und die Eigenbewegung des Roboters abzuschätzen. Unser Beitrag geht über die bestehenden Paradigmen hinaus, indem er das kooperative Lernen dieser wichtigen Schlüsseltechnologien der Roboterautonomie erleichtert und die geometrischen und strukturellen Bedingungen der Umgebung effektiv kodiert.

Wir werten die in dieser Dissertation vorgestellten Ansätze mit Hilfe von standardisierten Benchmarks umfassend aus und legen umfangreiche empirische Nachweise dafür vor, dass unsere Modelle auf dem neuesten Stand der Technik sind. Darüber hinaus präsentieren wir praxisnahe Versuche aller vorgeschlagenen Methoden mit einer Reihe von verschiedenen Robotern, und demonstrieren damit die Generalisierbarkeit auf eine Vielzahl von Umgebungen und Wahrnehmungsbedingungen. Wir glauben, dass wir Roboter mit Hilfe unserer Methoden einen Schritt näher daran gebracht haben, zuverlässig in komplexen realen Umgebungen eingesetzt zu werden, womit sie unsere Gesellschaft für immer grundlegend verändern werden.

# Abstract

Advances in robotics and machine learning are creating unprecedented seismic shifts in several domains including the service industry, mining, domestic chores and transportation. As we move deeper into the age of intelligent robots that are no longer confined to operational scenarios in simplistic controlled environments, we are faced with the critical challenge of equipping them with the capability of perceiving and understanding our continuously evolving complex world. Amongst others, this entails enabling them to recognize objects that have a multitude of variations, model highly dynamic environments, synthesize and make decisions from disparate streams of modalities and adapt to circumstances that change the visual appearance of places such as different weather conditions, times of the day, illumination and structural changes. Such factors render current methods that are hand-tuned for specialized situations to break down when presented with the complexity and richness of our vast perceptual space.

In this thesis, we introduce techniques that enable a robot to learn the semantic structure of the environment from prior experiences by leveraging robust representations from modalities beyond just visual images as well as by discovering shared structure between multiple diverse tasks. Firstly, taking inspiration from how humans sense and navigate on terrains in the absence of visual perception, we propose novel network architectures that exploit vehicle-terrain interaction sounds as a proprioceptive modality to classify a wide range of indoor and outdoor terrains. These include recurrent models that enable robots to accurately classify terrains without visual information and even in adverse real-world acoustic conditions using inexpensive microphones. Secondly, we present compact fully-convolutional architectures for scene understanding that aggregate multiscale contextual information and incorporate novel techniques for high-resolution semantic segmentation. Our models are efficiently deployable on robots with limited computational resources and the resulting pixel-level semantic classification effectively generalizes to objects that often appear at multiple scales in natural images of different environments ranging from urban autonomous driving scenarios to indoor scenes and unstructured forested environments. Thirdly, as a step towards neural models that reconfigure themselves according to the encountered situation, we propose self-supervised multimodal architectures that dynamically fuse information from complementary modalities such as visual images, depth and infrared to correspondingly exploit object properties such as appearance, geometry and reflectance for a more holistic semantic scene understanding. These techniques enable robots equipped with our models to resolve local ambiguities in their classification and robustly identify the various elements of the scene in challenging perceptual conditions attributed to changing weather and seasons. Fourthly, we propose the first end-to-end learning network for joint semantic motion segmentation that simultaneously exploits motion cues to improve learning of semantics and correspondingly exploits semantic cues to improve motion estimation. Our proposed models enable robots to understand the dynamics of the environment while concurrently being able to discern the semantics of the scene in a temporally coherent manner. Finally, we present a multitask learning strategy to automatically exploit synergies while learning to visually localize, predict the scene semantics and estimate the ego-motion of the robot. Our contributions goes beyond existing paradigms by facilitating cooperative learning of these key robot autonomy enablers as well as by effectively encoding geometric and structural constraints from the environment.

We comprehensively evaluate the approaches introduced in this thesis on standard benchmarks and present extensive empirical evidence that demonstrates that our models achieve state-of-the-art performance. Additionally, we present real-world evaluations of each of our proposed methods using a number of different robots that demonstrate the generalizability to a variety of environments and perceptual conditions. We believe that these techniques have brought robots a step closer towards being reliably deployed in complex real-world environments where they will fundamentally transform our society forever.

# Acknowledgments

This thesis would not have been possible without the support, encouragement, and guidance of many people, to whom I owe my profound gratitude.

First and foremost, I thank my advisor, Wolfram Burgard for giving me the opportunity to pursue my PhD under his guidance. I thank him for providing me the *ideal* research environment at the AIS lab, for the freedom to pursue my own ideas, for his constant encouragement to not only solve hard problems but to make them work on real robots, for always having his doors open for discussions, for providing every resource that I could ever ask for, and for the opportunity to attend the many conferences. I am immensely and sincerely grateful.

I would like to thank Dieter Fox for all the insightful discussions during the conferences over these years and for agreeing to be the second examiner of this thesis. I thank Joschka Boedecker and Matthias Teschner for taking the time to be on my thesis committee. I would like to thank Luciano Spinello for initially inspiring me to work on deep learning at the beginning of my doctoral studies. I would like to thank Daniel Büscher for reading every page of this thesis in such a short time and for his valuable feedback. I would also like to thank Christian Dornhege for the constructive feedback and precious insights. I thank Noha Radwan for wading through the original unedited manuscript and for ironing out the typos.

I had the privilege of collaborating with and learning from many exceptional researchers throughout my doctoral studies. Special thanks to Johan Vertens for our fruitful collaborations, both before and during the PhD. It was a lot of fun building the AIS perception car. I would like to thank Noha Radwan for tirelessly working towards the shared goal of beating the state-of-the-art localization network and the subsequent collaborations. I thank Gabriel Oliveira for all the collaborations that we did during the first year of our PhD. I would like to thank Federico Boniardi for the inputs on how to better formulate problems mathematically. I thank Andreas Wachaja for the numerous discussions on solving robot hardware related issues. I would like to thank Andreas Eitel for setting up and managing the high-performance computing cluster with me all these years. If we had not organized and maintained the cluster, none of this research would have been practically possible. I would like to thank Nichola Abdo for the valuable comments on my research papers during my doctoral studies.

I thank all the students who I have had the opportunity to supervise and collaborate with. Thank you for your hard work and dedication. Special thanks to Rohit Mohan and Ankit

Dhall, for working endless days and not giving up until we had the results.

This work would not have been possible without the tremendous support and friendship of everyone in the Autonomous Intelligent Systems lab all these years. Be it the discussions during coffee breaks, presentations or the numerous barbeques. You all always inspired me and made the time that I spent during my doctoral studies a lot of fun. Special thanks to the senior PhD students and postdocs who gave their valuable advice when I just joined the lab. I thank Gian Diego Tipaldi, Bastian Steder, Barbara Frank, Michael Ruhnke, Tobias Schubert, Christoph Sprunk, Mladen Mazuran, and Pratik Agarwal, for all the advice and valuable feedback. I would like to thank all of my colleagues Tim Caselitz, Ayush Dewan, Tayyab Naseer, Tim Welschehold, Wera Winterhalter, Freya Fleckenstein, Stefano Di Lucia, Henrich Kolkhorst, Marina Kollmitz, Alexander Schiotka, Camilo Gordillo, Felix Burget, Philipp Ruchti, Alexander Schaefer, Lukas Luft, Chau Do, Daniel Kuhner, Andreas Kuhner, Oier Mees, Michael Krawez, Niklas Wetzel, and Martina Deturres, for always being supportive and encouraging. I thank Jan Wülfing, Manuel Watter and Maria Huegle for always readily helping me translate German when ever I was clueless what those letters meant.

Lots of thanks to Susanne Bourjaillat, Evelyn Rusdea, Bettina Schug and Michael Keser for handling all the administrative and technical issues during this entire time. Thank you for being so kind and going out of your way to help.

Special thanks to all my officemates over the years: Federico Boniardi, Gabriel Oliveira, Barbara Frank, Stefano Di Lucia, and Maria Huegle, for putting up with my occasional ranting and everything from talking about work to life. I am grateful to all my friends in Freiburg and beyond, with whom I have had great times and who have kept me sane through my entire doctoral studies. Special thanks to Diego Saul, Sebastian Castaño, Julia Berger, Christoph Scho, Eteri Sokhoyan, Aaron Klein, Carolin Kleber, Anna Doga, Valentina Doga, Branka Mirchevska, Georg Theil, and Eva-Maria Hanke. I thank Shruti Sadani and Rashmi Kumar for being the greatest friends ever since high school. I would like to thank Noha Radwan for constantly motivating me, putting up with my grumpiness during all the crazy deadlines, and always being there through thick and thin.

Finally, I thank my family and my extended family for always supporting me in every decision that I made. I thank my parents, my grandparents and my sister for their unconditional love and support.

# Contents

# Chapter 1

# Introduction

The Dartmouth Conference of 1956, where the term *Artificial Intelligence* was coined [1], fueled an era of discovery in several domains including cognition, language understanding, and visual perception. This soon led to some of the foundations of artificial intelligence for which the four Turing Awards were awarded successively to Marvin Minsky in 1969, John McCarthy in 1971, Herbert Simon and Allen Newell in 1975. These founders of the discipline of artificial intelligence, explored how to make machines capable of human-like perception and intelligence. Each of them predicted that completely intelligent machines capable of doing any work a man can do would be achieved within twenty years [2, 3]. However, immense obstacles lead to repeated failures to meet the set expectations and to the subsequent "AI winters" [4]. Towards the end of the twentieth century Marvin Minsky himself was quoted describing the progress over the years as

> *"In the fifties, it was predicted that in 5 years robots would be everywhere.*
> *In the sixties, it was predicted that in 10 years robots would be everywhere.*
> *In the seventies, it was predicted that in 20 years robots would be everywhere.*
> *In the eighties, it was predicted that in 40 years robots would be everywhere."*
>
> – Marvin Minsky, *1987*

These unanticipated setbacks were primarily due to the underestimation of the complexities of deploying robots in our vast extraordinary world and the oversimplification of the ability to artificially emulate the capabilities of the human brain [3]. Nevertheless, due to the tremendous progress in robotics, computer vision and machine learning over the years [5, 6, 7, 8], we have more robots now than ever today in our factories and homes. The majority of robots perform the most laborious, repetitious and mundane tasks. Development of advanced dextrous manipulators have transformed various industries from automotive manufacturing to electronics assembly. Most cars today are welded, painted and assembled by a robot, as well as most mobile phones today are assembled and quality-inspected largely by robots [9]. The sale of industrial robots reached 16.2 billion USD in 2018 [9]. Robots deployed in these industries have substantially increased production, reduced defects and as a result boosted our economies.

Robots have also made their way into critical sectors such as space exploration [10] and healthcare [11]. All the exploration missions to the moon and mars in the last four decades have been carried out using autonomous rovers. Robots have enabled the exploration of the deepest depths of our oceans where the pressure is over 1,000 times the atmospheric pressure on the surface of the earth [12]. In healthcare, robots are currently being used to perform complicated surgeries such as coronary artery bypasses which is extremely intricate to be performed by surgeons [11]. Consumer home robots for vacuuming, lawn mowing and pool cleaning have been extensively adopted [13]. The worldwide sales of these robots reached USD 6.6 billion in 2018 [14]. Robots are also being deployed for material handling and logistics where they are used for picking and sorting in e-commerce fulfillment warehouses [15]. Telepresence robots [16] are also becoming popular due to their relatively low price and wide range of application domains such as for remote business meetings, tour guides, distance education and security patrolling. Education and therapeutic robots such as the Nao [17] and Paro [18] have gained considerable interest. Moreover, it is projected that the next few years will see a more significant growth in the aforementioned areas [14]. However, this raises a question: *Are we any closer to having robots deployed for tasks that were envisioned in the Dartmouth Conference of 1956?*

Most of these robots today are constrained to operate in fairly controlled environments. Industrial robots that are used for manufacturing are enclosed in safety cages and are fixed to the ground at certain locations in the factory. They are pre-programmed to only handle specific parts and for a particular task. Robots that are used for space exploration operate at a very slow pace in static environments and are mostly programmed in advance by scientists analyzing various forms of imagery. For example the Curiosity rover has a top speed of $0.14\,\mathrm{km\,h^{-1}}$ and it has traveled a total distance of 19.75 km in 7 years [19]. Although, domestic robots operate around humans, they only perform very limited simplistic tasks in a highly structured environment where do they do not require a complete understanding of their surroundings or complex predictions on the various elements of the scene. Similarly, telepresence robots have basic obstacle avoidance functionalities but they are mostly teleoperated. The robots used for surgery have highly restricted motions and are teleoperated by the surgeon. Education and therapeutic robots have a lot of interaction capabilities but they do not autonomously navigate or perform complex tasks where the environment is constantly changing.

However, the recent advent of Convolutional Neural Networks (CNNs) [20] algorithms and parallel computing technology has revolutionized the field of machine learning, computer vision and cognitive robotics, opening doors to a wide range of new domains that can now be effectively addressed. Interestingly, the first publication titled "Neural Nets and the Brain Model Problem" [21] which describes theories about learning in neural networks was introduced by Marvin Minksy in his Ph.D. dissertation. However, one of the biggest milestones was achieved when Krizhevsky *et al.* [22] introduced a neural network architecture in 2010 with eight layers that reduced the classification error rate on the ImageNet challenge [23] by half. Subsequent successes in other areas such as semantic scene understanding [24], object detection [25] and speech recognition [26], demonstrated that classical methods that employ handcrafted feature descriptors could be significantly outperformed by employing this feature learning approach that is completely data-driven. Therefore, CNNs also demonstrate better generalization to different real-world scenarios that encompass our vast visual space. Moreover, several tasks that were tackled using long multi-stage pipelines were reduced to an end-to-end approach that substantially decreased the computation time [20]. In one of the distinguished talks of the early 21st century titled *"It's 2001: Where is HAL"*, Minsky was quoted saying

> *"No program today can distinguish a dog from a cat, or recognize objects in typical rooms, or answer questions that 4-year-olds can!"*
>
> – Marvin Minsky, *2001*

Today, these tasks can be solved with a high degree of accuracy using CNNs [27, 28], in fact surpassing human-level performance in several benchmarks [23, 29]. However, designing these networks is not remotely trivial as there is no clear analytical solution to architecting the right topology. There are several layers, configurations, pre-processing, channels and nonlinearities to choose from. Moreover, there are hundreds of hyperparameters that have to be optimized in addition to the various energy minimization techniques that can be used to train the network. In the context of the robot cognition domain, obtaining sufficient amount of real-world training data for every foreseeable scenario and for every task is an extremely arduous, if not infeasible. As these models have to be deployed for online operation in robots, they have to be highly efficient in their construction in order to meet the real-time performance requirements, while maintaining their accuracy, robustness and generalization ability. Additionally, as robots often have strict computational resource constrains and several cognition functions that have to be modeled, employing dozens of specialized task-specific networks on small embedded GPUs is highly impractical. Multitask learning learning can prove to be an efficient solution by sharing the resources and exploiting the training signals from complementary tasks to alleviate the need of requiring a large amount of domain-specific training data. This increases the complexity of designing suitable architectures as the topology should be structured to facilitate sharing of cross-domain information and enabling inter-task learning. In this thesis, we introduce several architectures to solve fundamental robot perception and localization problems by *discovering and leveraging the inherent structure from the sensor data* as well as by *exploiting the structure across different tasks and modalities*, with the overall goal of enabling these models to be deployed effectively on real robot systems.

Since the last demi-decade, robots are being developed for the next generation of applications for which they not only require complete understanding of our complex dynamic world but they are also required to acquire knowledge, reason about the space, be able to accurately localize in every scenario, learn from experience and be resilient to unexpected situations throughout their deployment. For example, autonomous vehicles such as self-driving cars [30], trucks [31] and shuttles [32], are being developed to provide a more safe mode of transportation and to conserve time that humans spend everyday while driving to their destination. This technology would reduce driver error which accounts for deaths of about 1.2 million people each year and also improve the overall driving efficiency, thereby reducing traffic jams and the fuel consumed [33]. The first step in any safe navigation is to be able to accurately perceive and identify the various objects in the environment surrounding the autonomous vehicle, therefore scene understanding is an essential prerequisite. One of the major challenges is that objects in natural images of outdoor environments often appear in multiple scales, which makes learning features that capture all the different scales of objects extremely challenging. Different semantic objects in the environment also have similar local appearances. In order to accurately recognize these objects, the receptive field of the filters in the network should encapsulate entire objects that are visible in the images. Additionally, as there are often many objects in the scene, it is essential to accurately capture the boundaries of these objects in order to effectively reason about their behavior so that the robot can plan its trajectory to avoid them in a safe manner. More importantly, all of the aforementioned factors should be addressed by the model while being able to perform online in real-time, as it is impractical if the autonomous car navigating amongst human drivers, stops at regular intervals to process its sensor data. This semantic information from scene understanding can then also be used in various other autonomy modules such as localization [34], navigation [35], trajectory planning [36] and human-robot interaction [37].

Another closely related application is focused on last mile logistics [38], where autonomous ground delivery vehicles and aerial delivery vehicles are being developed to provide a more efficient and cost-effective solution for the transportation of goods from retailers, distribution centers, postal services or restaurants to the consumers directly at their doorstep. It is estimated that this would save about 28% of a shipments total cost and this industry is valued at more than 83 billion, while it is further expected to double in value in roughly 10 years [39]. While the technology for these last mile logistics robots is similar to self-driving cars in the context of autonomously navigating in dense dynamic urban environments amongst pedestrians, cyclists and human drivers, it differs in a few distinct ways. Self-driving cars have to adhere more on road traffic regulations and societal driving norms, whereas the last mile delivery robots have to be more agile and be able to navigate around crowds as most of them traverse on sidewalks.

As opposed to the domains where robots are deployed currently such as industrial environments and domestic homes, perception in outdoor environments is exceedingly unpredictable. These environments undergo a wide range of lighting changes that cause shadows on the objects or over and under exposure of the cameras mounted on the robots. Changing weather conditions including fog, mist, haze and rain can drastically reduce the visibility. Seasonal variations such as shedding of leaves or snow can completely change the appearance of a place. It would be of little use if these aforementioned autonomous vehicles are only operable in ideal perceptual conditions which would entail less than 6 out of the 12 months of a year in most part of the world. As the primarily adopted visual RGB images are extremely susceptible to these disturbances, leveraging complementary information from other modalities such as infrared and depth can significantly improve the robustness to such perceptual changes. Additionally, as these changes vary with several factors such as location, time of the day and type of modality being used, the models employed should adaptively fuse features from the different modalities to effectively exploit the complementary information according to the scene condition at that moment.

Moreover, in both the aforementioned application domains, the robots will encounter a wide range of different terrains while navigating in the environment. For example, the self-driving car will come across terrains such as asphalt, offroad rocky surfaces and sand, while the last mile delivery robot will have to traverse a wider range of terrains including cobblestones, paving, asphalt and sand. Additionally, there are factors that transform the appearance of these terrains such as changing weather conditions, icy or wet roads and camouflaging due to leaves. These robots have to adapt their traversability strategy by sensing the terrain in order to avoid potentially dangerous situations. For example, if the self-driving car employs the same speed that it uses to traverse on asphalt to traverse on sand or icy roads, it can correspondingly lead to entrenchment of the wheels or slippage. Similarly, if it traverses with a high velocity on cobblestones with passengers in the car, it would result in a uncomfortable ride while leading to damage of the car suspensions. Often different terrain classes also have similar visual appearances such as sand and granite, therefore entirely relying on camera images makes the system vulnerable to failure. Alternate extroceptive and proprioceptive modalities can be leveraged for a more robust classification.

In the context of indoor environments, robot butlers [40] are a similar class of autonomous vehicles that are being developed to deliver room service orders in hotels, escort people to their destination and deliver packages inside office buildings. On the one hand, indoor environments pose a different set of challenges where robots have to navigate in tighter spaces, accurately localize themselves in the absence of GPS and recognize objects that are often severely occluded. On the other hand, tasks such as localization, motion estimation and scene understanding are similar to the techniques employed outdoors. In addition to understanding the semantics of the scene, in both indoor and outdoor environments it is critical to also estimate if the objects are static or moving. For example, in outdoor environments, the robot needs to first identify if the object is a pedestrian, car or cyclist and then estimate if its static or moving in order for it to plan its trajectory in such a way to avoid potential collision. Similarly, in indoor environments, robots often have to navigate in narrow corridors alongside humans, where they need to first semantically classify the object as a person and then estimate the motion status. Detecting the motion of objects using images is an extremely difficult task as both the motion parallax effect and the ego-motion of the robot critically influence the accuracy of the predictions. The motion parallax effect is often observed for distant moving objects, where the visible pixel displacement caused by the moving object between two successive frames decreases with increasing distance from the camera. Secondly, as the robot is itself moving in most situations, it needs to compensate for its ego-motion in order to accurately estimate the motion of other objects in the environment from consecutive camera images. Additionally, in outdoor environments, large objects such as cars and trucks can be partially occluded due to a tree or a pole causing it to appear as two separate objects. In such situations, the model needs to robustly estimate if both parts of the object are moving or static. Semantic information can also be used to enhance learning of motion features as it can provide the model with the prior on the semantic object classes that are potentially movable. Moreover, in the context of autonomous navigation, the robot simultaneously requires the information about both the semantics as well as the motion of the objects in order to effectively reason about the environment.

In all the applications that we discussed thus far, localizing the robot in the environment is one of the fundamental capabilities that is essential for autonomous navigation. Reliable localization in large-scale environments is an exceedingly challenging problem. Localization can be performed using active sensors such as LiDARs [41], passive sensors such as cameras [42] as well as stereo cameras that provide RGB-D data [43]. LiDAR based localization is often accurate due its ability to capture the geometry of the environment, however the cost of such sensors is prohibitively expensive, whereas vision-based approaches are preferred due to the relative low cost of cameras and reasonable performance in ideal conditions. Vision-based localization is most difficult when the environment contains homogeneously textured regions, reflective surface, repeated structures, motion blur and challenging perceptual conditions. In order to accurately localize using camera images, learning-based approaches can be employed as they have demonstrated substantial robustness in learning features that are resilient to challenging perceptual conditions. However, the network needs to be trained in a way that effectively encodes the geometry of the environment or the sequential trajectory information. Semantics about the environment can also be exploited to improve the robustness in scenes that contain reflective surfaces or repeating structures. The major challenge lies in how to incorporate the semantic information, as directly concatenating the semantic features into the localization network would not yield much benefit as the network needs to be supervised to attend to only the most informative semantic regions while discarding less informative or ambiguous features. Pre-defining stable structures that do not undergo seasonal changes such as buildings is one option [44]. However, this restricts the network from being able to focus on different semantic object classes in different scenes based on the scene context. Therefore, this fusion of semantic features into the localization network has to be learned in a self-supervised manner in order to most effectively utilize the semantic information.

Considering the aforementioned critical challenges that autonomous robots face in various application domains, we pose the following research questions that we address in this thesis.

- How can we enable a robot to use sound from vehicle-terrain interactions to reliably classify terrains using an end-to-end recurrent CNN architecture?
- How do we design a CNN topology for semantic segmentation that accurately segments scenes in indoor, outdoor as well as unstructured forested environments and can be effectively employed on a robot with limited computational resources?
- How do we design a self-supervised multimodal CNN fusion framework that improves the robustness of semantic segmentation both in regular perceptual conditions as well as in adverse conditions?
- How do we design a joint CNN architecture that can enable a robot to simultaneously predict the semantics and the motion status of various objects in the scene? Additionally, how do we improve learning of both tasks by exploiting complementary cues from the other task?
- How do we enable a robot to accurately localize, predict the semantics and estimate its ego-motion using a multitask CNN architecture? Accordingly, how do we enable inter-task learning and improve the performance of the mulitask model over the individual task models?

In the scope of this thesis, we tackle the problems raised by these questions and provide the corresponding solutions that exceed the performance of current state-of-the-art methods both in benchmarking metrics as well as computational efficiency, in addition to being practical for deployment in robots. Although the application scenarios that we described are in the context of autonomous navigation in indoor and outdoor environments, the principle behind these methods is generally applicable to any environment in which a robot needs to perceive and understand its surroundings. These fundamental methods also extend to other robotic domains such as the perception of objects in domestic homes or industrial settings. These solutions are a step towards enabling robots to reliably operate in our complex dynamic world which will pave the way for a new intelligent machine age.

## 1.1 Scientific Contributions

In this thesis, we make several contributions to the fields of robotics, computer vision and deep learning, and more specifically in area of robot perception and localization. Our contributions address the challenges that we highlighted in the previous section that enable robots to robustly perceive and understand our continuously evolving complex world. Thus, bringing us closer to the age of intelligent autonomous robots where these machines are no longer confined to controlled environments. We briefly describe each of these contributions in the rest of this section and we also list them concisely at the end of the introduction section of each chapter.

**Terrain Classification from Vehicle-Terrain Interaction Sounds:** In Chapter 3, we address the problem of proprioceptive terrain classification from vehicle-terrain interaction sounds by presenting two novel CNN architectures, TerrainNet and the recurrent TerrainNet++. We first transform the raw audio signals into its spectrogram representation and subsequently employ them as inputs to our network for learning highly discriminative deep representations. Our TerrainNet model only considers a single time window for classification, whereas our TerrainNet++ model incorporates recurrent units to additionally learn the temporal dynamics of the signal. We propose a Global Statistical Pooling strategy that employes three different pooling methods to aggregate statistics of the temporal features. In order to improve the robustness of our models to various forms of ambient environmental noises, we propose a noise-aware training scheme that randomly injects ambient noise samples of different signal-to-noise ratios while training the network. Our proposed networks are the first end-to-end learning techniques that classify terrains from proprioceptive sensor data. Extensive evaluations on our dataset consisting of over six hours of vehicle-terrain interaction sounds of nine different indoor as well as outdoor terrains demonstrate that our networks achieve state-of-the-art performance and significantly faster inference times than existing techniques. Additionally, we also present generalization experiments with different hardware setups that demonstrate the efficacy of our approach for deployment in environments with different ambient noises.

**Efficient Semantic Scene Understanding:** In Chapter 4, we address the problem of accurate and efficient semantic scene segmentation using visual images. A key challenge in this context is to enable the network to effectively learn features representing objects of multiple scales and ensuring that the network has a large effective receptive field to entirely encapsulate objects such as buses or furniture that occupy a substantial portion of the image in urban and indoor scenes correspondingly. Current techniques that achieve this consume a substantial amount of parameters that makes these models not employable in robotic applications that require fast inference times. We address these problems by introducing two novel fully-convolutional encoder-decoder architectures, AdapNet and AdapNet++. Both our architectures incorporate our proposed multiscale residual units for learning multiscale features throughout the network without increasing the number of network parameters. Additionally, the AdapNet++ architecture incorporates our proposed efficient Atrous Spatial Pyramid Pooling (eASPP) module that has a topology consisting of both parallel and cascaded atrous convolutions with different dilation rates for aggregating multiscale features and capturing long-range context. We employ a bottleneck structure in each of the branches of our eASPP module to conserve the amount of parameters consumed. As urban scenes consist of many thin pole-like structures that are often lost in the segmentation due to the inherent downsampling in the network, we propose a new strong decoder with multistage refinement and a multiresolution supervision strategy to recover the details as well as to also improve the segmentation along object boundaries. In order to efficiently deploy our models on embedded GPUs that are often employed in robots, we propose a network-wide holistic prunning approach that is invariant to shortcut connections in the network. Comprehensive emprical evaluations on multiple benchmark datasets that contain indoor, outdoor as well as unstructured forested scenes demonstrate that our networks achieve state-of-the-art performance, while being compact and having a fast inference time. Furthermore, we present experimental evaluations using our AIS perception car that demonstrate the generalization ability of our models to previously unseen cities.

**Robust Multimodal Semantic Segmentation:** Our AdapNet and AdapNet++ architectures that we introduce in Chapter 4 perform exceedingly well in normal perceptual conditions, however, robots should be able to operate even in adverse perceptual conditions such as rain, snow and fog. In order to achieve this, we propose to leverage features from complementary modalities such as depth and infrared to improve the resilience of the model in such challenging conditions. Moreover, there are several situations where unimodal semantic segmentation networks demonstrate missclassifications, especially due to inconspicuous object classes. For example, often networks classify pictures of people on decals or billboards as a real person. This could lead to unpredictable situations if a self-driving car encounters this scenario. The major challenge in this problem is to enable the network to effectively exploit the complementary features as several factors influence this decision including the spatial location of the objects in the world, the semantic category of the objects and the scene context. Due to the diversity of our world and the numerous weather as well as seasonal changes that alter it, the fusion of the modalities will not be beneficial by directly concatenating the features.

In order to address this problem, we propose two novel adaptive multimodal fusion mechanisms in Chapter 5. We employ a late fusion architecture, where in the first CMoDE approach, we fuse the features at the end of the decoder, while in the SSMA approach, we fuse the features at the end of the encoder. Our proposed CMoDE fusion module is trained in a fully supervised fashion and learns to probabilistically weight features of individual modality streams class-wise. Therefore, during deployment, the model adaptively fuses the features depending on the semantic objects in the scene as well as the information contained in the individual modalities. Our second fusion architecture termed SSMA dynamically fuses the features from individual modality streams according to the object classes, their spatial locations in the scene, the scene context as well as the information contained in the modalities. In order to effectively leverage complementary features in our SSMA model, we employ a self-supervised training strategy. Through extensive experiments on multiple indoor, outdoor and unstructured forest benchmarks, we show that both our networks set the new state-of-the-art while demonstrating substantial robustness in adverse perceptual conditions. In order to further evaluate the robustness and generalization of our network, we present evaluations from real world experiments using our Viona robot that employs the multimodal semantic segmentation as the only perception module during autonomously navigating in a forested environment.

**End-to-End Joint Semantic Motion Segmentation:** As autonomous robots require knowledge about the semantics of the scene as well as the motion of objects such as pedestrians and cars, we propose two end-to-end architectures to address the problem of joint semantic motion segmentation in Chapter 6. Existing approaches that learn to segment semantic objects do not utilize the valuable motion cues and existing motion segmentation techniques do not utilize the semantic cues. Learning both semantics as well as the motion of semantic objects can substantially improve the performance of both tasks. For example, incorporating semantics can enable the motion segmentation network to attend only to regions in the image that contain movable objects. Similarly, incorporating motion cues while learning semantics will enforce temporal consistency in the model. However, the major challenge lies in how to effectively exploit this complementary information from both tasks and how to alleviate the problem of the induced flow magnitudes due to the ego-motion of the robot. In our proposed SMSnet architecture, our main goal is to improve motion segmentation by incorporating semantic cues into the network. Therefore, we introduce a two stream architecture, where one stream learns coarse optical flow field features and the parallel stream learns semantic features. We propose a novel ego-flow suppression technique to remove the flow induced due to the ego-motion of the robot from the predicted full optical flow maps. We then fuse semantic features into the motion segmentation stream and subsequently upsample the feature maps to yield the pixel-wise semantic motion labels.

In our proposed SMSnet++ architecture, we additionally aim to improve the performance of semantic segmentation by exploiting motion cues. In order to achieve this, we propose to transform the learned optical flow maps into an edge-enhanced representation which is then used to warp and fuse intermediate network features from the previous frame into current frame. Furthermore, we employ our SSMA fusion module to adaptively incorporate semantic features into the motion segmentation stream. We present comprehensive experimental evaluations on three benchmark datasets containing scenes in urban autonomous driving scenarios and show that the performance of both our networks substantially exceed the state-of-the-art, as well as the performance of networks that address these two tasks separately. Furthermore, we present evaluations using our AIS perception car that demonstrate the efficacy of our approach to generalize to different challenging scenarios. The networks that we propose are the first end-to-end learning techniques to address the problem of joint semantic motion segmentation.

**Multitask Learning for Semantic Visual Localization and Odometry Estimation:** The problems that we address thus far, primarily focus on scene understanding. However, the robot requires the fundamental knowledge about where it is in the environment in order to navigate. Existing end-to-end networks that regress the 6-DoF camera pose are substantially outperformed by the state-of-the-art local-feature based approaches that utilize structure from motion information for accurate localization. However, these techniques often fail to localize in textureless regions due to the inadequate number of correspondences that are found, whereas CNN-based approaches are more robust to this factor. The key challenge in regressing the 6-DoF poses using CNNs is that the commonly employed Euclidean loss function does not enable the network to learn geometrical constraints about the environment. In order to address this problem, in Chapter 7, we adopt a mulitask learning strategy and propose two architectures complemented with a new loss function that effectively encodes motion-specific information. Our proposed Geometric Consistency loss function incorporates the relative motion information during training to enforce the predicted poses to be geometrically consistent with respect to the true motion model. In order to effectively employ this loss function, we propose the VLocNet architecture that consists of a global pose regression stream a Siamese-type double stream for odometry estimation that both partly share parameters. The network takes consecutive monocular images are input and regresses the relative odometry estimate which is then used in our proposed Geometric Consistency loss function to regress the global pose.

Inspired by how humans describe their location with respect to specific landmarks in the scene and to further improve the localization performance, we propose a new strategy in our VLocNet++ architecture to simultaneously encode geometric and structural constraints by temporally aggregating learned motion-specific information and effectively fusing semantically meaningful representations. We introduce a weighted fusion layer that learns to optimally fuse semantic features into the localization stream based on region activations in a self-supervised manner. Additionally, we utilize the relative motion from the odometry stream for warping and fusing semantic features temporally to improve the performance of semantic segmentation. We further fuse feature maps from the localization stream into the semantic stream using our weighted fusion layer to provide weak supervision signals for training and to enable inter-task learning. Extensive experimental evaluations on benchmark datasets show that our networks are the first deep learning approach to outperform local-feature based techniques, thereby setting the new state-of-the-art, while simultaneously performing multiple tasks with a fast inference time. Furthermore, we present experimental results using our Obelix robot in urban scenarios that are challenging for both perception and localization tasks to demonstrate the exceptional robustness of our models.

## 1.2 Live Demos

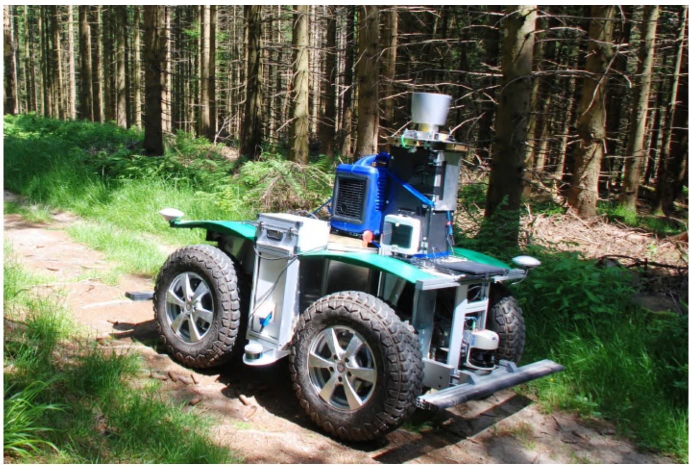
A live demo and several videos of the methods introduced in this thesis are publicly available in the websites listed below.

- Audio Terrain Classification: `http://deepterrain.cs.uni-freiburg.de`.
- Semantic Scene Segmentaion: `http://deepscene.cs.uni-freiburg.de`.
- Multimodal Semantic Segmentation: `http://deepscene.cs.uni-freiburg.de`.
- Semantic Motion Segmentation: `http://deepmotion.cs.uni-freiburg.de`.
- Semantic Visual Localization: `http://deeploc.cs.uni-freiburg.de`.

## 1.3 Dataset Contributions

Generally, research on CNN-based techniques for various robotic tasks is facilitated by the availability of datasets. In the context of this thesis, we published the following annotated datasets that have thereafter been adopted for standardized benchmarking in many works. They are publicly available at `http://aisdatasets.cs.uni-freiburg.de`.

- DeepTerrain Dataset: Audio recordings of over 15 hours of vehicle-terrain interaction sounds on nine different indoor and outdoor terrains with manually annotated classification labels.
- Freiburg Forest Dataset: Multimodal and multispectral images of unstructured forested environments with manually annotated pixel-level semantic labels.
- Cityscapes-Motion Dataset: Manually annotated pixel-level motion labels for moving objects in images from the Cityscapes semantic segmentation benchmark dataset.
- KITTI-Motion Dataset: Manually annotated pixel-level motion labels for moving objects in images from the KITTI semantic segmentation benchmark dataset.
- ApolloScape-Motion Dataset: Manually annotated pixel-level motion labels for moving objects in images from the ApolloScape semantic segmentation benchmark dataset.
- DeepLoc Dataset: Ten image sequences of a large urban environment, with manually annotated pixel-level semantic labels and localization groundtruth labels for each of the images.


(a) Viona autonomous robot

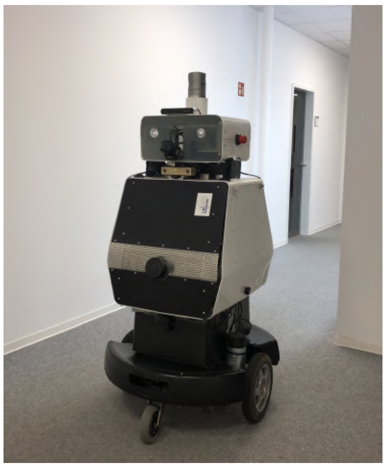

(b) Obelix autonomous robot

**Figure 1.1:** The autonomous robots that we use in our experiments in (a) unstructured, and (b) urban environments correspondingly.

## 1.4 Experimental Robot Platforms

Throughout this thesis, we present several experiments with different robotic platforms to validate the suitability of our models for real-world deployment. In this section, we briefly describe these robots and their corresponding sensor setups. Some of these robots were actively designed and developed by the author of this thesis during the course of this PhD work.

**Viona Robot:** Our Viona robot shown in Figure 1.1 (a) was designed for navigating unstructured forested environments with rugged terrain. The robot is equipped with highly geared strong motors and large wheels that enable it to have a substantial ground clearance. It has a four wheel swerve drive system that can dynamically reconfigure to also run in a differential mode. It runs on four large lead acid batteries that further stabilize its mass while traversing on uneven terrain. Viona is equipped with a suite of sensors including a Velodyne HDL-64E LiDAR scanner on top a tower on the robot that is used for localization and mapping, two Velodyne VLP-16 PUCK LiDAR scanners mounted on the sides to detect obstacles close to the wheels, two SICK LMS LiDAR scanners on the front and back of the robot to evaluate the traversability of the terrain, four Bumblebee2 stereo cameras that give it a 360° field of view around the robot and a front facing NIR camera to detect bushes that can be driven over or avoided. It is also equipped with a Applanix POS LV system that consists of two GPS differential antennas as well as an IMU with three accelerometers and three gyroscopes. For computation, the robot is equipped with a quad core Intel Core i7 CPU, 3.50GHz processor and a NVIDIA TX1 that are both connected using a Gigabit networking switch. We employ this robot for the scene understanding and

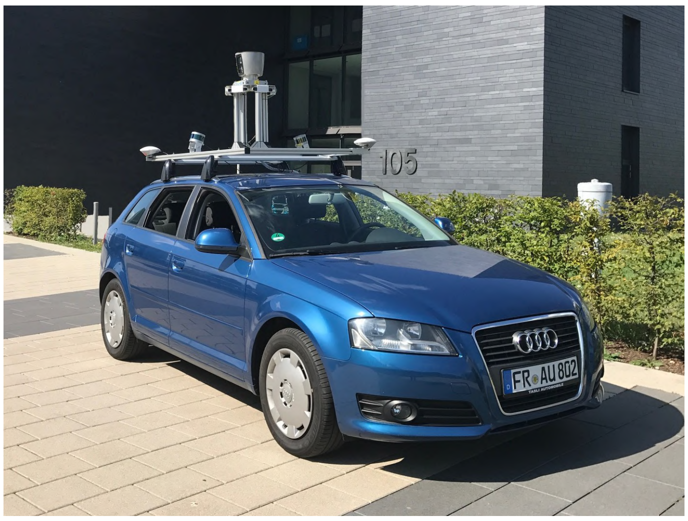


**Figure 1.2:** The AIS perception car that we use in our experiments in urban driving scenarios.

autonomous navigation experiments presented in Chapter 5 and Chapter 4.

**Obelix Robot:** Our Obelix robot [45] shown in Figure 1.1 (b) was designed for autonomous navigation in pedestrian environments. The robot runs on four lead acid batteries and has two unidirectional wheels as well as two castor wheels in the front and back of the robot that enables it to take tight turns in a differential configuration. The sensor system of the robot consists of a Velodyne HDL-32E LiDAR scanner on the top of the robot that is used for localization and mapping, two SICK LMS-151 scanners in the front and back of the robot for traversability analysis, a tilting Hokuyo UTM-30LX in the front of the robot for obstacle avoidance, two Delphi ESR radar sensors which are mounted to the left and right of the robot for tracking moving objects, a Bumblebee2 and ZED stereo cameras facing the front of the robot. It is also equipped with a Trimble GPS Pathfinder Pro and and XSens IMU. We use a laptop with an Intel Core i7 and an NVIDIA GTX 980M GPU for all the computations. We primarily use our Obelix robot for all the experiments presented in Chapter 7 for the evaluation of visual localization, temporal semantic segmentation and odometry estimation.

**AIS Car:** We built the AIS perception car shown in Figure 1.2 for evaluating our perception and localization algorithms in self-driving car scenarios. The sensor system mounted on the roof of an Audi A3 car consists of a Velodyne HDL-64E LiDAR scanner, two Velodyne VLP-16 PUCK scanners mounted an angle of 30° to the horizontal plane, six ZED stereo cameras that are mounted in a hexagonal configuration to give a 360° field of view and Point Grey Blackfly cameras that are mounted facing the front of the robot to acquire wide baseline stereo images. It is also equipped with an Applanix POS LV system consisting of two GPS differential antennas and an IMU. For acquiring all the sensor data and computing, we use two custom built computers, each consisting of an Intel Core i7 7700K, 4.20GHz processor and an NVIDIA GTX 1070 GPU. In order to record all the sensor data simultaneously, we equip the computing systems with 7 hard disks. We used the Kalibr Toolbox [46] for the camera-camera calibrations to estimate the intrinsic and extrinsic parameters. We present results of our scene understanding models using our AIS perception car in Chapter 4 and Chapter 6.

## 1.5 Publications

Major parts of the research presented in this thesis have been published in journal articles, book chapters, conference and workshop proceedings. A chronological overview of the corresponding publications is presented in the following list:

- A. Valada, L. Spinello, and W. Burgard. Deep Feature Learning for Acoustics-based Terrain Classification. In *Proc. of the International Symposium on Robotics Research (ISRR)*, Robotics Research, Vol. 2, Springer Proceedings in Advanced Robotics, ISBN: 978-3-319-60916-4, 2015. **Selected in Top 10 papers.**

- A. Valada, G. Oliveira, T. Brox, and W. Burgard. Towards Robust Semantic Segmentation using Deep Fusion. In *Proc. of the RSS Workshop on Limits and Potentials of Deep Learning in Robotics*, 2016.

- A. Valada, G. Oliveira, T. Brox, and W. Burgard. Deep Multispectral Semantic Scene Understanding of Forested Environments. In *Proc. of the International Symposium on Experimental Robotics (ISER)*, Springer Proceedings in Advanced Robotics book series (SPAR, volume 1), ISBN: 978-3-319-50115-4, 2016.

- A. Valada, A. Dhall, and W. Burgard. Convoluted Mixture of Deep Experts for Robust Semantic Segmentation. In *Proc. of the IROS Workshop on State Estimation and Terrain Perception for All Terrain Mobile Robots*, 2016.

- A. Valada, and W. Burgard. Deep Spatiotemporal Models for Robust Proprioceptive Terrain Classification. *The International Journal of Robotics Research (IJRR)*, 36(13-14):1521-1539, pp. 1521-1539, 2017, doi: 10.1177/0278364917727062. **ISRR Invited Journal**

- A. Valada, J. Vertens, A. Dhall, and W. Burgard. AdapNet: Adaptive Semantic Segmentation in Adverse Environmental Conditions. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

- J. Vertens*, A. Valada*, and W. Burgard. SMSnet: Semantic Motion Segmentation using Deep Convolutional Neural Networks. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

- W. Burgard, A. Valada, N. Radwan, T. Naseer, J. Zhang, J. Vertens, O. Mees, A. Eitel and G. Oliveira. Perspectives on Deep Multimodel Robot Learning. In *Proc. of the International Symposium on Robotics Research (ISRR)*, 2017.

- A. Valada*, N. Radwan*, and W. Burgard. Deep Auxiliary Learning for Visual Localization and Odometry. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

- A. Valada, and W. Burgard. Learning Reliable and Scalable Representations Using Multimodal Multitask Deep Learning. In *Proc. of the RSS Pioneers Workshop at Robotics: Science and Systems (RSS)*, 2018.

- A. Valada*, N. Radwan*, and W. Burgard. Incorporating Semantic and Geometric Priors in Deep Pose Regression. In *Proc. of the Workshop on Learning and Inference in Robotics: Integrating Structure, Priors and Models at Robotics: Science and Systems (RSS)*, 2018.

- N. Radwan*, A. Valada*, and W. Burgard. VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry. *IEEE Robotics and Automation Letters (RA-L)*, 2018, doi: 10.1109/LRA.2018.2869640.

- A. Valada, R. Mohan, and W. Burgard. Self-Supervised Model Adaptation for Multimodal Semantic Segmentation. *arXiv preprint arXiv:1808.03833, International Journal of Computer Vision* (Under Review), 2018.

Moreover, the following publications of the author of this thesis present work related to perception and localization. However, they are outside the main focus of this thesis and therefore are not covered.

- F. Boniardi, A. Valada, W. Burgard, and G. D. Tipaldi. Autonomous Indoor Robot Navigation Using Sketched Maps and Routes. In *Proc. of the Workshop on Model Learning for Human-Robot Communicationat Robotics: Science and Systems (RSS)*, 2015.

- G. Oliveira, A. Valada, W. Burgard, and T. Brox. Deep Learning for Human Part Discovery in Images. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

*Denotes equal contribution

- F. Boniardi, A. Valada, W. Burgard, and G. D. Tipaldi. Autonomous Indoor Robot Navigation Using a Sketch Interface for Drawing Maps and Routes. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

- N. Radwan, A. Valada, and W. Burgard. Multimodal Interaction-aware Motion Prediction for Autonomous Street Crossing. *arXiv preprint arXiv:1808.06887, International Journal of Robotics Research* (Under Review), 2018.

- M. Mittal*, A. Valada*, and W. Burgard. Vision-based Autonomous Landing in Catastrophe-Struck Environments. *Proc. of the IROS Workshop on Vision-based Drones: What's Next?*, 2018.

## 1.6 Collaborations

This thesis covers work that involved collaborations with other researchers. Prof. Wolfram Burgard was the supervisor of this thesis and therefore, contributed through scientific discussions. The collaborations beyond this supervision are outlined below.

- Chapter 6: The initial work on the SMSnet architecture for semantic motion segmentation was formulated in collaboration with Johan Vertens for his Master's thesis, which the author of this thesis supervised. The insights gained during the aforementioned thesis supervision influenced the subsequent improved implementations of SMSnet that the author of this thesis carried out. No parts of the experiments on SMSnet reported in this thesis intersects with the work carried out in collaboration with Johan Vertens. The results on SMSnet reported in this thesis outperforms the initial work [54]. Furthermore, the subsequent work on the SMSnet++ architecture introduced in the second part of this chapter was entirely carried out by the author of this thesis.

- Chapter 7: The VLocNet and the VLocNet++ architectures are a result of collaboration with Noha Radwan. Noha Radwan contributed to the derivation of the Geometric Consistency Loss function and the formulation of the self-supervised warping layer in the VLocNet++ architecture. The topologies of the VLocNet and VLocNet++ architectures consisting of the global localization stream, the visual odometry stream and the semantic segmentation stream as well as the weighted fusion layer, were developed by the author of this thesis. The related publications are Valada *et al.* [56] and Radwan *et al.* [34].

*Denotes equal contribution

## 1.7 Outline

This thesis is organized as follows. In Chapter 2, we introduce the main theoretical concepts and principles that are foundation for our proposed techniques. Chapter 3 presents our architectures for proprioceptive terrain classification using sound from vehicle-terrain interactions. In Chapter 4, we present our novel architectures for efficient semantic segmentation and subsequently in Chapter 5, we present our proposed self-supervised adaptive multimodal fusion frameworks for semantic segmentation. Chapter 6 presents our proposed architectures for joint semantic motion segmentation. In Chapter 7, we introduce our novel architectures for multitask learning of visual localization, semantic segmentation and odometry estimation. We review the relevant related research work at the end of each chapter in this thesis. Moreover, for each of the problems, we benchmark our proposed models on standard datasets and also provide generalization evaluations on data collected by our robots in Freiburg. Additionally, each chapter lists the corresponding qualitative results as videos that are published online. Finally, we conclude the thesis and discuss directions for future research in Chapter 8.

# Chapter 2

# Background Theory

In this chapter, we briefly discuss the theoretical concepts and principles that are the foundation of the approaches presented in this thesis. We first describe the camera model that maps points in the three-dimensional world to the two-dimensional image created by the camera. We then give an overview of feed-forward neural networks, followed by a description of the basic components of a convolutional neural network and loss functions that can be employed for training the networks.

## 2.1 Camera Model

The pinhole camera model describes the geometric relationship between the coordinates of a point in a three-dimensional space and its two-dimensional corresponding projection

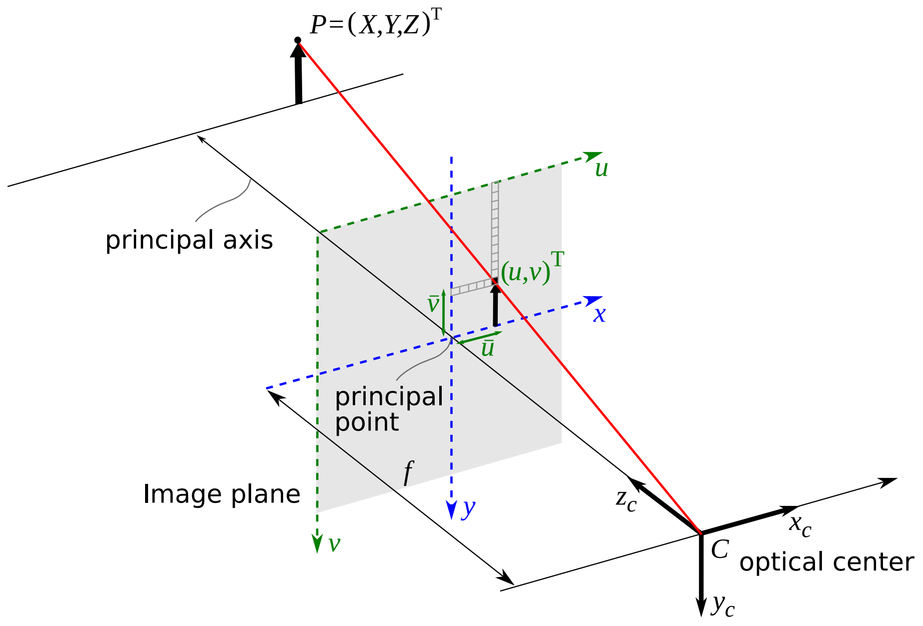


**Figure 2.1:** The pinhole camera model that describes the relationship between a 3D point $[X\ Y\ Z]^\mathsf{T}$ and its corresponding 2D projection $[u\ v]^\mathsf{T}$ onto the image plane.

onto the image plane. This geometric mapping from 3D $\rightarrow$ 2D is referred to as perspective projection. A point $P$ in the three-dimensional world can be mapped into a two-dimensional point on the image $P'$ using the projective transformation $\pi$ defined as

$$\pi : \mathbb{R}^3 \rightarrow \mathbb{R}^2 \tag{2.1}$$

The geometry related to the mapping of a pinhole camera is illustrated in Figure 2.1. The pinhole camera allows only a single ray from each point on the object to pass through an infinitely small aperture. The center of the perspective projection where all the rays intersect is called as the optical center $C$ and the line perpendicular to the image plane passing through the optical center is called the principal axis or the optical axis. In addition, the intersection point of the image plane with the optical axis is called the principal point, while the distance from the optical center to the principal point is the focal length $f$ of the camera. For convenience, we put the image plane behind the optical center as this avoids the need to flip the image coordinates about the origin. The standard coordinate system of the camera has its origin at the center of the projection and its Z axis along the optical axis, perpendicular to the image plane. The image plane is located at $Z = f$ in the camera coordinate system. A point $P \in \mathbb{R}^3$ on an object with coordinates $[X\,Y\,Z]^\intercal$ will be projected at a pixel position $[u\,v]^\intercal$ in the image plane. The relationship between these two coordinate systems is given by

$$\pi[X\,Y\,Z]^\intercal \rightarrow [u\,v]^\intercal \tag{2.2}$$

$$u = \frac{Xf}{Z} + c_x \tag{2.3}$$

$$v = \frac{Yf}{Z} + c_y \tag{2.4}$$

As the origin of the pixel coordinate system is defined at the top-left corner of the image, the 2D points are offset by a translation vector $(c_x, c_y)$ to account for the misalignment between the optical center and the origin of the image coordinates. In Eq. (2.3) and Eq. (2.4), it was implicitly assumed that the pixels of the image sensor are square. In the case when the pixels on the sensor chip of the camera are rectangular of size $1/s_x$ and $1/s_y$, different scaled focal lengths $f_x = s_x f$ and $f_y = s_y f$ are employed, where, $s_x$ and $s_y$ are in units of horizontal and vertical pixels per meter respectively. The 2D point can also be back-projected to the 3D space, given the depth $Z$ and the camera parameters. We can express this transformation as

$$\pi^{-1}[u\,v\,Z]^\intercal \rightarrow [X\,Y\,Z]^\intercal \tag{2.5}$$

$$X = \frac{u - c_x}{f_x} Z \tag{2.6}$$

$$Y = \frac{v - c_y}{f_y} Z \tag{2.7}$$

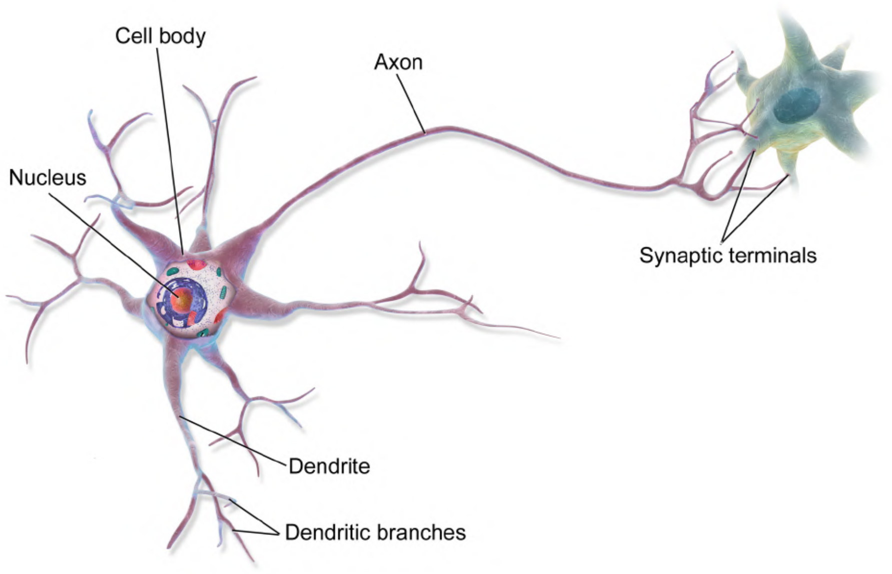


**Figure 2.2:** Depiction of the biological neuron cell [64].

where the intrinsic camera parameters ($f_x$,$f_y$, $c_x$, $c_y$) constituting the focal length and the principal point can be obtained by standard camera calibration. We employ the aforementioned equations in the proposed self-supervised warping layer in Chapter 7.

## 2.2 Feed-Forward Neural Networks

Neural networks, also called as artificial neural networks were originally inspired by the goal of modeling biological neural systems. About 86 billion neurons make up the human nervous system which are connected via approvimately $10^{14}$ to $10^{15}$ synapses to form an enormous communication network that manifests complex activation patterns. We first describe the biological neural cell at the high-level that inspired the modeling of artificial neural networks.

The biological neural cell, also known as the neuron shown in Figure 2.2 is the basic computational unit of the brain. Each neuron receives electrical signals through its dendritic branches to the dendrites that are directly connected to the main body of the cell. The neuron accumulates signals from from multiple sources and once the signal reaches a certain threshold, it generates a pulse called the action potential. However, the signals received via the dendritic branches are not uniform as they depend on several factors such as the strength of the signal generating neuron and the number of redundant connections. The action potential propagates through the axon to the synaptic terminals that are then connected to the dendritic branches of other neurons. The neurons learn to adapt to new situations by varying the threshold of the action potential.

The computational model of an artificial neuron shown in Figure 2.3 is a highly simplified

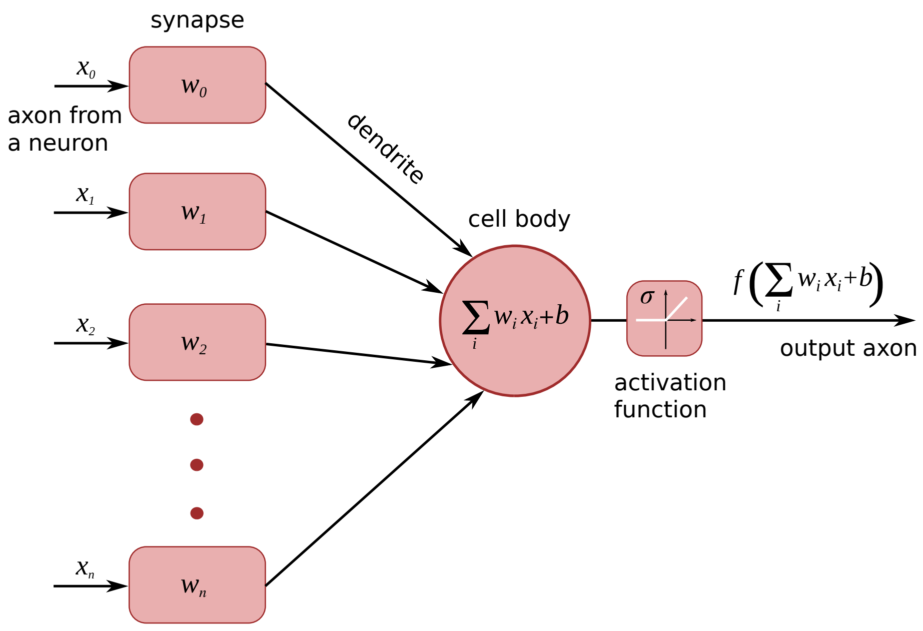


**Figure 2.3:** Depiction of the artificial neuron cell inspired by the biological model.

model of the biological neuron. It has $n$ input values $x_i \in \mathbb{R}$ that propagate through the axons and interact multiplicatively with the dendrites of other neurons based on the synaptic strength at that synapse $w_i$. The synaptic strengths or the weights $w$ are learnable and they control and propagation of signals from one neuron to another. The bias $b$ acts as another weight to the constant input. All the signals are first summed at the cell body and when the sum reaches a certain threshold, the neuron fires with the help of a non-linear activation function $\sigma$. We model the artificial neuron as

$$f(x) = \sigma\left(\sum_{i=0}^{N} w_i x_i + b\right) \tag{2.8}$$

We can also write Eq. (2.8) in a vectorized form considering the inputs as a vector $\mathbf{x}$ with the same activation function $\sigma$ and the sum as a scalar product or a matrix-vector multiplication in the case of multiple neurons with the same inputs. Neural Networks are modeled as collections of artificial neurons that are connected in an directed acyclic graph, also called feed-forward neural networks. They can be organized into layers of neurons, where the outputs of one layer become inputs to the following layer. Cyclic connections are not allowed as they can lead to an infinite loop and yield undefined network outputs. The most common layer is the fully-connected layer, also called the inner-product layer in which neurons within the same layer are not connected but neurons between two adjacent layers are fully pairwise connected. Figure 2.4 shows an example of a three layer neural network consisting of two hidden layers. The term hidden layer was coined as its values are not observed in the training set.

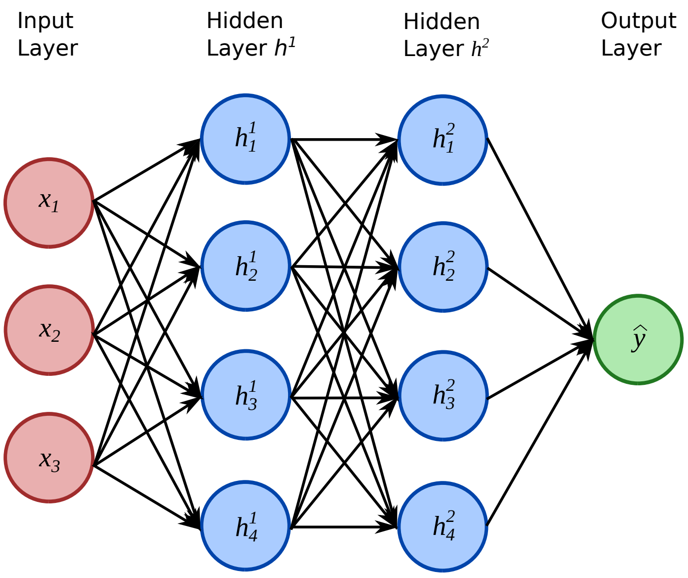


**Figure 2.4:** Example of a three-layer feed-forward neural network architecture with three inputs, two hidden layers and one output layer.

We denote the network shown in Figure 2.4 as a three-layer network, as the input layer is usually not counted. All the neurons in the same layer have the same properties such as the activation function and the connections do not exist within a layer. In the aforementioned example, there are a total of 9 neurons with 32 weights and 9 biases. This amounts to a total of 41 learnable parameters. Note that the output layer does not have an activation function as the number of units usually represent the number of classes for classification tasks or a real-valued target in case of regression. We represent each layer $l$ in this network as

$$f^l(x) = \sigma^l\left(\mathcal{W}^{(l)}\mathbf{x} + \mathbf{b}^{(l)}\right) \tag{2.9}$$

where $\mathcal{W}^{(l)} \in \mathbb{R}^{m\times n}$ is the weight matrix of layer $l$, $\mathbf{b}^{(l)} \in \mathbb{R}^m$ is the corresponding bias vector and $\sigma^l$ is the activation function. The weights and biases are commonly defined with the variable $\theta$ and the output of the network then be represented as $\hat{\mathbf{y}} = f\,(\mathbf{x};\theta)$. The common activation functions that are usually employed include tanh, sigmoid, Rectified Linear Unit (ReLU) [65] and Exponential Linear Unit (ELU) [66]. The tanh non-linearity squashes a real-valued number to the $[-1, 1]$ range, while sigmoid reduces it to the $[0, 1]$, ReLU thresholds at zero and ELU are similar to ReLUs except for negative inputs. These activation functions can be represented as

$$\tanh(x) = 2\sigma(2x) - 1 \tag{2.10}$$

$$\sigma(x) = \frac{1}{(1 + e^{-x})} \tag{2.11}$$

$$relu(x) = \max(0, x) \tag{2.12}$$

$$elu(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - a) & \text{if } x \leq 0 \end{cases} \tag{2.13}$$

We primarily use ReLUs in the architectures presented in this thesis. However, we also employ the sigmoid activation function and ELUs in the networks that we introduce in Chapter 6 and Chapter 7 respectively. We further detail the loss functions that can be used for training the networks in the following sections.

## 2.3 Convolutional Neural Networks

In the previous section, we briefly introduced neural networks in which adjacent layers are fully connected to one another and every neuron in the network is connected to every neuron in the adjacent layers. However, it can be observed that the number of connections grows quadratically with the size of each layer and the number of trainable weights increases correspondingly. Moreover, such networks do not model the spatial relationship in the input data due to the feed-forward interconnection and as a consequence there is no translation invariance, which is a critical factor to model, especially while learning visual concepts from images. As a solution to these problems, Convolutional Neural Networks (CNNs) that consist of several convolutional layers with local connections, followed by pooling layers that provide translational invariance and one or more fully-connected layers can be employed. CNNs are easier to train and have fewer parameters than standard neural networks with the same number of hidden units. In the following sections, we discuss the basic constituting components of a CNN architecture and the loss functions that can be employed for training the models.

### 2.3.1 Architecture Overview

A convolutional neural network consists of several convolutional and pooling layers, optionally followed by fully-connected layers or deconvolutional layers. The input to a convolutional layer is an image $\mathbf{x}$ of dimensions $H \times W \times C$ such that $\mathbf{x} \in \mathbb{R}^H \times \mathbb{R}^W \times \mathbb{R}^C$, where $H$ is the height, $W$ is the width and $C$ is the number of channels. Each convolutional layer has $k$ filters or kernels of size $F \times F \times Q$, where $F$ is smaller than the dimensions of the input image and $Q$ varies for each kernel. This restricts the connections to a local spatial neighborhood of the input where each filter is convolved with the image to yield $K$ feature maps of height $= H - F + 1$ and width $= W - F + 1$. Restricting the connections enables a reduction of the number of neuron inputs and weights substantially. Subsequently, an additive bias and a nonlinearity is applied to each feature map. The feature maps are then downsampled using different pooling layers along the spatial dimensions. Depending

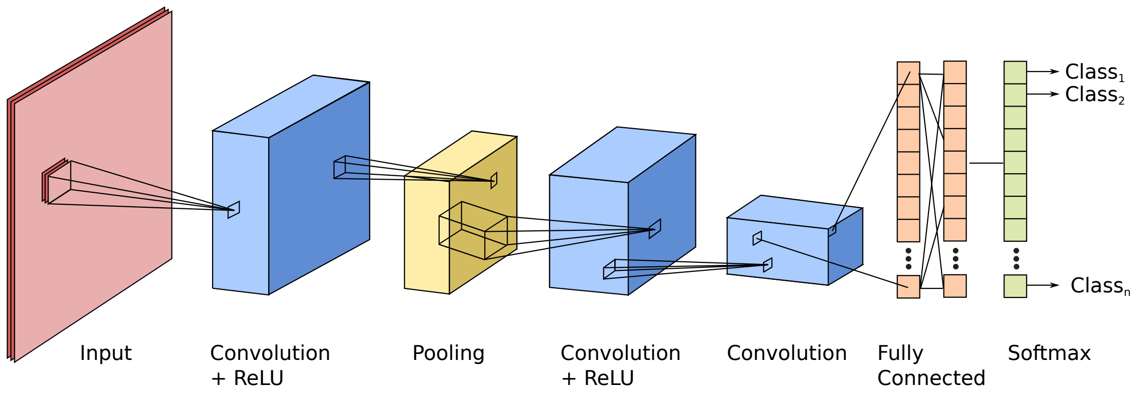


**Figure 2.5:** Example architectural topology of a convolutional neural network. The convolution and pooling layers learn about the local structure in the input, while the fully-connected layers learn at a more abstract level by integrating global information across the entire input.

upon the task at hand, fully-connected layers or deconvolution layers with a softmax layer follows after alternating convolution and pooling layers to yield the class scores. Figure 2.5 shows an example topology of a CNN consisting of three convolution layers, one pooling layer, two fully-connected layers and a softmax layer to yield the classification scores. In the following sections, we describe the individual layers and their corresponding hyperparameters.

### 2.3.2 Layers

There are a number of different layers that have been proposed over the years. In this section, we describe the fundamental layers that we employ to build our proposed architectures that we present in this thesis.

**Convolution Layer:** The convolutional layer is one of the building blocks of CNN and it consists of learnable filters having small spatial dimensions while extending through the entire depth of the input tensor. Each filter is convolved over the width and height of the input tensor to yield a two-dimensional activation map that represents the responses of that filter at a spatial location. The activation maps are then stacked along the depth dimension to yield the output tensor. An important aspect of convolution layers that differs from fully-connected layers is that each neuron is only connected to a local region of the input tensor whose spatial extent is indicated by the receptive field, while its depth extends through the entire input tensor. There are three hyperparameters that control the size of the output tensor, namely, depth, stride and zero-padding, where depth denotes the number of filters that the convolution layer will use, stride denotes the number of pixels that the filter should move (stride 1 indicates that the filter will move one pixel at a time), and zero-padding pads the input tensor with zeros around the border. The spatial size of the output tensor can be computed as a function of the input tensor size ($W$), size of the receptive field ($F_w, F_h$), stride ($S$) and the zero-padding ($P$) as

$$OC_w = \frac{W - F_w + 2P}{S_w} + 1, \tag{2.14}$$

$$OC_h = \frac{W - F_h + 2P}{S_h} + 1, \tag{2.15}$$

where $OC_w$ is the output width and $OC_h$ is the output height.

A convolution layer conserves parameters by sharing the same weight vector and bias with all the neurons in the same two-dimensional slice of the depth. Therefore, in the forward-pass, each depth slice is computed as a convolution of the weights of the neurons with the input tensor and during backpropagation, the gradient of every neuron in the tensor is added across each depth slice and only a single set of weights are updated for each slice.

**Atrous Convolution Layer:** Pooling and striding that is commonly employed in CNNs, decreases the spatial resolution of the feature maps. For tasks such as semantic segmentation, this decimates the details which cannot be completely recovered even using deconvolution or transpose convolution layers. To alleviate this problem, atrous convolutions also known as dilated convolutions [67] can be used to enlarge the field of the filter thereby effectively capturing larger context. Moreover, by using atrous convolutions of different dilation rates, we can aggregate multiscale context. While the standard convolution filters are contiguous, atrous convolution is equivalent to convolving with a filter which has zeros inserted between two consecutive filter values across the spatial dimensions. Let $F : \mathbb{Z}^2 \rightarrow \mathbb{R}$ be a discrete function, $\Omega = [-r, r]^2 \cap \mathbb{Z}^2$, $k : \Omega_r \rightarrow \mathbb{R}$ be a discrete filter of size $(2r + 1)^2$ and $r$ be a dilation rate. The atrous convolution $*_r$ can be defined as

$$(F *_r k)(\mathbf{p}) = \sum_{(\mathbf{s}+r\mathbf{t}=\mathbf{p})} F(\mathbf{s})\, k(\mathbf{t}), \tag{2.16}$$

where the dilation rate $r$ denotes the stride with which we sample the input signal. Therefore a larger dilation rate indicates a larger receptive field. The standard convolution can be considered as a special case by setting $r = 1$.

**Pooling Layer:** The pooling layer is employed after the convolution layer to downsample each depth slice of the input tensor independently using different operations depending on the type of pooling. As convolution layers restrict the neuron inputs to a local neighborhood, employing only a series of convolution layers makes the network loose the global context. Pooling provides translation invariance to the information in the input tensors to a certain extent by aggregating information globally. It reduces the the number of parameters and computational operations in the network. The two hyperparameters, the filter size ($F$) and the stride ($S$) control the size of the output tensor. The different types of pooling operations include max, average, L2-norm and stochastic. In practice, max pooling is most commonly employed compared to the other methods. The spatial size of the output tensor can be computed as function of the input tensor size ($W$), size of the receptive field ($F$) and the stride stride ($S$) as

$$OP_w = \frac{W - F}{S} + 1, \tag{2.17}$$

$$OP_h = \frac{W - F}{S} + 1, \tag{2.18}$$

where $OP_w$ is the output width and $OP_h$ is the output height.

During the forward-pass, it is common to keep track of the index of the resulting activation so that during the backpropagation, the gradients can be routed through it. Architectures that are designed for certain tasks such as semantic segmentation and adversarial learning do not incorporate pooling layers as they cause the network to loose the exact position information of the features. Instead, they use a larger stride in the convolution layer to perform the downsampling.

**Fully-connected Layer:** Neurons in fully-connected layers have connections to every neuron in the previous later. This is similar to the standard neural network, also called the multi-layer perceptron that we described in Section 2.2. The activations of fully-connected layers are computed with matrix multiplication, followed by a bias offset. The only difference between a fully-connected layer and a convolution layer is that the convolution layers have local connectivity and parameters are shared for the neurons in the same depth slice, however their functional form is identical. Fully-connected layers are often used towards the end of CNN architectures to aggregate information from all the feature maps and output a vector that corresponds to the number of objects to classify. We only use fully-connected layers in the architectures that we present in Chapter 3 and Chapter 7 in which we propose a network for classification and regression respectively.

**Deconvolution Layer:** Deconvolution layers, also called transposed convolutions or fractionally strided convolutions, transform the input tensor in the opposite direction of standard convolutions such as mapping the feature maps from low-dimensional space to a higher-dimensional space while maintaining the connection patterns of the convolution. In practice, the deconvolution operation is implemented as taking the gradient of the convolution with respect to its inputs. In other words, the only difference between the standard convolution and the deconvolution is how the forward and backward passes are computed. Deconvolution layers are often employed to obtain a mapping from the convolved values to the original inputs. Similar to convolution layers, three hyperparameters control the size of the output tensor including depth, stride and zero-padding. The spatial size of the output tensor can be computed as a function of the input tensor size ($W$), size of the receptive field

$(F_w, F_h)$, stride stride used ($S$) and the zero-padding. Deconvolution layers are extensively used in architectures that are designed for semantic segmentation, depth estimation and optical flow estimation. As opposed to naively resizing the outputs to a larger dimension, employing a deconvolution layer with learnable parameters yields an upsampled output with finer details.

**Dropout Layer:** CNNs are plagued by the overfitting problem where the weights of the network are excessively tuned for the training data which causes the network to perform poorly on new examples that have not been seen before. In order to alleviate this problem the dropout layer can be employed as a regularization technique to randomly *drop* a set of activations by setting them to zero while training the network. This forces the network to learn redundant connections and therefore improves the generalization of the network. Dropout layers are often used towards the end of the network on the convolution or fully-connected layers but certain architectures also employ them on max pooling layers to create image noise that acts as data augmentation. More recently [68], dropout has also been employed during test-time to obtain the predictive mean and predictive uncertainty by Monte Carlo averaging of stochastic forward passes through the model.

**Network-in-Network Layer:** A $1 \times 1$ convolution layer was initially called as the network-in-network layer by Lin *et al.* [69]. Unlike in standard applications of the convolution layer where the receptive fields are large, it employs a convolution layer with the filter size of $1 \times 1$. As convolutions operate on three-dimensional tensor that span the entire depth of the input, a $1 \times 1$ convolution layer performs a $N$-dimensional element-wise multiplication, where $N$ is the depth of the input tensor into the layer. We extensively employ the $1 \times 1$ convolution layer in the architectures proposed in this thesis.

### 2.3.3 Architectural Network Design

Thus far, in the previous sections, we introduced the basic building blocks a CNN. In this section, we discuss how these layers are often stacked together to form entire architectures. Classical CNNs usually stack convolution layers with ReLU, followed by a pooling layer and then repeat this pattern until learned features are highly discriminative and spatially small in size. Multiple convolution layers can also be stacked before employing a pooling layer to learn more complex features before loosing the location information. As opposed to employing one convolution layer with a large receptive field, a stack of convolution layers with smaller filters are more beneficial as a stack of convolutions with non-linearity makes the learned features more discriminative and consumes lesser number of parameters. For example, a stack of three $3 \times 3$ convolution layers makes the effective receptive field at the third convolution layer $7 \times 7$ with $27C^2$ parameters with $C$ as the number of channels in the convolution layers, while in comparison, a single $7 \times 7$ convolution layer consumes $49C^2$ parameters. However, a disadvantage being that it requires more memory to store the results of the intermediate convolutions while performing backpropagation. For classification and regression architectures, two or more fully-connected layers are added after the stack of convolution and pooling layers to aggregate information and yield the final object class scores or regress the outputs.

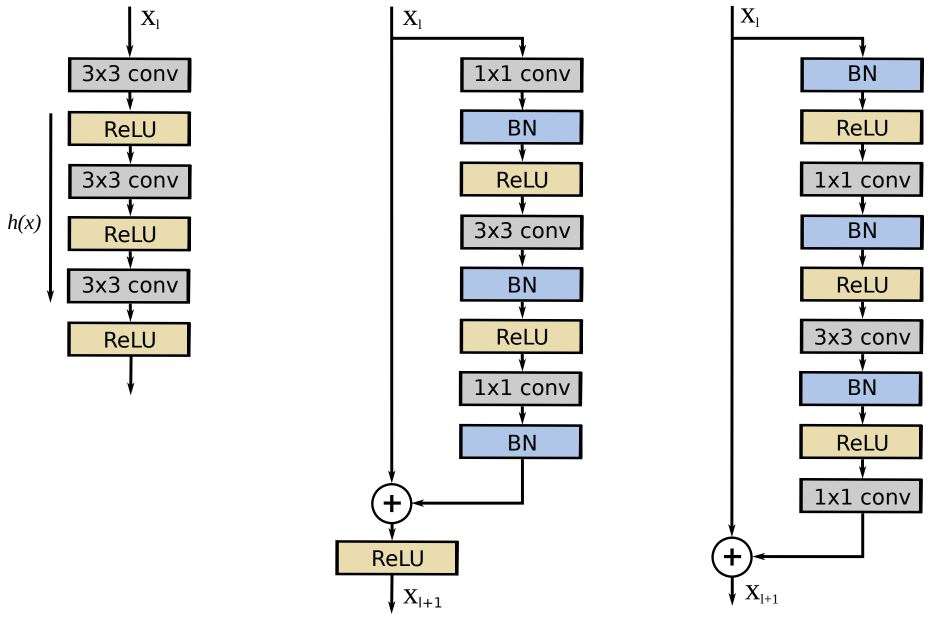
However, recent architectures that have achieved state-of-the-art performance as the Inception architectures [70], residual networks [27] and dense networks [71] have moved away from this conventional paradigm of employing linear list of layers. They have a more intricate connectivity structure. In most of the architectures that we present in this thesis, we employ the residual learning framework. Therefore, we give a brief overview of residual networks in the rest of this section.

**Residual Networks:** The residual network architecture won the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) challenge [23] in 2015. It is generally believed that deeper networks are capable of learning more complex functions and correspondingly have a larger representational capacity. However, in practice this does not appear to be true due to two main challenges, vanishing gradients and the optimization difficulty. Vanishing gradients occur when the gradient becomes very small as it reaches the earlier layers resulting in hampering the convergence of the network. Although better parameter initialization techniques and batch normalization enables deeper networks to converge, their performance is still substantially lower than their shallower counterparts. Secondly, introducing more layers increases the number of parameters that have to be trained which makes the optimization of the network harder and leads to larger training errors. The ResNet architecture addresses these aforementioned challenges that have prevented training of very deep networks.

The main difference between ResNets and CNN architectures that employ a linear list of layers is that ResNets provide a clear path for the gradients to propagate to earlier layers of the network. A standard CNN stream shown in Figure 2.6 (a) maps the input to the output with a non-linear function $h(x)$, where each layer is expected to learn distinct feature maps. ResNets introduce the residual units in which the intermediate units learn a residual function with reference to the input tensor. Residual architectures contain multiple such units serially connected to each other, in addition to a shortcut connection that runs parallel to each unit. These shortcut connections enable the gradient to flow easily through them which results in faster training. The standard residual unit shown in Figure 2.6 (b) can be expressed as

$$\begin{aligned}\mathbf{y}_l &= h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l),\\ \mathbf{x}_{l+1} &= f(\mathbf{y}_l),\end{aligned} \tag{2.19}$$

where $\mathcal{F}$ is the residual function (convolutional layers in the residual unit), $\mathbf{x}_l$ is the input feature to the $l$-th unit, while $\mathbf{x}_{l+1}$ is the output, $\mathcal{W}_l = \{\mathcal{W}_{l,k}|_{1\leq k\leq K}\}$ are the set of weights



(a) Standard CNN stream (b) Standard residual unit (c) Full Pre-activation residual unit

**Figure 2.6:** Topology of the standard CNN stream in comparison to the original residual unit [27] and the improved full pre-activation residual unit [72].

and biases of the $l$-th residual unit and $K$ is the number of layers in the unit. The function $f$ is a non-linearity such as a ReLU and the function $h$ is set to the identity mapping $h(\mathbf{x}_l) = \mathbf{x}_l$. Note that we described the bottleneck structure of the residual unit that enables training of deeper networks, as opposed to the two layer design that contains consecutive $3 \times 3$ convolution layers. Instead, the bottleneck structure consists of three layers $1 \times 1$ convolution, $3 \times 3$ convolution and a $1 \times 1$ convolution. The first $1 \times 1$ convolution layer reduces the number of feature maps, making the $3 \times 3$ convolution have a smaller input/output dimension, while the last $1 \times 1$ convolution restores the dimensions to that of the input. There are two types of residual units, identity shortcuts and projection shortcuts. Identity shortcuts are used when the input and outputs are of the same dimensions and the projection shortcut is used when the dimensionality of the output is different from the input so a $1 \times 1$ convolution layer to used on the input tensor to match the dimensions of the output tensor.

The activation $f$ in the original residual unit described above affects both paths in the next unit. Therefore, He *et al.* [72] proposed an improved residual unit where the activation $\hat{f}$ only affects the path $\mathcal{F}$, which can be defined as

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\hat{f}(\mathbf{x}_l), \mathcal{W}_l), \tag{2.20}$$

This pre-activation residual unit shown in Figure 2.6 (c) enables the gradient to flow through the shortcut connection to any unit $l$ without any obstruction. Batch normalization is extensively used in residual networks that help in regularizing the network. However, in the standard residual unit, the batch normalization is applied before the addition which causes the output of the residual unit be un-normalized and consequently, the input to the next residual unit is also not normalized. In the full pre-activation residual units structure, the input is normalized at the beginning of the identity unit which makes the convolutions always receive the normalized inputs. These full pre-activation residual units have been demonstrated to reduce overfitting, improve the convergence and also yield improved performance. The initial architectures that we propose in this thesis employ the standard residual unit, while the improved architectures employ the full pre-activation residual units.

### 2.3.4 Loss Functions

Loss functions are used to supervise the training process of a neural network. In this thesis, we primarily use two different loss functions according to the task at hand. In Chapter 3, 4, 5, and 6, we employ the softmax function that uses the cross-entropy loss for training our proposed architectures that perform classification and pixel-wise classification. Let $f(x_i)$ be the activations of the output layer for the CNN, $y_i$ be the groundtruth label, $C$ is the number of object classes to classify, and $N$ is the number of training examples. Let the weights and biases of the last layer of the CNN be $W$ and $b$. The softmax classifier that uses the cross-entropy loss can be defined as

$$\mathcal{L}_s = -\frac{1}{N}\sum_{i=1}^{N} y_i \log \frac{\exp\left(W_{yi}^{\mathsf{T}} f(x_i) + b_{yi}\right)}{\sum_{j=1}^{C} \exp\left(W_j^{\mathsf{T}} f(x_i) + b_j\right)} \tag{2.21}$$

While, for regressing the 6-DoF poses in the architectures that we propose in Chapter 7, we use the $L^2$-norm loss function. It minimizes the sum of the square of the differences between the target and the estimated value as

$$\mathcal{L}_{L^2} = \sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{2.22}$$

where $y_i$ is the groundtruth and $f(x_i)$ is the estimated output of the model. The protocol that we employ for training our architectures is described in the respective chapters.

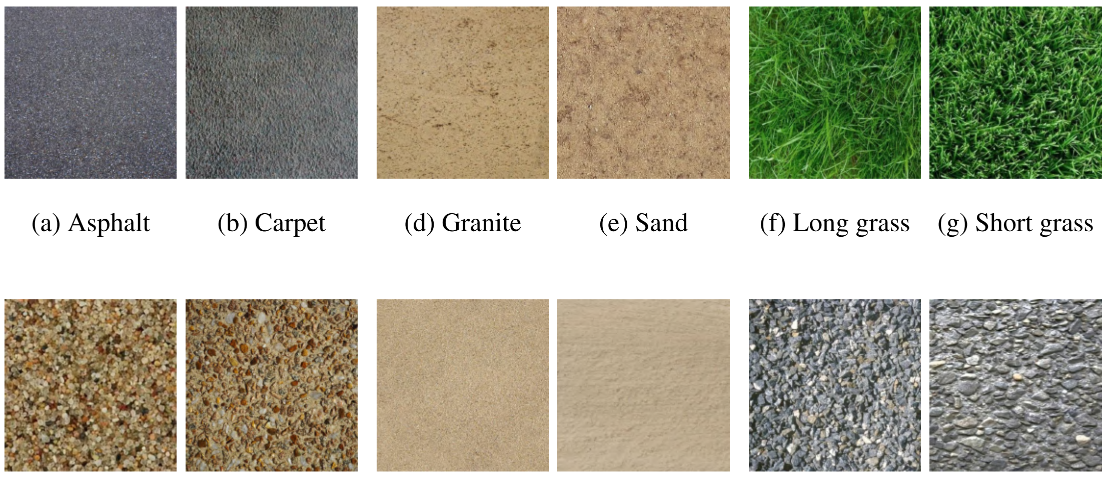
# Chapter 3

# Proprioceptive Terrain Classification

**In this chapter, we address the problem of terrain classification using sound from vehicle-terrain interactions. Terrain classification is a critical component of any autonomous mobile robot system operating in environments with unknown spatially-varying terrains. Most existing techniques are predominantly based on using visual features which makes them extremely susceptible to appearance changes and occlusions. Therefore, relying solely on them inhibits the robot from functioning robustly in all conditions. As a solution to this problem, we propose an approach that uses vehicle-terrain interaction sounds as a proprioceptive modality to classify a wide variety of terrains. We present a novel convolutional neural network architecture to learn deep features from spectrograms of audio signals. As audio signals are inherently sequential and temporal, we also present a recurrent variant of the model that incorporates Long Short-Term Memory units to learn the complex temporal dynamics. Experiments on two extensive datasets collected with different microphones on various indoor and outdoor terrains demonstrate that our networks achieve state-of-the-art performance compared to existing techniques. In addition, we present experimental results in adverse acoustic conditions with high ambient noise and propose a noise-aware training scheme that enables learning of more generalizable models which are essential for robust real-world deployments.**

## 3.1 Introduction

As robots are increasingly being employed for tackling complex tasks in unknown environments such as in mining, forestry, space exploration as well as search and rescue, they face an immense amount of perceptual challenges. Mobile robots in particular should have



(a) Asphalt (b) Carpet (d) Granite (e) Sand (f) Long grass (g) Short grass

(h) Beach sand (i) Concrete (j) Sand (k) Tile (l) Gravel (m) Concrete

**Figure 3.1:** Ambiguous appearance of terrains that make traversability analysis using optical sensors extremely challenging. Visually similar terrains having different traversability properties are shown in pairs.

the ability to distinguish the terrain that they traverse on to determine the corresponding trafficability. Failing to adapt their navigation strategy according to the terrain including the control, driving style and planning strategy, can lead to disastrous situations. For instance, if a robot employs the same speed that it uses to traverse on asphalt to traverse on sand, it would lead to entrenchment of the wheels. Similarly, several such scenarios may be encountered while operating in environments with unknown spatially-varying terrains, where the wheels of the robot may experience slippage and entrenchment. Often situations may also be encountered, where a rocky terrain can act as a non-geometric hazard and can cause mechanical damage to the robot. Moreover, the predominately adopted extroceptive sensor-based approaches such as using cameras are highly susceptible to failure due to the similar visual appearance of terrains as shown in Figure 3.1. Appearance of terrains also often rapidly change with varying weather conditions including shadows due to lighting changes, dampness due to rain and camouflaging with leaves, which make reliable terrain classification using vision-based approaches extremely challenging.

Over the years, these factors have motivated the development of techniques that exploit other extroceptive modalities for terrain classification such as depth and remission values from Lidars [73, 74], as well as proprioceptive modalities such as vibrations induced on the vehicles body [75], vehicle-terrain interaction sounds [47, 76, 77, 78] and the change in acceleration perpendicular to the ground [79, 80]. Each of these approaches have their own benefits and drawbacks: optical sensors perform well in the presence of good illumination but they are drastically affected by visual appearance changes. Classification with RGB images is typically performed using color and texture features extracted from images [81]. The use of texture-based image descriptors such as local ternary patterns [82] have also been explored. In such approaches, features extracted using local ternary patterns are used on sequences of images and classified using Recurrent Neural Network (RNN) configurations [83]. Lidars have also been extensively used for traversability analysis where features extracted often include statistics on remission values, roughness and slope. Recently, semi-supervised learning approaches [74, 84] that only require partially labeled Lidar data have been proposed. However, they are not suitable for fine-grained terrain classification where two or more terrains may have the same degree of coarseness.

As visual images and Lidar data provide complementary information, techniques have also been proposed that combine features from both modalities. The combined feature set includes image features such as color and texture, as well as geometric features from Lidar data such as surface normals, curvature, ground height, point feature histograms, linearity and planarity [85, 86]. CNN-based approaches for both near-range and far-range terrain classification using stereo and RGB images have been demonstrated during the Learning Applied to Ground Robots (LAGR) program [87, 88] of the Defense Advanced Research Projects Agency (DARPA). These approaches have demonstrated substantial robustness in the real-world by combining both supervised and unsupervised learning using deep hierarchical networks. However, their scope was limited to segmenting navigable areas from obstacles and not distinguishing among the various types of navigable terrains.

Humans from an early age are self-trained to leverage the sound of their footsteps to identify the terrain they there are walking on, especially in darkness. Similarly, robots can exploit the vehicle-terrain interaction sounds to characterize the terrain. This is extremely beneficial to mobile robots that perform repeated long-term tasks in unknown environments as most of the robot's energy is spent on mobility and the rate of expenditure is a function of the terrain over which its driving [89]. By predicting the traversability as the robot is driving over a terrain, we can build spatial traversability maps based on the robot's experience which can be used to plan minimal-energy paths. Moreover, acoustics-based approaches are robust to the disturbances that affect optical or active sensors which enables us to employ them as complementary classifiers or for fusion and self-supervision of classifiers based on other modalities. Although several proprioceptive terrain classifiers have been proposed over the last decade, they have failed to gain widespread adoption. This can be attributed to the following:

- Manually designing feature sets that perform well for every imaginable real-world condition is tedious and impractical.
- Existing techniques have slow run time making them unusable for real-time robotics applications.
- Most techniques require specific hardware setups that are difficult to replicate.
- Approaches often lack reliability in real-world scenarios.

In this chapter, we propose a novel multiclass proprioceptive terrain classification ap-

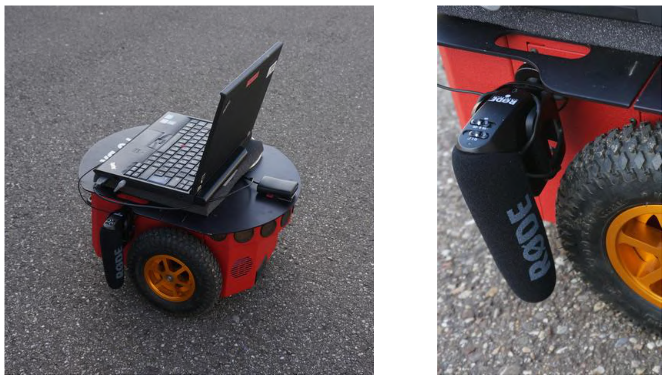


**Figure 3.2:** The Pioneer P3-DX platform equipped with a shotgun microphone with an integrated shock mount that we use in our experiments. The microphone is mounted close to the wheel with the supercardioid pointing towards the tire-terrain interface.

proach that overcomes these impediments by using sound from vehicle-terrain interactions to classify both indoor and outdoor terrains. We propose two DCNN architectures: one that considers a single time window for classification and a recurrent variant that incorporates Long Short-Term Memory (LSTM) units to capture the temporal dynamics of the signal. Both our architectures incorporate our proposed Global Statistical Pooling (GSP) strategy in which three temporal pooling methods are combined to achieve the time-series representation learning. In addition, our recurrent model forms compositional representations in the time domain, similar to convolutional models that form compositional representations in the spatial domain. The structure of our proposed network exploits the temporal information in each clip as well as in the transitioning clips. In order to evaluate the efficacy of our approach, we gathered two extensive vehicle-terrain interaction datasets consisting of data from nine different indoor and outdoor terrains. The first dataset was collected using a mobile robot equipped with a shotgun microphone as shown in Figure 3.2 and the second dataset was collected using a mobile phone microphone. The rationale behind using two different microphones was to evaluate the generalizability of the model to different hardware setups and to demonstrate that even inexpensive microphone can be effectively utilized with our proposed system.

Although DCNN-based approaches are more robust to environmental noise than classical approaches that employ handcrafted features, their performance still degrades in high ambient noise environments. Noise sensitivity is strongly correlated to the capability to operate in different real-world environments. Moreover, there are two different types of noise sources in our application; ambient noise and the noise from the drive motors of the robot. As the motor noise is captured in the training data, the model learns to account for this noise corruption. In order to make our model tolerant to ambient noise and generalize effectively to different real-world environments, we present a noise-aware training scheme. Our training scheme randomly injects ambient noise signals from the

Diverse Environments Multichannel Acoustic Noise Database (DEMAND) [90] at different Signal-to-Noise (SNR) ratios. We demonstrate that our network trained using our noise-aware scheme shows substantial improvement in the robustness of the model to adverse ambient noises. The approach presented in this thesis is the first proprioceptive terrain classification technique to be able to robustly classify nine different indoor and outdoor terrains with a classification rate of over 80 Hz.

Concretely, we make the following contributions in this chapter:

- A novel convolutional neural network architecture for terrain classification from vehicle-terrain interaction sounds that operates on a single audio clip of fixed length.
- A recurrent convolutional neural network architecture for terrain classification from vehicle-terrain interaction sounds.
- A new Global Statistical Pooling (GSP) scheme for aggregating statistics of learned temporal features.
- A noise aware training scheme that substantially increases the generalizability of the model to real-world environments with different ambient noises.
- Thorough empirical evaluations of our proposed system on over 6 h of audio data using two different audio acquisition systems and in seven different environments having adverse acoustic conditions.

The remainder of this chapter is organized as follows. In Section 3.2, we detail our proposed network architectures and the noise-aware training strategy. We then describe our data collection methodology in Section 3.3, followed by extensive experimental evaluations that demonstrate the efficacy of our approach in Section 3.4. Finally, we discuss the related work that has been previously done using acoustics and other proprioceptive sensors in Section 3.5, before concluding the chapter in Section 3.6.

## 3.2 Technical Approach

In this section, we describe our approach to terrain classification using vehicle-terrain interaction sounds. Recent work has shown that learning audio features from simple transformations such as spectrograms are more effective than directly leaning from raw audio waveforms [26, 91]. Therefore, we first transform the vehicle terrain interaction signals into their spectrogram representation, and then use them as inputs to our network for feature learning and classification. Figure 3.3 depicts the preprocessing that we perform on the training set. In the following sections, we first describe the spectrogram transformation technique that we apply on the raw audio waveforms. We then detail our proposed TerrainNet architecture that operates on spectrograms of independent audio frames of vehicle-terrain interactions, followed by our recurrent TerrainNet++ architecture which incorporates LSTM units to model the sequential nature of the signal and more effectively exploit the temporal dynamics.

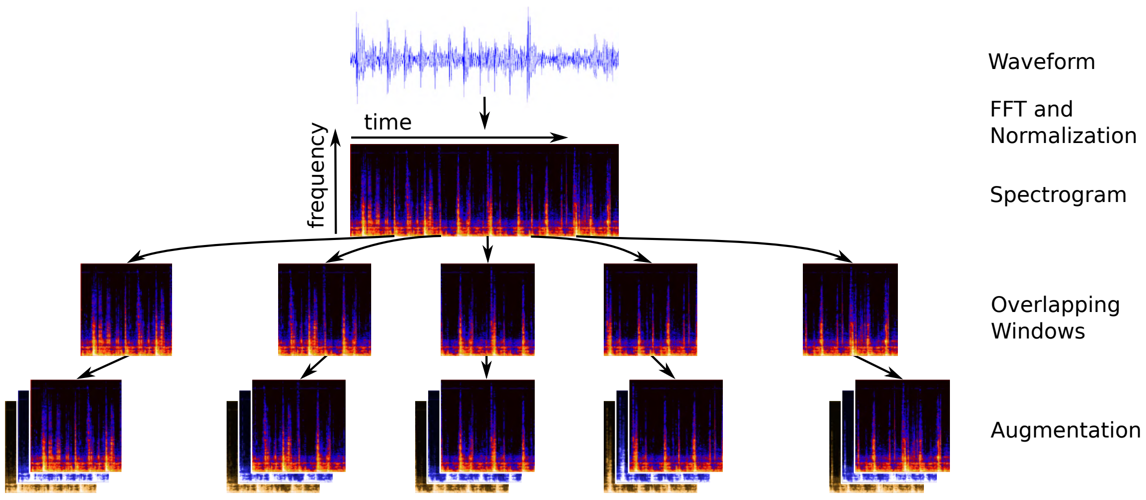


**Figure 3.3:** Overview of the preprocessing pipeline that we employ on the training data. The raw audio waveform is first transformed into its spectrogram representation and then a series of waveform augmentations is applied in order to enable the learned representation to be invariant to commonly observed signal perturbations.

### 3.2.1 Spectrogram Transformation and Augmentation

Unlike speech decoding or handwriting recognition tasks that require specific alignment between the audio and transcription sequences, terrain classification from acoustic signals does not require a specific target relation for each frame. Therefore, we use individual audio clips of fixed length as new samples for classification. We first split the raw audio signal into short clips of length $t_w$. We then experimentally determine the shortest clip length that achieves the right trade-off between the classification performance and the computation time.

We compute the spectrogram based on the Short-Time Fourier Transform (STFT) of each clip in the sequence by breaking each audio clip into $M$ frames with 50% overlap between the frames in order to avoid boundary artifacts. Each frame is then Fourier transformed and successive frames are then concatenated into a matrix to form the spectrogram. Let $x[n]$ be the recorded raw audio signal with duration of $N_f$ samples, $f_s$ be the sampling frequency, $S(i, j)$ be the spectrogram representation of the 1D audio signal and $f(k) = kf_s/N_f$. We apply STFT on the length $M$ windowed frame of the signal as

$$X(i, j) = \sum_{p=0}^{N_f-1} x[n]\ w[n-j] \exp\left(-p\frac{2\pi k}{N_f}n\right),\ p = 0, \ldots, N_f - 1. \tag{3.1}$$

We apply a Hamming window function $w[n]$ to compensate for the Gibbs effect while computing STFT by smoothing the discontinuities at the beginning and end of the audio signal. The Hamming window function can be expressed as

$$w[n] = 0.54 - 0.46 \cos\left(2\pi\frac{n}{M-1}\right),\ n = 0, \ldots, M - 1. \tag{3.2}$$

We then compute the log of the power spectrum as

$$S_{\log}(i,j) = 20\log_{10}(|X(i,j)|). \tag{3.3}$$

We chose $N_f$ as 2048 samples, therefore the spectrogram contains 1024 Fourier coefficients. By analyzing the spectrum, we found that most of the spectral energy is concentrated below the lower 512 coefficients. Hence we only use the lower 512 coefficients to reduce computational complexity and to accelerate the inference time. The noise and the sound intensity levels vary substantially across the entire dataset as we recorded the data in different environments. Several factors contribute to this imbalance including ambient environmental noise at different times of the day, wind conditions and servo noise at different speeds. Therefore, we normalize the spectrograms with the maximum amplitude. Normalization also makes our approach invariant to the capture level of the microphone, distance of the microphone from the vehicle-terrain interface and the type of microphone being used. We compute the normalized spectrogram as

$$S(i,j) = \frac{S_{\log}(i,j)}{\max_{i,j} S_{\log}(i,j)}. \tag{3.4}$$

In order to create additional training data and to enable the learned representations to be invariant to commonly observed signal perturbations, we apply several frequency domain signal augmentations. We use the two dimensional affine transform and warping, with random offsets in time and frequency to shift the spectrogram randomly. Furthermore, we create additional samples using time stretching, modulating the tempo, random equalization, varying the volume gain, using frequency and time normalization with a sliding window and local contrast normalization.

In the following section, we first describe our TerrainNet architecture that operates on spectrograms of independent audio frames of vehicle-terrain interaction sounds, followed by the recurrent TerrainNet++ architecture which incorporates LSTM units to model the sequential nature of the signal and more effectively exploit the temporal dynamics. By incorporating LSTMs, we enable our model to capture how the frames change during a time period and thereby improve the robustness in the real-world.

### 3.2.2 TerrainNet Architecture

The spectrograms in our training set are of the form $S = \{s_1, \ldots, s_M\}$ with $s_i \in \mathbb{R}^N$. Each of them are of size $v \times w$ and number of channels $d$ ($d = 1$ in our case). We assume $M$ to be the number of samples and $y_i$ as the class label in one-hot encoding $y_i \in \mathbb{R}^C$, where C is the number of classes. Our architecture shown in Figure 3.4 consists of twelve convolutional layers, two inner-product layers, eight pooling layers and a softmax layer. All the convolutional and pooling layers are one dimensional and the convolutions convolve along the temporal dimension, making the resulting learned features translation-invariant

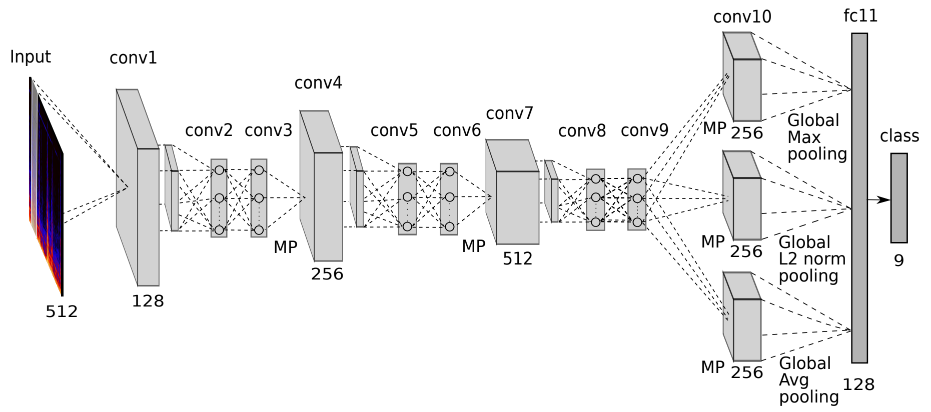


**Figure 3.4:** Depiction of our proposed TerrainNet architecture. The term MP refers to max-pooling. We first transform the raw audio signal of vehicle-terrain interactions into its spectrogram representation which is taken as input by the DCNN for feature learning and classification.

only in time domain. Unlike images, the two axes of the spectrogram have different units and represent different quantities, time vs. frequency, therefore we do not use square filters in the convolutional layers as typically employed in DCNNs using image data. The layers *conv1*, *conv4* and *conv7* have a kernel size of three with a fixed stride of one. We then ass two $1 \times 1$ convolutional layers with a stride of one after each of the aforementioned *conv1*, *conv4* and *conv7* layers in order to enhance the discriminability for local patches within the receptive fields. Each of the $1 \times 1$ convolutions have the same number of filter channels as the preceding convolutional layer and the filters learned by them are a better non-linear function approximator. The use of $1 \times 1$ convolutions for better abstraction was introduced in the influential Network in Network architecture [69]. We add a max-pooling layer with a kernel of two after *conv3*, *conv6* and *conv9* to downsample the intermediate representations in time. Max-pooling achieves partial invariance by only taking the high activations from adjacent hidden units that share the same weight, thereby providing invariance to small phase shifts in the signal.

DCNNs that are designed to operate on images for various perception tasks, specifically preserve the spatial information of features learned in order to be able to localize the objects of interest in the image. However, for the task of estimating the type of the terrain from spectrograms of the audio signal, the network only needs to be able to identify the presence or absence of terrain-specific spectrogram features, rather than having the ability to localize these features. Therefore, we introduce a new Global Statistical Pooling (GSP) scheme that applies different pooling mechanisms across the entire input tensor and combines them to aggregate statistics of the features learned across a particular dimension. Our architecture employs the proposed GSP in our network by incorporating three different global pooling layers after *conv9* to compute the statistics of the features learned across time.

We use an inner product layer *fc11* to combine the outputs of global max pooling, global $L^2$-norm pooling and global average pooling. Subsequently, we feed the resulting output to another inner-product layer to yield the number of classes in the dataset. We investigated various combinations of different pooling layers and found that for other configurations, the accuracy dropped by over 3%. We also investigated the effect of adding global stochastic pooling, however, the network did not show any significant improvement in performance. We use Rectified Linear Units (ReLUs) $f(x) = \max(0, x)$ after the convolutional layers and dropout regularization [92] on the inner-product layer *fc11*. ReLUs have significantly aided in overcoming the vanishing gradient problem and they have been shown to considerably accelerate training compared to $\mathsf{tanh}$ units. We also investigated the effect of employing Parameterized Rectified Linear Units (PReLU) [93] that improve model fitting. However, it drastically affected the performance of our network.

We use the multinomial logistic loss with $\mathsf{softmax}$ to train our network. Let $f_j(s_i, \theta)$ be the activation value for spectrogram $s_i$ and class $j$, $C$ is the total number of terrain classes, $\theta$ be the parameters of the network with weights $\mathcal{W}$ and biases $b$. The $\mathsf{softmax}$ function and the loss is computed as

$$P(y = j \mid s_i, \theta) = \mathsf{softmax}(f(s_i, \theta)) = \frac{\exp(f_j(s_i, \theta))}{\sum\limits_{k=1}^{C} \exp(f_k(s_i, \theta))}, \tag{3.5}$$

where $P(y = j \mid s_i, \theta)$ is the probability of the $j^{th}$ terrain category. The loss function can then be computed as

$$\mathcal{L}(u, y) = -\sum_k y_k \log u_k. \tag{3.6}$$

Using Stochastic Gradient Decent (SGD), we then solve

$$\min_\theta \sum_{i=1}^{N} \mathcal{L}(\mathsf{softmax}(f(s_i, \theta)), y_i). \tag{3.7}$$

We detail the training protocol that we employ and our choice of hyperparameters in Section 3.4.1.

### 3.2.3 TerrainNet++ Architecture

In the previous section, we presented an architecture that operates on individual spectrograms of fixed length, representing independent audio frames. However, as an audio signal is a function that measures the air pressure variation over time, modeling this sequential nature of the signal can improve the classification performance and the robustness in the real-world. LSTM-based Recurrent Neural Networks (RNNs) have recently achieved state-of-the-art results for sequential learning tasks such as language translation [94], speech recognition [26] and image captioning [95]. The standard RNN learns the temporal dynamics from a sequence by mapping inputs $x = (x_1, \ldots, X_T)$, to a sequence of hidden

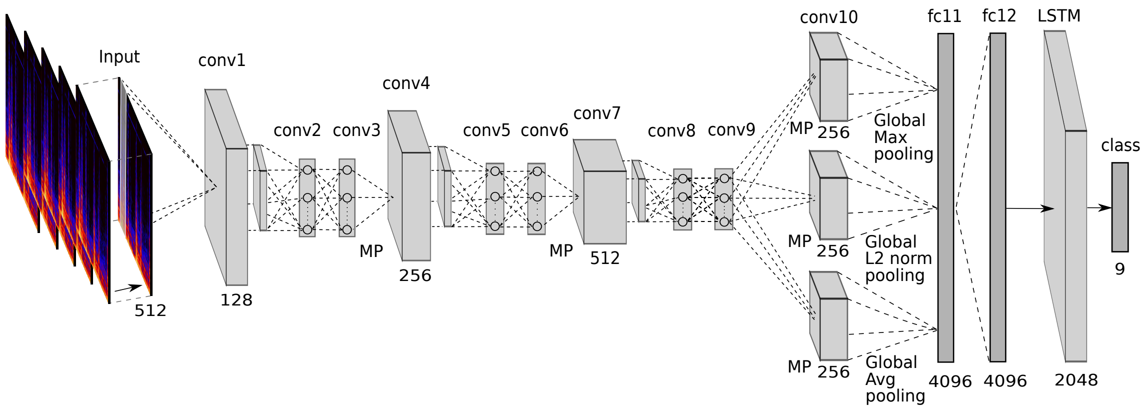


**Figure 3.5:** Depiction of our proposed recurrent TerrainNet++ architecture. The input to the network is a window of spectrograms representing the vehicle-terrain interaction sounds. The LSTM is unrolled in time and MP refers to max pooling.

states $h = (h_1, \ldots, h_T)$, and the hidden states to an output sequence $y = (y_1, \ldots, y_T)$. This is achieved by iterating through the following equations from $t = 1$ to $T$, as

$$h_t = g(\mathcal{W}_{ih}x_t + \mathcal{W}_{hh}h_{t-1} + b_h), \tag{3.8}$$

$$y_t = \mathcal{W}_{ho}h_t + b_o, \tag{3.9}$$

where $g$ is is the hidden layer activation function such as an element-wise application of sigmoid non-linearity, $\mathcal{W}$ and $b$ terms denote the weight matrices and bias vectors, with subscripts $i, h$, and $o$ denoting the input layer, the hidden layer and the output layer respectively. For example, $\mathcal{W}_{ih}$ is the weight matrix connecting the input layer and the hidden layer, while $\mathcal{W}_{hh}$ is the weight matrix connecting the different hidden layers. Similarly, $b_h$ and $b_o$ are the bias vectors for the hidden and output layers respectively. As discussed extensively in previous works [26, 95, 96], traditional RNNs often suffer from vanishing and exploding gradient problems that occur from propagating gradients through several layers of the RNN. The larger the length of temporal input, the harder it is to train the RNN. The Long Short-Term Memory (LSTM) [96] was proposed as a solution to this problem and to enable exploitation of long-term temporal dynamics from a sequence. LSTMs incorporate memory units containing several gates that regulate the flow of information in and out of the cells. We incorporate LSTMs in our proposed TerrainNet++ architecture which builds upon the TerrainNet architecture described in Section 3.2.2.

The input to TerrainNet++ is a spectrogram sequence of a certain time window length $T$. The TerrainNet++ architecture shown in Figure 3.5 has a structure similar to TerrainNet until the global statistical pooling (GSP) layer which combines three different pooling strategies. Two inner-product layers *fc11* and *fc12* with 4096 filters are then added after the GSP module. Similar to TerrainNet, we add dropout on the inner-product layers to regularize the model. The resulting sequence of features from *fc12* is then fed into an LSTM layer which exploits the temporal dependencies, followed by a time-distributed

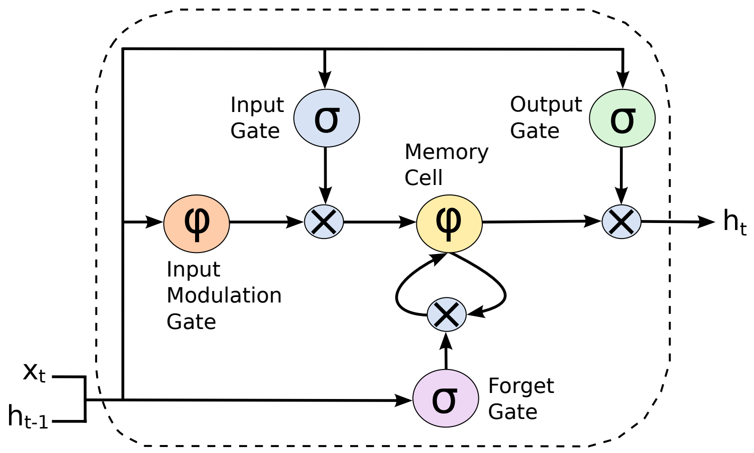


**Figure 3.6:** Depiction of the Long Short-Term Memory (LSTM) unit.

inner-product layer and a softmax layer. Figure 3.6 illustrates the topology of the LSTM unit, where $x_t$ is the input at time $t$. The activations can then be formulated as

$$i_t = \sigma(\mathcal{W}_{xi}x_t + \mathcal{W}_{hi}h_{t-1} + b_i), \tag{3.10}$$

$$f_t = \sigma(\mathcal{W}_{xf}x_t + \mathcal{W}_{hf}h_{t-1} + b_f), \tag{3.11}$$

$$o_t = \sigma(\mathcal{W}_{xo}x_t + \mathcal{W}_{ho}h_{t-1} + b_o), \tag{3.12}$$

$$g_t = \phi(\mathcal{W}_{xc}x_t + \mathcal{W}_{hc}h_{t-1} + b_c), \tag{3.13}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t, \tag{3.14}$$

$$h_t = o_t \odot \phi(c_t), \tag{3.15}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the sigmoid nonlinearity and $\phi(x) = (e^x - e^{-x}) \cdot (e^x + e^{-x})^{-1} = 2\sigma(2x)$ is the hyperbolic tangent nonlinearity. The symbol $h_t \in \mathbb{R}^N$ is the hidden unit, $g_t \in \mathbb{R}^N$ is the input modulation gate, $c_t \in \mathbb{R}^N$ is the memory cell, $i_t \in \mathbb{R}^N$ is the input gate, $f_t \in \mathbb{R}^N$ is the forget gate, and $o_t \in \mathbb{R}^N$ is the output gate. The symbols $\mathcal{W}$ and $b$ are the weight matrices and bias vector with subscripts $i, f, h, c$, and $o$ representing the input gate, forget gate, hidden gate, cell state and output gate respectively. The symbol $\odot$ represents element-wise product operator. The hidden state $h_t$ models the terrain that the robot is traversing on at time $t$. The output of the memory cell changes over time based on the past states and the current state of the cell. Therefore, the hidden state is formed based on the short-term memory of the past clip. At timestep $t$, the predicted distribution $P(y_t)$ can be computed by taking the softmax over the outputs of the sequence from the LSTM units $z_t$, i.e.,

$$P(y_t = c) = \frac{\exp(\mathcal{W}_{zc}z_{t,c} + b_c)}{\sum\limits_{c' \in C} \exp(\mathcal{W}_{zc}z_{t,c'} + b_c)}. \tag{3.16}$$

We train the entire model end-to-end using minibatch Stochastic Gradient Decent (SGD), similar to TerrainNet. As the TerrainNet++ architecture utilizes LSTM units to model the temporal relationships in addition to the proposed GSP module, we evaluate its utility in the architectural analysis presented in Section 3.4.3.1.

### 3.2.4 Noise-Aware Training

Noise robustness is a critical factor that substantially influences the performance of the model in real-world environments. The performance especially degrades when the vehicle-terrain interaction signal has a significantly lower Signal-to-Noise Ratio (SNR) making the model more susceptible to additive noise and reverberations. Conventional techniques employed in the speech recognition domain are loosely based on known mechanisms of auditory frequency encoding [97], therefore they are not directly employable while classifying vehicle-terrain interaction sounds. Moreover, as vehicle-terrain interaction sounds have very similar spectral properties as noise sources modeled for speech recognition tasks, existing front-end signal processing techniques are not applicable.

As a solution to this problem, we propose a noise-aware training scheme that substantially improves the robustness and the generalizability of our model without any additional preprocessing or computational burden. The basic principle of our noise-aware training scheme is to randomly inject common ambient environmental noises into the training data so that the model learns to distinguish between noise corruption patterns and the relevant vehicle-terrain interaction features. As our model learns hierarchical representations, the lower layers learn both pure vehicle-terrain interaction features as well as ambient noise features, whereas the higher layers learn compositional representations by distinguishing between them. Furthermore, the temporal component of our network enables the model to learn the evolution of both signals. Due to the temporal depth, the network is able learn heterogeneous signal patterns and at what stage to de-emphasize the noise while making decisions. Noise-conditioned decision boundaries are learned as we train the network with signals corrupted with noise at different SNRs. Additionally, the perturbation introduced by the injected noise regularizes the network and improves the generalization ability of the model. It helps improve the classification of pure signals as the easily degraded parts of the signal are blurred by noise which forces the network to learn the more dominant features, hence avoiding over-fitting. In summary, the goal of our noise-aware training is to enable the model to learn the common discriminative noise patterns and better regularize the network to improve the generalization performance to real-world environments with different ambient noises.

We randomly select a noise following a multinomial distribution $\mathrm{Mult}(\mu_1, \mu_2, \ldots, \mu_n)$ while training, where $n$ is the types of noises and $\mu_i$ is sampled from a Dirichlet distribution as $(\mu_1, \mu_2, \ldots, \mu_n) \sim \mathrm{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_n)$, where $\alpha_i$ is set to control the base distribution of the noise types. The SNR of the noised sample follows a Gaussian distribution $\mathcal{N}(\mu_{\mathrm{SNR}}, \sigma_{\mathrm{SNR}})$, where $\mu_{\mathrm{SNR}}$ and $\sigma_{\mathrm{SNR}}$ are the mean and variance. Following a uniform distribution, we randomly select a start time $s$ on the noise signal and scale it according to the desired SNR before adding it with the pure vehicle-terrain interaction signal. We use the Praat framework [98] for all the signal processing computations and the ambient noise recordings from the Diverse Environments Multichannel Acoustic Noise Database (DEMAND). The recordings in the DEMAND database were captured using 16 omnidirectional electret condenser microphones in over 18 different environments. We categorize the ambient noise environments into seven classes for in-depth experiments as follows:

- White: White noise has a very wide band and it is one of the most common noise sources. It has a very similar effect to that of various physical and environmental disturbances including wind and water sources.
- Domestic: This category includes ambient noises from living rooms, kitchens and washing rooms. As our terrain categories also contain indoor terrains, it is crucial to train the model with common indoor ambient noises.
- Nature: The nature category contains outdoor noise samples from a sports field, a river creek with flowing water and a city park.
- Office: This category contains recordings from an office with people working on computers, a hallway where people pass by occasionally and from interior public spaces.
- Public: The public category contains recordings from interior public spaces such a bus station, a cafeteria with people and from a university restaurant.
- Street: The street category contains noise recordings from outdoor inner-city public roads and pedestrian traffic. It contains a busy traffic intersection, a town square with tourists and a cafe at a public space.
- Transportation: This category contains recordings of vehicle noises such as cars, subways and buses.

As vehicle-terrain interaction sounds are unstructured in nature, corrupting it with various heterogeneous noises might quash the most relevant discriminative features leading to poor performance in the real-world. Therefore, we first investigate the effect of these noise disturbances by evaluating the model trained on the pure uncorrupted training set with the noise corrupted test set. We then perform noise-aware training with different SNRs. The training procedure that we employ is further detailed in Section 3.4.1.

### 3.2.5 Baseline Feature Extraction

There is a wide range of audio features that have been proposed for various speech and sound recognition tasks. Recent work has explored the utilization of these features for audio-based terrain classification [76, 77, 78]. We evaluate the performance of our proposed architectures against the following set of classical baseline audio features that include both time and frequency domain features:

- Ginna features [99]
- Spectral features [76]
- Ginna & Shape features [99, 100]
- MFCC & Chroma features [101]
- Trimbral features [102]
- Cepstral features [103]

Recently, Ginna and Shape features have demonstrated the best results for terrain classification using wheeled mobile robots [76]. Ginna features [99] is a six dimensional feature vector consisting of three features extracted from the time domain and three features extracted from the frequency domain. The time domain features are Zero Crossing Rate (ZCR), Short Time Energy (STE) and entropy (U), while the frequency domain features include Spectral Centroid (SC), Spectral Flux (SF) and Spectral Rolloff (SR). Zero Crossing Rate (ZCR) can be defined as the rate at which the signal changes from positive to negative and it is a measure of the noisiness of the signal. ZCR is one of the most traditional audio feature used for speech and music classification. STE is the energy in a short segment of the signal which can be computed as the square of the amplitude of the signal. Entropy characterizes the abrupt energy changes in the signal. ZCR, STE and entropy can be computed as

$$\mathrm{ZCR} = \frac{1}{2N}\sum_{m=0}^{N-1} |\,\mathrm{sgn}[y(m)] - \mathrm{sgn}[y(i-1)]|, \tag{3.17}$$

$$\text{where, } \mathrm{sgn}[x(m)] = \begin{cases} 1, & x(n) \geq 0 \\ -1, & x(n) < 0 \end{cases},$$

$$\mathrm{STE} = \sum_{m=0}^{N-1} X((m).w(n-m))^2, \tag{3.18}$$

$$\mathrm{U} = -\sum_{i=0}^{N-1} \sigma^2 \cdot \log_2(\sigma^2), \tag{3.19}$$

where $X(n)$ is a sample of the power spectrum at bin $n$, $f(n)$ is the center frequency of the bin $n$, $w(n)$ is the windowing function, $N$ is the size of the window and $\sigma$ denotes the normalized energy.

The most commonly used frequency domain feature is the Spectral Centroid (SC), which is also referred to as the median of the power spectrum. It indicates where the center of mass of the spectrum lies and it provides a measure of the spectral shape. Spectral Flux (SF) indicates the change in the power spectrum between successive windows and it is a measure of the amount of local spectral change. Spectral Rolloff (SR) is the frequency below the 95th percentile of the power in the spectrum and it is a measure of the skewness of the spectral shape. SC, SF and SR can be computed as

$$\mathrm{SC} = \frac{\sum\limits_{m=0}^{N-1} f(m)X(m)}{\sum\limits_{m=0}^{N-1} X(m)}, \tag{3.20}$$

$$\mathrm{SF} = \frac{\sqrt{\sum\limits_{m=1}^{N-1} (X(m,n) - X(m,n-1))^2}}{N-1}, \tag{3.21}$$

$$\mathrm{SR} = K, \tag{3.22}$$

$$\text{where} \sum_{m=0}^{K} X(m) = 0.95 \sum_{m=0}^{\mathrm{fmax}} X(m),$$

where fmax is the maximum frequency at bin *m*.

Shape features [100] is a four dimensional feature vector characterizing the shape of the distribution. It consists of spectral centroid, spectral standard deviations, spectral skewness and kurtosis. Spectral Skewness (SS) measures the symmetry of the distribution of the spectral magnitude around the mean. Spectral Kurtosis (SK) measures the similarity of the distribution of the spectral magnitude to that of a Gaussian distribution. SS and SK can be computed as

$$\mathrm{SS} = \frac{1}{S}\sum_{m=0}^{N-1}\left(\frac{X(m)-\mu}{\sigma}\right)^3, \tag{3.23}$$

$$\mathrm{SK} = \frac{1}{S}\sum_{m=0}^{N-1}\left(\frac{X(m)-\mu}{\sigma}\right)^4 - 3, \tag{3.24}$$

$$\text{where } \mu = \frac{1}{S}\sum_{m=0}^{N-1} X(m),$$

$$\sigma = \sqrt{\frac{1}{S}\sum_{m=0}^{N-1}(X(m)-\mu)^2},$$

where S is half the size of the window $N$, $\mu$ is the mean across the windows and $\sigma$ is the standard deviation across the windows.

A combination of Mel-frequency Cepstral Coefficients (MFCCs) [104] and Chroma features [105] have demonstrated state-of-the-art performance for music classification tasks [101]. MFCCs are one of the most widely used features for audio classification and Chroma features are strongly related to the harmonic progression of the audio signal. We use a combination of 12-bin MFCCs and 12-bin Chroma features for our comparisons.

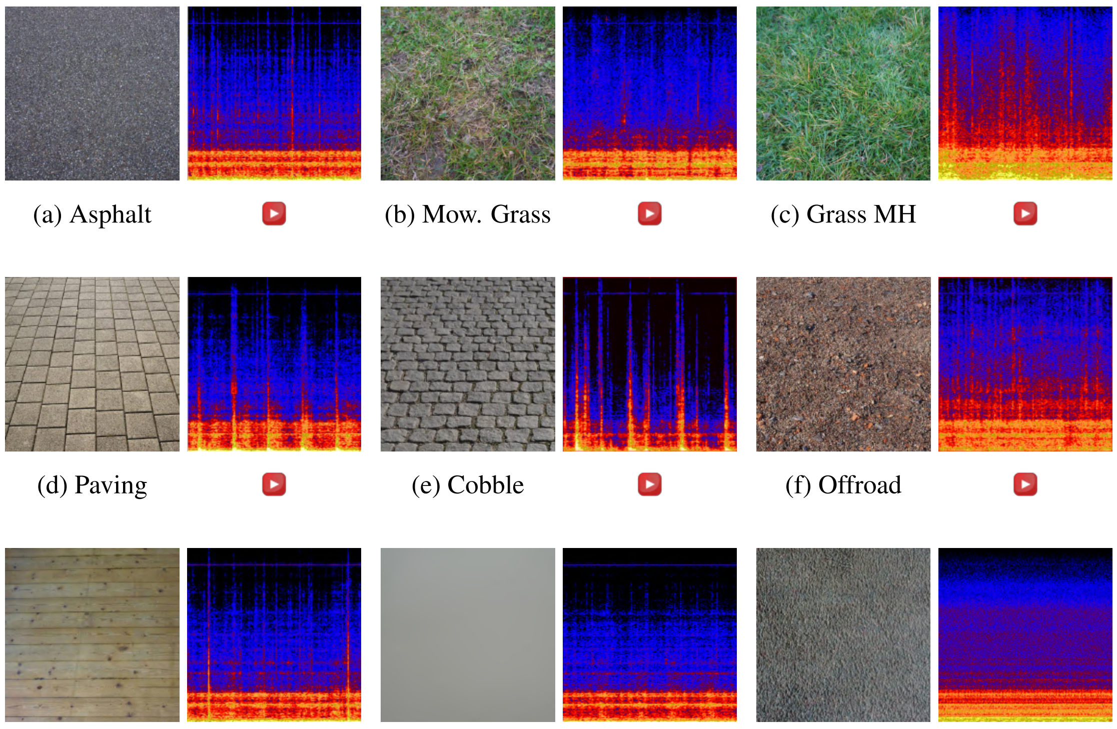
Timbral features have shown impressive results for music genre classification [102]. They contain means and variances of spectral centroid, spectral rolloff, spectral flux, zero crossing rate, low energy, and first 5 MFCCs. We also use this 19 dimensional feature vector for our baseline comparison. For our final baseline, we use a combination of cepstral features containing 13-bin MFCCs, Line Spectral Pair (LSP) and Linear Prediction Cepstral Coefficients (LPCCs) [103].

## 3.3 Data Collection and Labelling

To the best of our knowledge, there is no publicly available audio-based terrain classification dataset till date. Therefore, in order to facilitate this work, we collected an extensive dataset of vehicle-terrain interactions that we made publicly available. We used the Pioneer P3-DX platform which has a small footprint and feeble motor noise. We equipped the P3-DX with rugged wheels that enabled us to collect data both indoors as well as in unstructured outdoors environments. Interference from nearby sound sources in the environment can drastically influence the classification. It can even augment the vehicle-terrain interaction data by adding its own attributes from each environment. In order to prevent such biases in the training data, we use a shotgun microphone that has a supercardioid polar pattern which helps in rejecting off-axis ambient sounds. We chose the Rode VideoMic Pro and mounted it near the right wheel of the robot as shown in Figure 3.2. The microphone has an integrated shock mount that prevents the pickup of any unwanted vibrations caused during the traversal. Note that our approach presented in this chapter is platform independent, as the network learns features from the given training data.

Thus far, audio-based terrain classification using mobile robots has only been focused on outdoor terrains that have a large variation in geometric properties and therefore produce very distinct audio signatures. Recently, audio-based terrain classification has been demonstrated for both indoor and outdoor terrains using legged robots. However, legged robots have significantly lesser drive motor noise and they produce very isolated sounds from their leg-terrain interaction which are substantially easier to detect and classify. In this chapter, our objective is to classify a wide variety of both indoor and outdoor terrains using vehicle-terrain interaction signals. Therefore, we collected over 15 h of audio data from a total of 9 different indoor and outdoor terrains. We particularly choose our terrain categories such that some of them have similar visual features (Figures 3.7 (a), (h), and (i)) and hence pose a big challenge for classification using vision-based approaches.

We collected the data in several different locations to have enough variability in the terrains and to enable our model to generalize effectively to a wide range of environments. Therefore, even the audio signals in the same terrain category have varying spectral and temporal characteristics. Furthermore, we varied the speed of the robot from $0.1\,\mathrm{m\,s^{-1}}$ to



(a) Asphalt (b) Mow. Grass (c) Grass MH

(d) Paving (e) Cobble (f) Offroad

(g) Wood (h) Linoleum (i) Carpet

**Figure 3.7:** Examples of terrain categories from our dataset showing the visual image of the terrain with the corresponding spectrogram for a 2 s clip. Grass MH refers to *Grass Medium-High*.

1.0 m s$^{-1}$ during the data collection runs. We recorded the data in the lossless 16-bit WAV format at 44.1 kHz to avoid any recording artifacts. There was no software level boost added during the recordings as we identified that it amplifies the ambient noise substantially, instead we used a 20 dB hardware level boost. We manually labeled the data using the live tags and timestamps made during the recordings. A waveform analyzer tool was then used to crop out significant noise disturbances that had substantially higher amplitudes than the vehicle-terrain interaction signals. By noisy disturbances, we refer to uncommon temporary environmental disturbances such as a nearby car or train passing. Finally, we split the dataset into train and test sets, ensuring that each terrain class approximately has the same number of samples to prevent any bias towards a specific class. We also ensured that the training and testing sets do not contain clips from the same location. We made the dataset publicly available at `http://deepterrain.cs.uni-freiburg.de`.

## 3.4 Experimental Evaluation

In this section, we first describe the training protocol that we employ in Section 3.4.1 and extensive results using our proposed TerrainNet and TerrainNet++ architectures in comparison to approaches that employ state-of-the-art audio features in Section 3.4.2. We also present comprehensive ablation studies on the various architectural configurations and parameter choices in Section 3.4.3, followed by detailed class-wise analysis in Section 3.4.4 and performance analysis in the presence of substantial amount of ambient noise in Section 3.4.5. In Section 3.4.6, we then present experiments that demonstrate the efficacy of our noise-aware training scheme in improving the robustness of our models in adverse ambient noise conditions. Finally, in Section 3.4.7, we present experiments using a low-cost mobile phone microphone that demonstrates the hardware independence of our approach. Note that we do not retrain the network on data using the inexpensive microphone, we only test our previously trained model.

We use the Caffe [106] deep learning library for all the experiments presented in this chapter and the LSTM described in [107]. We employ a more efficient LSTM implementation that is faster than the default Caffe LSTM [95] by computing gradient with respect to recurrent weights with a single matrix computation. All our models were trained end-to-end and the experiments were run on a system with an Intel Xeon E5, 2.4GHz and an NVIDIA TITAN X GPU with cuDNN acceleration. We primarily report the classification accuracy metric to quantify the performance of our models but we also report the precision, recall, confusion matrices, classification error and the inference time for in the detailed comparisons. The results from our experiments are described in the following sections.

### 3.4.1 Training Protocol

As the input to our network is the spectrogram of the audio clip, we performed experiments training our network with spectrograms of audio clips of varying lengths from 200 ms to 2000 ms and identified the shortest clip length that yields a good trade-off between performance and computation time. Results from this experiment is shown in the ablation study presented in Section 3.4.3.2. We initialize the convolution and inner-product layers with Xavier weight initialization [108]. The Xavier weight filler initializes weights by drawing from a zero mean uniform distribution from $[-a, a]$ and the variance as a function of the number of input neurons, where $a = \sqrt{3 / n_{in}}$ and $n_{in}$ is the number of input neurons. This enables us to move away from the traditional layer by layer generative pre-training. Furthermore, we use a dropout probability of 0.5 on the inner-product layers in order to regularize the network.

We found the network to be extremely sensitive to the hyperparameters employed, especially the hyperparameters in the recurrent TerrainNet++ architecture. Therefore, we use the Spearmint Bayesian optimization library [109] to tune the hyperparameters. We first optimized the learning policy and the initial learning rate over fixed, inverse, step and poly learning rate policies. We obtained the best performance using an initial learning rate of $\lambda_0 = 0.01$ and the poly learning rate policy which can be defined as $\lambda_n = \lambda_0 \times (1 - N/N_{\max})^c$,

**Table 3.1:** Comparison of the classification accuracy of TerrainNet and TerrainNet++ against several state-of-the-art audio features extracted from our dataset. Results are reported for an audio clip of length 300 ms in terms of the mean across all the classes and its standard deviation.

| Features | SVM Linear (%) | SVM RBF (%) | k-NN (%) |
|---|---|---|---|
| Ginna [99] | $44.87 \pm 0.70$ | $37.51 \pm 0.74$ | $57.26 \pm 0.60$ |
| Spectral [76] | $84.48 \pm 0.36$ | $78.65 \pm 0.45$ | $76.02 \pm 0.43$ |
| Ginna & Shape [99, 100] | $85.50 \pm 0.34$ | $80.37 \pm 0.55$ | $78.17 \pm 0.37$ |
| MFCC & Chroma [101] | $88.95 \pm 0.21$ | $88.55 \pm 0.20$ | $88.43 \pm 0.15$ |
| Trimbral [102] | $89.07 \pm 0.12$ | $86.74 \pm 0.25$ | $84.82 \pm 0.54$ |
| Cepstral [103] | $89.93 \pm 0.21$ | $78.93 \pm 0.62$ | $88.63 \pm 0.06$ |
| TerrainNet (Ours) | | $\mathbf{97.52 \pm 0.016}$ | |
| TerrainNet++ (Ours) | | $\mathbf{98.76 \pm 0.009}$ | |

where $N$ is the iteration number, $N_{\max}$ is the maximum number of iterations and $c$ is power. We further tuned the number of outputs in the inner-product layers and the LSTM layer as they were both highly correlated and they had a big impact on the performance of the model. This parameter tuning is further discussed in the ablation study presented in Section 3.4.3. We train the entire model end-to-end using minibatch Stochastic Gradient Decent (SGD) with a batch size of 256. We optimize SGD by smoothing the gradient computation for minibatchs using a momentum coefficient $\alpha$ as $0 < \alpha < 1$. We train the models for a maximum of 350,000 iterations ($\sim$ 135 epochs), which took about 4 days on a system with a single NVIDIA TITAN X GPU.

## 3.4.2 Comparison with the State-of-the-Art

In this section, we empirically evaluate our proposed TerrainNet architecture and our recurrent TerrainNet++ architecture in comparison to several state-of-the-art audio features described in Section 3.2.5. For all the state-of-the-art comparison experiments, we use a fixed audio clip length of 300 ms. As a preprocessing step for the baseline features, we normalize the data to have zero mean. We compare against Support Vector Machines (SVM) and k-Nearest Neighbor (kNN) classifiers, which have previously achieved the best performance for audio-based terrain classification [76]. SVMs perform well in high dimensional spaces and kNNs perform well when there are irregular decision boundaries. We use the one-vs-all voting scheme with SVMs to handle multiple classes and experiment with both Linear and Radial Basis Function (RBF) as decision functions. We use the inverse distance weighting for kNNs and we optimize the hyperparameters for both classifiers by performing a grid-search using leave-one-out-cross-validation. We use scikit-learn [110] and LibSVM [111] for the implementation of the baseline classifiers.

Table 3.1 shows the results from this comparison. The best performing baseline feature combination was Cepstral features using a linear SVM kernel which achieves an accuracy of 89.93%. While, Trimbral features using a linear SVM kernel achieves a comparable performance. Libby *et al.* [76] used Ginna and Shape features using an SVM RBF kernel which was the previous state-of-the-art for audio-based terrain classification. This approach achieves an accuracy of 80.37% on our dataset, which is 9.56% lesser than the best performing baseline. The lowest performance was obtained using only Ginna features with an SVM RBF kernel. Moreover, it can be observed that the feature sets containing MFCCs (MFCC & Chroma, Trimbral, and Cepstral) outperform other feature combinations by a substantial margin, thereby validating their utility for audio classification tasks.

Our proposed TerrainNet architecture achieves an accuracy of 97.52%, which is an improvement of 7.59% over the best performing Cepstral features and 12.02% over the previous state-of-the-art which uses Ginna and Shape features. Our recurrent TerrainNet++ architecture achieves an improved accuracy of 98.76% with a temporal window length of three, thereby setting the new state-of-the-art on this dataset. This demonstrates that the recurrent TerrainNet++ architecture is able to exploit the complex temporal relationships in the vehicle-terrain interaction signal and further improve the performance. Moreover, our TerrainNet model performs inference in 9.15 ms and our recurrent TerrainNet++ model performs inference in 12.37 ms for a audio clip length of 300 ms, whereas, the previous state-of-the-art approaches presented in Table 3.1 have feature extraction and classification times in the order of a few seconds. Note that noise-aware training and parameter tuning discussed in the following sections further improves upon the classification performance of our models reported in this section.

Furthermore, in order to compare the per-class classification performance of TerrainNet and TerrainNet++, we show the comparison of error rates of both these models in Figure 3.8. It can be seen that the TerrainNet++ model substantially decreases the classification error of most terrains. *Paving* and *Offroad* classes have the largest decrease in error, followed by *Linoleum* and *Wood* which have the highest false positives in the predictions of TerrainNet as seen on the confusion matrix shown in Figure 3.11. This is due to the fact that these terrains have very similar spectral responses when we consider short clip lengths. However, as our TerrainNet++ architecture learns features over a window of frames, it is able to better exploit the temporal dynamics of the signal. The improvement in performance achieved by TerrainNet++ for these classes demonstrates that even this complex relationship between these terrains can be learned by our temporally recurrent model.

## 3.4.3 Ablation Study

In this section, we describe the various experiments that we performed to gain insight on the learning of temporal relationships from the vehicle-terrain interaction signal using different DCNN architecture configurations. We first describe each of the architectures

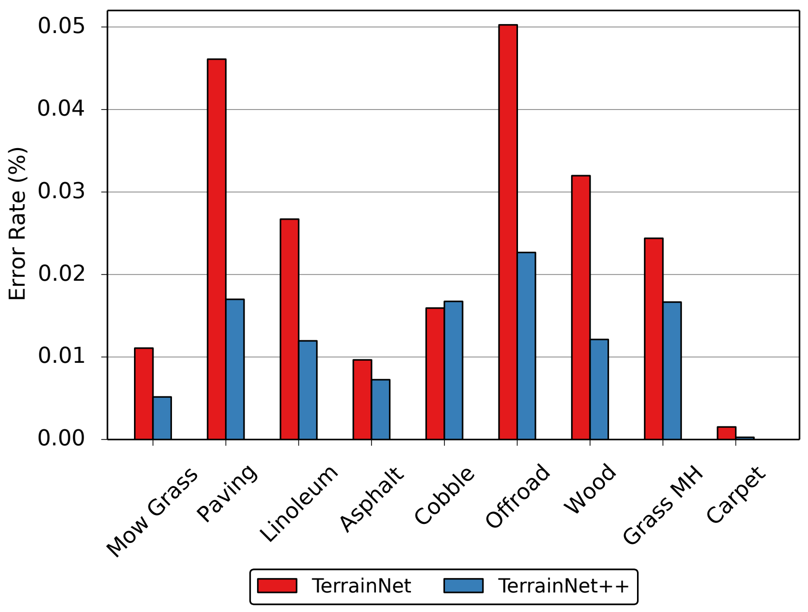


**Figure 3.8:** Comparison of the per-class classification error rate of TerrainNet and TerrainNet++ for an audio clip of length 300 ms. The recurrent TerrainNet++ model achieves a 1.24% decrease in the error rate compared to the TerrainNet model.

and then present an in-depth analysis on their performance. We also study the effect of different parameter configurations for our architecture and identify the critical parameters that yield the best performance.

#### 3.4.3.1 Architectural Analysis

We compare the performance of the following DCNN architectural configurations:

1. M1 (DCNN): Network topology similar to the TerrainNet architecture shown in Figure 3.4 with alternating convolution and max-pooling layers and without the GSP module. *Conv9* is followed by two inner-product layers and a softmax layer.

2. M2 (DCNN with GSP): Network topology similar to the M1 architecture, but it includes the GSP module after *Conv9*, which is followed by two inner-product layers and a softmax layer. This model is the TerrainNet architecture shown in Figure 3.4.

3. M3 (DCNN with LSTM): Network topology similar to the TerrainNet++ architecture shown in Figure 3.5 with alternating convolution and pooling layers and without the GSP module. *Conv9* is followed by two inner-product layers, an LSTM unit, another inner-product layer to yield the number of classes and a softmax layer.

4. M4 (DCNN with GSP & LSTM): Network topology similar to the M3 architecture, but it includes the GSP module after *Conv9*, which is followed by two inner-product

**Table 3.2:** Performance comparison of various architecture configurations using an audio clip length of 300 ms. Our temporally recurrent DCNN architecture with our proposed GSP and LSTM (M4), achieves a 7.47% improvement in accuracy over the architecture without any temporal learning component.

| Model | Acc. (%) | Prec. (%) | Rec. (%) |
|---|---|---|---|
| M1 (DCNN) | 91.29 | 91.88 | 91.56 |
| M2 (DCNN + GSP) | 97.52 | 97.56 | 97.61 |
| M3 (DCNN + LSTM) | 95.73 | 95.93 | 95.88 |
| M4 (DCNN + GSP & LSTM) | **98.76** | **98.75** | **98.82** |

layers, an LSTM unit, another inner-product layer to yield the number of classes and a softmax layer. This model is the TerrainNet++ architecture shown in Figure 3.5.

For the experiments described in this section, we use an audio clip of length 300 ms and for the LSTM models, we use a window size of three. Results from the comparison of the aforementioned architectural configurations is shown in Table 3.2. The M1 model which does not include any temporal representational learning component achieves an average accuracy 91.29%, which outperforms the previous state-of-the-art. This shows the ability of our proposed DCNN architecture to learn features that are more discriminative than classical audio features. The M2 model (TerrainNet) which includes our proposed GSP module that aggregates the statistics of temporal features, achieves an improvement of 6.23% over the M1 model demonstrating the utility of the GSP for temporal feature learning tasks. Interestingly, the M3 model which incorporates an LSTM unit and does not include the GSP module, achieves a lower accuracy than the M2 model. This can be attributed to the fact that the LSTM model does not benefit substantially from learning using a large audio clip length and a short temporal window. In the following section, we investigate the influence of the audio clip length and the temporal window length on the classification performance. Finally, our M4 model (TerrainNet++) which includes both our proposed GSP module that aggregates the statistics of pooled features across time and the LSTM unit that learns the temporal dynamics across several clips, outperforms all the other models by achieving an accuracy of 98.76%, which is a 7.47% improvement over the performance of the M1 model. Similar improvement can also be seen in the precision and recall values of the models. This illustrates that our proposed GSP module is critical for learning effective temporal relations from the vehicle-terrain interaction signals.

Figure 3.9 shows the per-class recall of the various TerrainNet architecture configurations discussed above. The DCNN model without the temporal feature learning components (M1) perform substantially worse than the other configurations for almost all the classes. While, the performance of the DCNN with GSP and LSTM model (M4) surpasses all

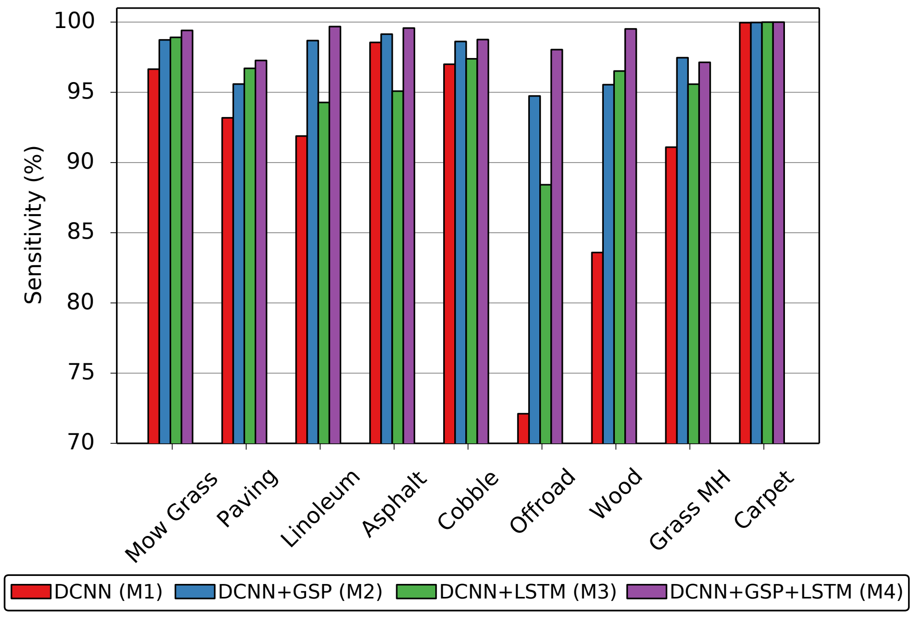


**Figure 3.9:** Comparison of the per-class recall of the various architecture configurations. Our temporally recurrent TerrainNet++ (M4) which includes the GSP and an LSTM unit substantially outperform the other model configurations in most of the terrain categories.

the other models for most terrain classes by a large margin demonstrating that learning temporal features is a necessary step for accurate audio-based terrain classification. All the models have near perfect recall for the *Carpet* class which is primarily due to the flatness of the spectral responses of the carpet terrain compared to the other terrain classes which can also be seen in the spectrograms of the terrains shown in Figure 3.7. The LSTM models (M3 and M4) perform better for flatter terrains such as *Wood, Carpet* and *Paving*, while the GSP (M2 and M4) models perform better when the terrains are more irregular. Therefore combining both the GSP and LSTM in the M4 model enables us to learn temporal relations that generalize to a wide range of coarse and flat terrains.

#### 3.4.3.2 Influence of Audio Clip length and LSTM window size

The audio clip length is one of the most critical parameters in our TerrainNet and TerrainNet++ architectures as it influences both the accuracy and the inference time. The biggest trade-off is with selecting an optimal clip length. As each clip is a new sample for classification, the shorter the clip length, the faster is the inference time and the higher is the rate at which we can infer the terrain. However, a larger clip length yields a better accuracy, but it correspondingly increases the inference time and decreases the rate at which we can infer the terrain, which is undesirable. For real-world robotic systems, fast

**Table 3.3:** Classification accuracy of our TerrainNet++ model for varying audio clip lengths and LSTM window lengths. Accuracy is shown is in percent.

| Clip Length | Window Length | | | | |
|---|---|---|---|---|---|
| | 2 | 3 | 4 | 5 | 6 |
| 300 ms | 97.43 | **98.76** | 80.09 | 98.70 | 98.62 |
| 250 ms | 96.74 | 97.38 | **98.93** | 79.96 | 98.67 |
| 200 ms | 96.69 | 96.22 | 97.86 | **99.03** | 79.58 |

**Table 3.4:** Performance comparison of the TerrainNet model and the TerrainNet++ model at varying audio clip lengths. We use an LSTM window length of three in the TerrainNet++ architecture. The corresponding inference time consumed is also shown.

| Clip Length | TerrainNet | | TerrainNet++ | |
|---|---|---|---|---|
| | Accuracy (%) | Time (ms) | Accuracy (%) | Time (ms) |
| 2000 ms | 99.86 | 45.40 | 99.88 | 23.27 |
| 1500 ms | 99.82 | 34.10 | 99.83 | 21.16 |
| 1000 ms | 99.76 | 21.40 | 99.78 | 16.43 |
| 500 ms | 99.41 | 13.30 | 99.45 | 13.75 |
| 300 ms | 97.36 | 9.15 | 98.76 | 12.37 |
| 250 ms | 94.05 | 9.15 | 98.93 | 12.36 |
| 200 ms | 91.30 | 9.14 | 99.03 | 12.23 |

classification rates are essential for making quick trafficability decisions. Moreover, the TerrainNet++ architecture also has an LSTM window length that needs to be optimized as it dictates the temporal length to be considered for classification. However, using an LSTM to learn temporal relationships allows us to use a shorter audio clip length. In this section, we investigate the relationship between these parameters by individually training models with various audio clip lengths and LSTM window lengths.

The results shown in Table 3.3 demonstrate that for a certain clip length, the classification accuracy of the model does not increase by just considering a larger LSTM window length. In fact, we found the models to be increasingly difficult to train for large LSTM window lengths. For a 300 ms audio clip, we obtained the best performance using an LSTM window length of three, above which the accuracy dropped significantly for a window length of four and then saturates for higher window lengths. We also experimented with smaller audio clip lengths of 250 ms and 200 ms, for which we observe the same pattern of obtaining the best performance for a certain LSTM window length, followed by a significant drop for larger window length and then the performance saturates for increasingly larger window lengths. Furthermore, we also trained models with lower audio clip lengths and with

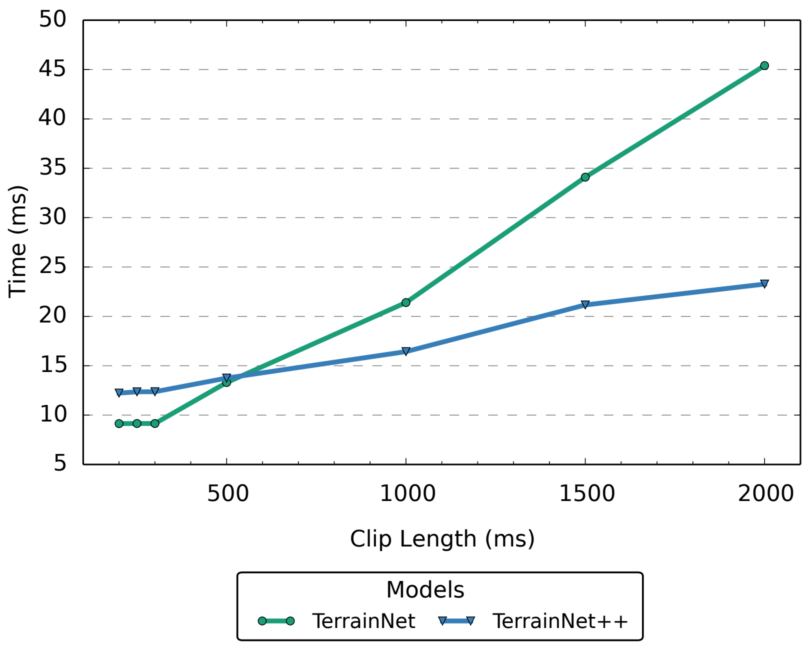


**Figure 3.10:** Inference time comparison of the TerrainNet model and the TerrainNet++ model for varying audio clip lengths. Our TerrainNet++ model has a comparatively faster inference time even for very long clip lengths.

higher LSTM window lengths, but they performed worse than the results reported in Table 3.3. The overall best performance among the models trained with various LSTM window lengths and audio clip lengths was obtained using a clip length of 200 ms and with a window size of five. This model achieves a classification accuracy of 99.03%, which is an improvement of 9.1% over the previous state-of-the-art models that employ classical audio features.

We further compared the performance of our TerrainNet model and our TerrainNet++ model for a wide range of audio clip lengths as shown in Table 3.4. It can be seen that our recurrent TerrainNet++ model outperforms our TerrainNet model for all audio clip lengths between 200 ms and 2000 ms. Furthermore, the TerrainNet++ model also has a faster inference time for most of the clip lengths. The speed-up is achieved by computing gradients with respect to recurrent weights in a single matrix computation. It can be seen that using audio clip lengths spanning longer than 500 ms yields an accuracy over 99%. However, it is impractical to use such a large clip length as the robot would not be able to continuously navigate at a reasonable speed while sensing the terrain. We observe that the best trade-off is achieved by the TerrainNet++ model with a clip length of 200 ms where it achieves a classification rate of over 81 Hz and an accuracy of 99.03%. This demonstrates the effectiveness of efficiently learning temporal relationships using our proposed GSP and LSTM units in our TerrainNet++ architecture.

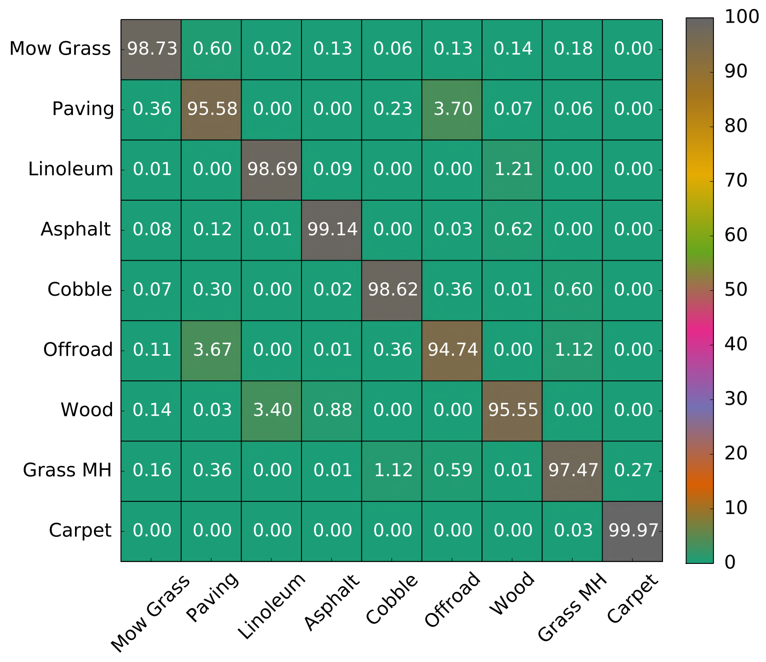
Figure 3.10 shows a plot of the classification rates of both the TerrainNet and TerrainNet++ models at varying audio clip lengths from 200 ms to 2000 ms. Although the TerrainNet model has a faster inference time for a clip length of 200 ms, it achieves an accuracy which is 7.73% lower than that of the TerrainNet++ model. Considering the trade-off between the audio clip length which influences the inference time and the classification accuracy, we choose the TerrainNet++ model which takes a 200 ms audio clip as input for the in-depth performance evaluation experiments described in the following sections.

### 3.4.4 Performance Evaluation

In order to further investigate the performance of our TerrainNet and TerrainNet++ architectures, we present the confusion matrix in Figure 3.11, which gives us insights on the misclassifications between the terrain classes. Figure 3.11 (a) shows the confusion matrix of the TerrainNet model in which we can see that the best performing classes were *Carpet* and *Asphalt*, while the most misclassified classes were *Offroad* and *Paving*. The *Offroad* and *Paving* classes have similar spectral responses when the audio clip length is smaller than 500 ms. *Wood* and *Linoleum* also have a fair amount of misclassifications as both these terrains are almost flat and have similar spectral responses when the robot is navigating at low speeds.

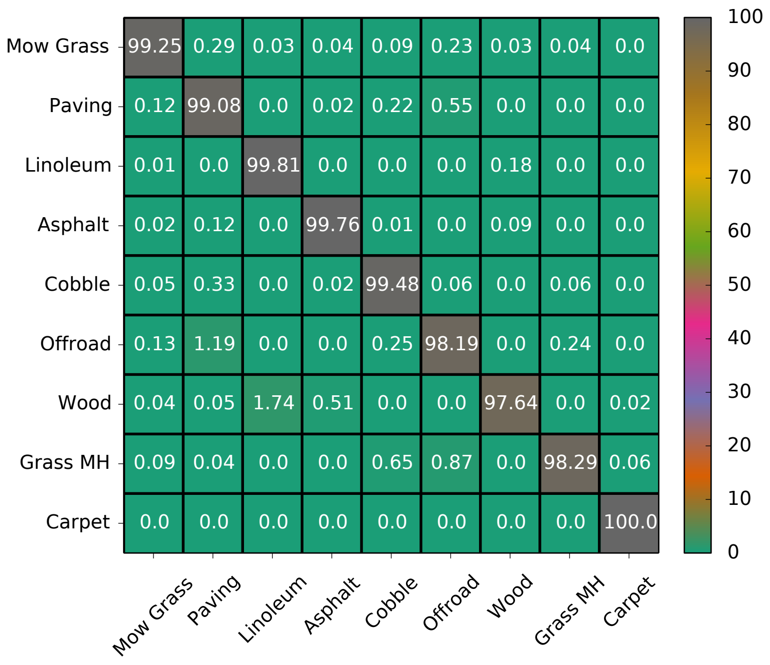
Figure 3.11 (b) shows the confusion matrix of our best performing recurrent TerrainNet++ model for a clip length of 200 ms and an LSTM window length of five. We can see that there is a substantial reduction in the number of false positives compared to our TerrainNet model. *Paving* and *Offroad* classes have a reduction of about 3.15% in the number of false predictions. While *Wood* and *Linoleum* have a reduction of 1.66% in the number of false predictions. Another notable reduction of 0.88% can be observed in the misclassification between *Offroad* and *Grass Medium-High.* The best performing classes were *Carpet* and *Linoleum.* In fact, the *Carpet* class demonstrated no false positives. Moreover, most of the classes demonstrate an accuracy of over 99%. The TerrainNet++ model outperforms TerrainNet by 1.51% in accuracy and with a shorter audio clip as input, thereby increasing the classification rate.

Furthermore, we also compare the per-class recall of the TerrainNet and TerrainNet++ models in Figure 3.12 as it gives us insights on the ratio of the correctly classified instances. For the TerrainNet model, the lowest recall was obtained for *Offroad, Wood* and *Paving* terrains, whereas the highest recall was obtained for the *Carpet* terrain. A substantial improvement in the recall of these terrains can be observed in the recall plot of the recurrent TerrainNet++ model shown in Figure 3.12 (b). The overall recall achieved by TerrainNet was 97.61%, while the TerrainNet++ model demonstrates an improvement of 1.44% and achieves an overall recall of 99.05%.



| | Mow Grass | Paving | Linoleum | Asphalt | Cobble | Offroad | Wood | Grass MH | Carpet |
|---|---|---|---|---|---|---|---|---|---|
| Mow Grass | 98.73 | 0.60 | 0.02 | 0.13 | 0.06 | 0.13 | 0.14 | 0.18 | 0.00 |
| Paving | 0.36 | 95.58 | 0.00 | 0.00 | 0.23 | 3.70 | 0.07 | 0.06 | 0.00 |
| Linoleum | 0.01 | 0.00 | 98.69 | 0.09 | 0.00 | 0.00 | 1.21 | 0.00 | 0.00 |
| Asphalt | 0.08 | 0.12 | 0.01 | 99.14 | 0.00 | 0.03 | 0.62 | 0.00 | 0.00 |
| Cobble | 0.07 | 0.30 | 0.00 | 0.02 | 98.62 | 0.36 | 0.01 | 0.60 | 0.00 |
| Offroad | 0.11 | 3.67 | 0.00 | 0.01 | 0.36 | 94.74 | 0.00 | 1.12 | 0.00 |
| Wood | 0.14 | 0.03 | 3.40 | 0.88 | 0.00 | 0.00 | 95.55 | 0.00 | 0.00 |
| Grass MH | 0.16 | 0.36 | 0.00 | 0.01 | 1.12 | 0.59 | 0.01 | 97.47 | 0.27 |
| Carpet | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.03 | 99.97 |

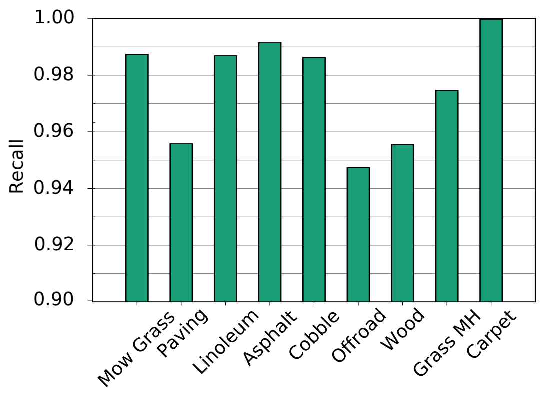
(a) Confusion matrix of our TerrainNet model for a clip length of 300 ms



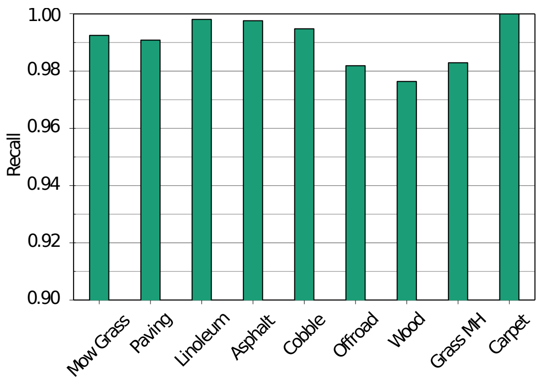
| | Mow Grass | Paving | Linoleum | Asphalt | Cobble | Offroad | Wood | Grass MH | Carpet |
|---|---|---|---|---|---|---|---|---|---|
| Mow Grass | 99.25 | 0.29 | 0.03 | 0.04 | 0.09 | 0.23 | 0.03 | 0.04 | 0.0 |
| Paving | 0.12 | 99.08 | 0.0 | 0.02 | 0.22 | 0.55 | 0.0 | 0.0 | 0.0 |
| Linoleum | 0.01 | 0.0 | 99.81 | 0.0 | 0.0 | 0.0 | 0.18 | 0.0 | 0.0 |
| Asphalt | 0.02 | 0.12 | 0.0 | 99.76 | 0.01 | 0.0 | 0.09 | 0.0 | 0.0 |
| Cobble | 0.05 | 0.33 | 0.0 | 0.02 | 99.48 | 0.06 | 0.0 | 0.06 | 0.0 |
| Offroad | 0.13 | 1.19 | 0.0 | 0.0 | 0.25 | 98.19 | 0.0 | 0.24 | 0.0 |
| Wood | 0.04 | 0.05 | 1.74 | 0.51 | 0.0 | 0.0 | 97.64 | 0.0 | 0.02 |
| Grass MH | 0.09 | 0.04 | 0.0 | 0.0 | 0.65 | 0.87 | 0.0 | 98.29 | 0.06 |
| Carpet | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 100.0 |

(b) Confusion matrix of our recurrent TerrainNet++ model for an audio clip length of 200 ms and a LSTM window length of five.

**Figure 3.11:** Comparison of the best performing TerrainNet model with our recurrent TerrainNet++ model. *Paving* and *Offroad* terrain classes have a substantially lesser number of false positives in the predictions of the TerrainNet++ model. Furthermore, the *Carpet* class demonstrates no false positives in the predictions of the TerrainNet++ model.



(a) Per-class recall of our TerrainNet model for an audio clip length of 300 ms



(b) Per-class recall of our TerrainNet++ model for an audio clip length of 200 ms and an LSTM window length of five.

**Figure 3.12:** Comparison of the per-class recall of our TerrainNet model with our recurrent TerrainNet++ model. The TerrainNet++ model achieves an improvement of 1.44% in the overall recall compared to the TerrainNet model.

### 3.4.5 Evaluation of Noise Tolerance

In this section, we investigate the noise tolerance of our TerrainNet++ model by adding ambient noise only to the test set and evaluating its performance on the model trained on clean noise-free signals. For the experiments described in this section, we use the vehicle-terrain interaction sounds corrupted with ambient noises from the DEMAND noise database which is categorized into seven classes and at varying SNRs as described in Section 3.2.4. We first only added noise to our test set using the technique described in [112] and evaluated the performance of the TerrainNet++ model trained on clean noise-free signals to obtain an estimate of the noise tolerance before performing the noise-aware training. Results from this experiment shown in Table 3.5 demonstrate that the performance drastically decreases with decreasing SNRs. Moreover, different ambient noises affect the performance of the model at varying degrees. Overall, ambient noises from the *Domestic, Street* and *White* categories are the most damaging.

Figure 3.13 shows the accuracy of our model for the various ambient noises at decreasing SNRs. It can be seen that for all the ambient noise categories the models are fairly robust for SNRs until 20 dB, thereafter the performance drops rapidly for noises from categories such as *Domestic, Street*, as well as white noise. This can be attributed to the broadband of these noises which corrupts vehicle-terrain interaction signals in most frequencies, while the other categories of ambient noises are present only at certain isolated frequencies. Hence, vehicle-terrain interaction signals consisting of high frequency components are relatively robust to ambient noises from the *Nature* category due to their presence mostly only in the low frequencies of the spectrum.

**Table 3.5:** Influence of various ambient noises on the classification accuracy of our TerrainNet++ model. Noise samples were extracted from the DEMAND noise database. Ambient noises in the *White, Domestic* and *Street* categories are the most damaging, while the noises in the *Nature* category are the least damaging. Results are shown in terms of the average accuracy of the model in percent.

| Noise | SNR (dB) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Clean | 30 | 20 | 10 | 0 | -10 | mean |
| White | 99.03 | 99.00 | 93.42 | 69.66 | 20.85 | 9.16 | 58.42 |
| Domestic | 99.03 | 98.63 | 82.84 | 55.24 | 25.38 | 9.16 | 54.25 |
| Nature | 99.03 | 99.03 | 99.02 | 98.99 | 98.63 | 90.27 | 97.19 |
| Office | 99.03 | 99.03 | 99.02 | 98.65 | 77.40 | 22.23 | 79.27 |
| Public | 99.03 | 99.02 | 99.01 | 98.17 | 73.01 | 12.99 | 76.44 |
| Street | 99.03 | 98.93 | 95.23 | 71.10 | 36.19 | 9.68 | 62.22 |
| Transportation | 99.03 | 99.03 | 98.99 | 97.48 | 57.02 | 28.65 | 76.23 |
| mean | 99.03 | 98.95 | 95.36 | 84.18 | 60.87 | 26.02 | |

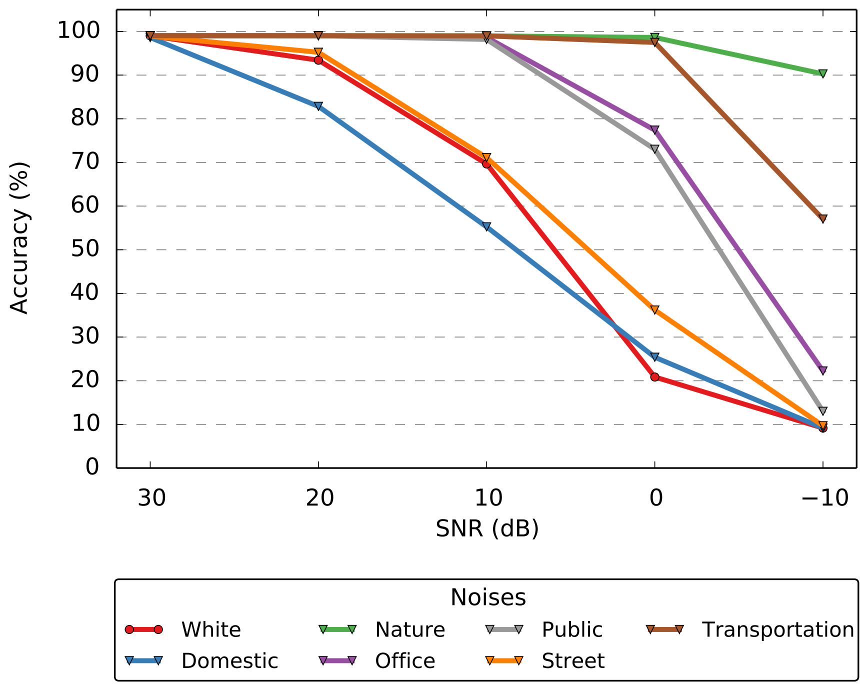
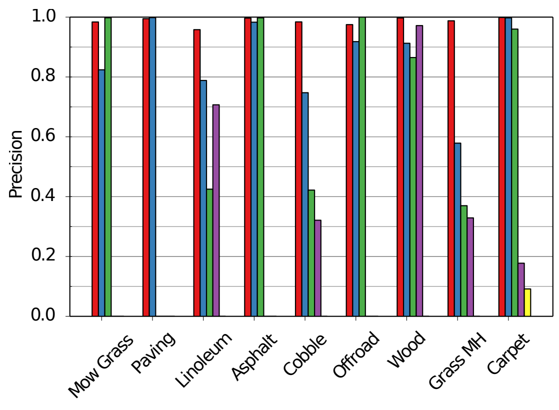
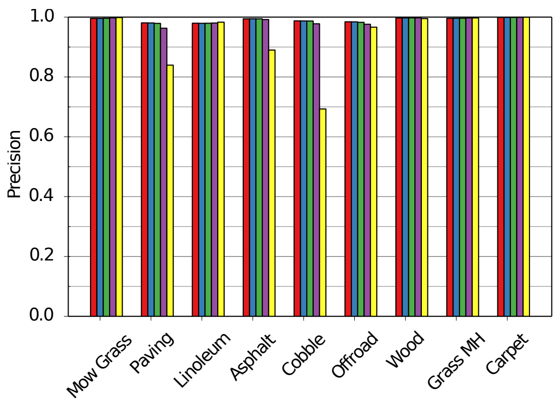


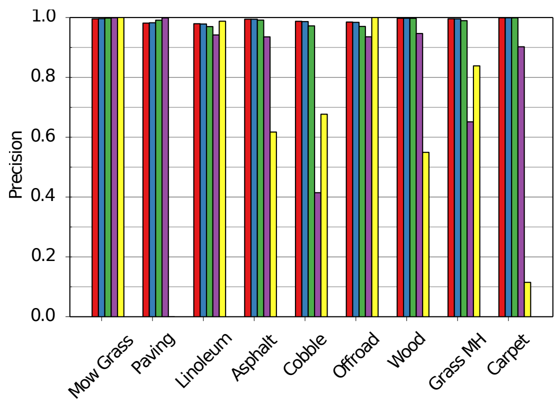
**Figure 3.13:** Performance evaluation of our TerrainNet++ model when subject to different ambient noises from the DEMAND noise database at varying SNRs. Our model demonstrates substantial robustness against ambient noises with SNRs of upto 20 dB and the performance quickly decreases for noises with SNRs lower than 20 dB.



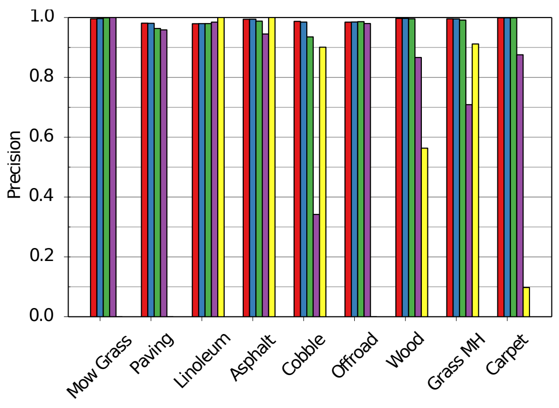
(a) Domestic Noise



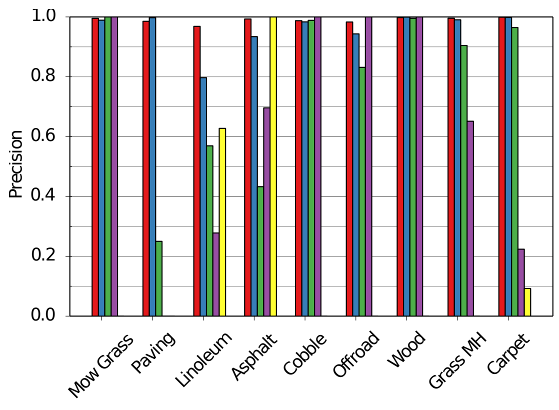
(b) Nature Noise



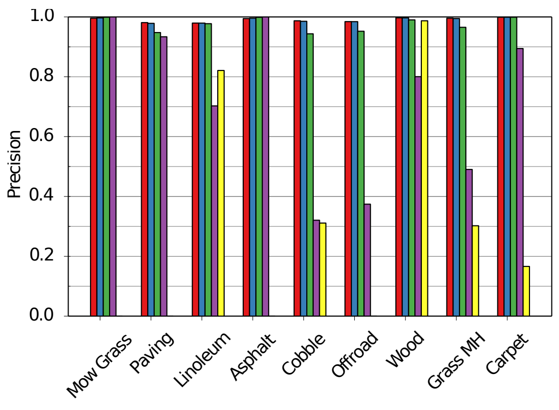
(c) Office Noise



(d) Public Noise



(e) Street Noise



(f) Transportation Noise

**Figure 3.14:** Per-class precision of our TerrainNet++ model when subject to various ambient noises from the DEMAND noise database at different SNRs mentioned in the legend. Terrains are still recognizable for ambient noises in the *Nature* category, even for SNRs lower than 0 dB.

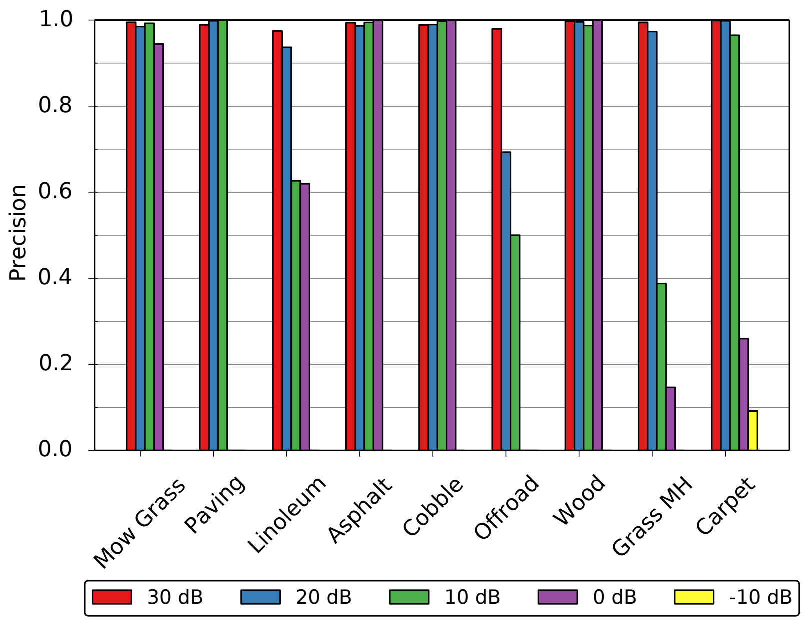


**Figure 3.15:** Per-class precision of our TerrainNet++ model when subject to different levels of white Gaussian noise. The levels mentioned in the legend are SNRs. White Gaussian noise is only damaging at very low SNRs.

Ambient noises in the *Domestic* category have the most damaging effect, where the mean accuracy at SNRs from 30 dB to –10 dB was 61.71%. From the results shown in Figure 3.13, we can see that the performance of the model drops linearly for decreasing SNRs for noises from the *Domestic* category. The second most damaging noises were from the *Street* category for which our model achieved a mean accuracy of 68.36%. Nevertheless, we observe remarkable robustness to ambient noises from the *Nature* category for which our model yields an average accuracy of 97.49% and even for extremely low SNRs. This is primarily due to the fact that the noises in the *Nature* category have very distinct spectral patterns that have prominent structures throughout the signal unlike vehicle-terrain interaction signals. Therefore, the model is inherently robust to these corruption patterns.

Often, it is also of interest to know how each of these ambient noises influence the classification of a specific terrain. Therefore, we computed the per-class precision of the TerrainNet++ model for each of the ambient noise categories. Results from this experiment is shown in Figure 3.14 for ambient noises from the DEMAND noise database and in Figure 3.15 for white noise. The *Carpet* terrain is recognizable for all the ambient noises and at all SNRs. While, the *Paving* terrain is the most affected by noise corruption, followed by *Offroad* and *Asphalt* terrains. Interestingly, classification of indoor terrains such as *Linoleum*, *Wood* and *Carpet* are less affected by indoor noises than outdoor terrains. However, the converse does not appear to be true, where the outdoor terrains are less affected by outdoor noises. This is due to the fact that, generally, outdoor ambient noises have continuous noise corruption patterns that span long periods, whereas, indoor ambient

**Table 3.6:** Influence of various ambient noises on the classification accuracy of our noise-aware TerrainNet++ model. There is a substantial increase in the robustness of the model to ambient noises from all the categories in the DEMAND noise database. Results are shown in terms of the average accuracy of the model in percent.

| Noise | SNR (dB) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Clean | 30 | 20 | 10 | 0 | -10 | mean |
| White | 99.72 | 99.68 | 98.77 | 97.63 | 97.11 | 96.38 | 97.91 |
| Domestic | 99.72 | 99.66 | 98.69 | 97.87 | 97.04 | 96.97 | 98.05 |
| Nature | 99.72 | 99.71 | 99.69 | 99.51 | 99.08 | 97.28 | 99.05 |
| Office | 99.72 | 99.72 | 99.65 | 98.70 | 98.61 | 96.00 | 98.53 |
| Public | 99.72 | 99.70 | 99.67 | 98.71 | 98.49 | 97.63 | 98.84 |
| Street | 99.72 | 99.68 | 98.73 | 97.90 | 97.81 | 96.36 | 98.10 |
| Transportation | 99.72 | 99.69 | 98.77 | 98.62 | 97.65 | 97.47 | 98.44 |
| mean | 99.72 | 99.69 | 99.13 | 98.42 | 97.97 | 96.87 | |

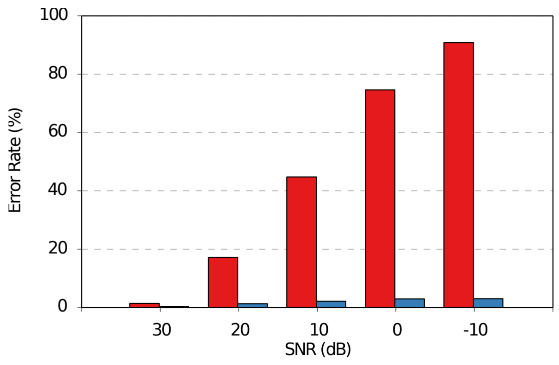
noises are usually short impulses.

Moreover, a curious phenomenon can be observed in the prediction of certain terrain classes when they are corrupted with ambient noises from categories such as *Office, Public, Street* and *Transportation*. In these cases, the model demonstrates a sudden increase in precision for samples that are noised at an SNR of –10 dB in comparison to samples that are noised at higher SNRs. We hypothesize that this is caused by the spectral responses of these noises that blur the inherently noisy lower frequency components of the vehicle-terrain interaction signals, which enhances the ability of the network to classify these terrains by focusing on the more distinct higher frequency responses.

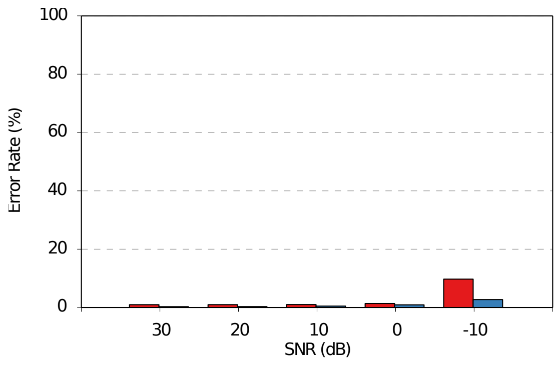
On the contrary, white noise severely affects the classification ability when the vehicle-terrain interaction signals are corrupted with SNRs below 0 dB. Terrain classes such as *Mowed-grass, Asphalt, Cobble* and *Wood* are robust to noise corruption upto a SNR of 0 dB. Whereas, classes such as *Linoleum, Offroad, Grass Medium-High* and *Carpet* have an exponential increase in classification error for decreasing SNRs. Furthermore, *Paving* and *Offroad* terrains are not recognizable for SNRs greater than 10 dB. These experiments demonstrate the critical need for noise robustness in order for these models to be effectively utilizable in high ambient noise environments.

### 3.4.6 Evaluation of Noise-Aware Training

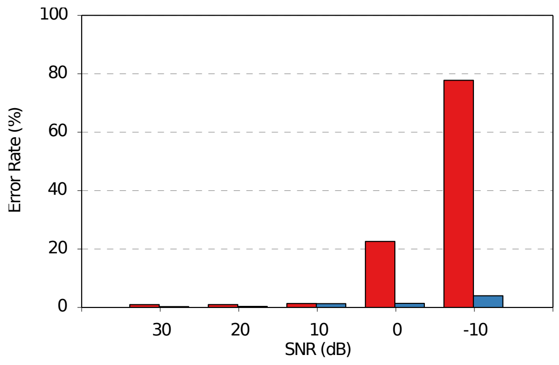
In order to make our model robust to the presence of ambient noises in the environment, we employ our proposed noise-aware training scheme described in Section 3.2.4. We use transfer learning to first initialize the network with weights and biases from the



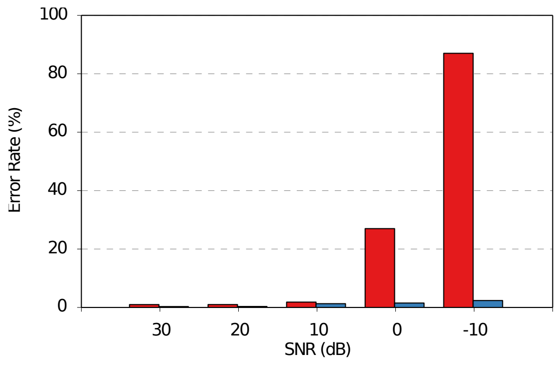
(a) Domestic Noise



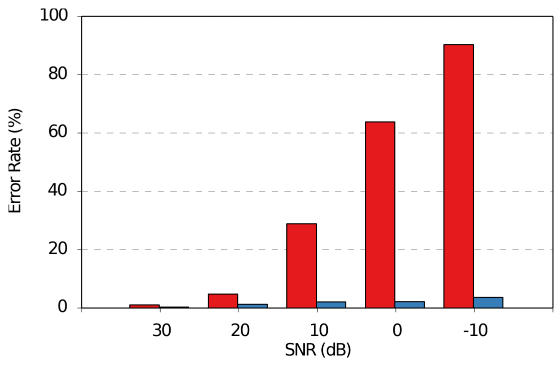
(b) Nature Noise



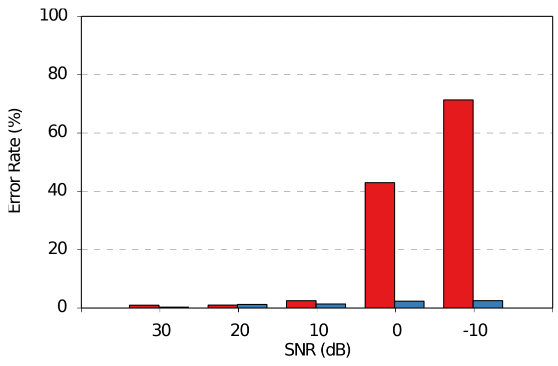
(c) Office Noise



(d) Public Noise



(e) Street Noise



(f) Transportation Noise

🟥 Base model 🟦 Noise-Aware model

**Figure 3.16:** Comparison of the classification error rates of our base TerrainNet++ model and our noise-aware TerrainNet++ model for each of the ambient noise categories in the DEMAND noise database. Our noise-aware TerrainNet++ model achieves substantially lower error rates compared to our base TerrainNet++ model.

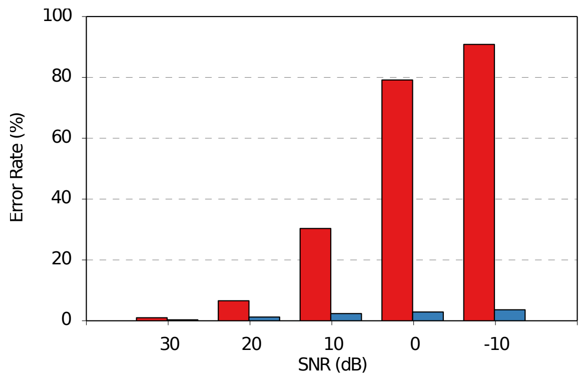


**Figure 3.17:** Comparison of classification error of our base TerrainNet++ model and our noise-aware TerrainNet++ for white Gaussian noise at varying SNRs. Our noise-aware model shows a significant decrease in the classification error compared to the model trained only on pure signals.

model trained on pure vehicle-terrain interaction signals. We then perform noise-aware training by injecting ambient noises from the DEMAND noise database at varying SNRs. We use a learning rate $1/10th$ of the initial leaning rate used for training the network. This noise injection enables the network to learn to distinguish between probable noise corruption patterns that may occur in real-world environments and the relevant vehicle-terrain interaction signals. We evaluate the models trained with our noise-aware scheme on the same noise corrupted test set used for the noise tolerance evaluation experiments presented in Section 3.4.5.

Table 3.6 shows the results from this experiment. It can be observed that there is a substantial increase in the overall classification accuracy for SNRs lower than 10 dB. Even for extremely low SNRs, our model demonstrates state-of-the-art performance. Strikingly, for a SNR of −10 dB, our noise-aware model achieves an improvement of 70.85% in the classification accuracy. Moreover, the noise-aware model shows an improved performance even for clean signals without any noise corruption. This verifies our hypothesis that noise-aware training can also act as a regularization for the network and improve its generalization capability.

Furthermore, in Figures 3.16 and 3.17, we compare the classification error rates of the model trained on pure signals and the noise-aware model for each of the noise categories in the DEMAND noise database and for white Gaussian noise. The plots further illustrate that there is a significant decrease in the classification error after the noise-aware training. Our base model trained on pure signals shows increasing classification error for decreasing SNRs, while our noise-aware model consistently demonstrates substantial robustness in each of the ambient noise categories. This demonstrates that our noise-aware training scheme enables the network to learn a general distribution of noise corruption signals for a wide range of SNRs as well as for different types of ambient noises, thereby making it a critically necessary step for robust terrain classification in noisy environments.

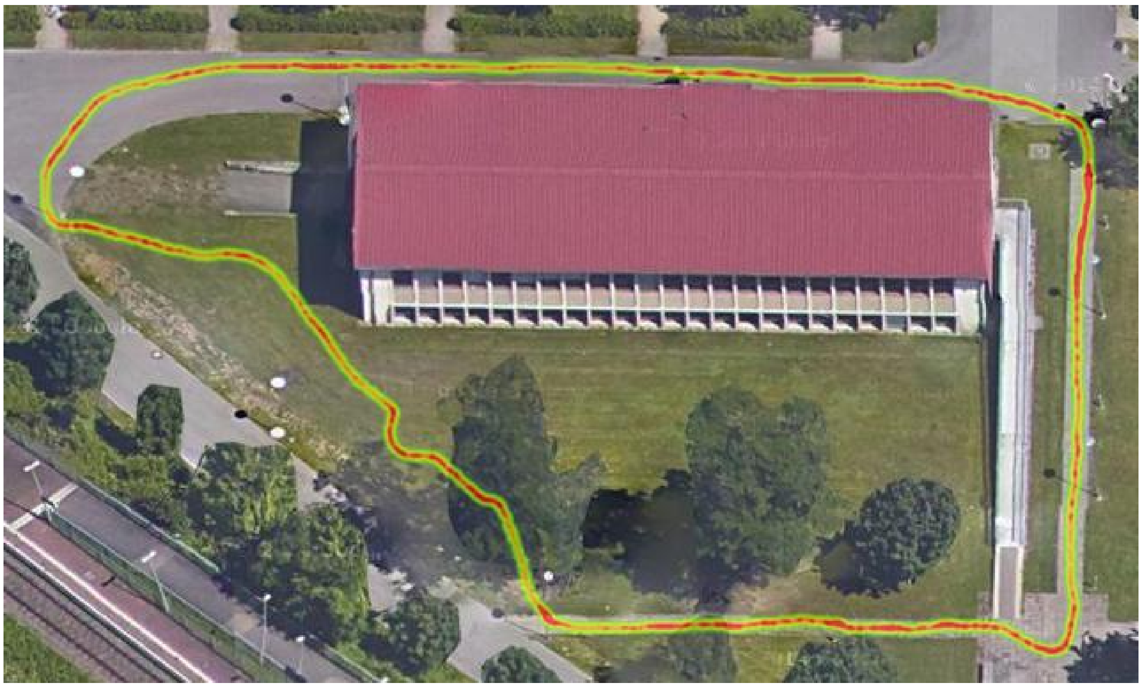


**Figure 3.18:** Map showing one of the trajectories that the robot followed during the classification experiments using a mobile phone microphone mounted on the robot. The variation in the robots speed along the path is shown as a heatmap on the trajectory. Thicker red lines indicate slower speed.

### 3.4.7 Evaluation of Hardware Independence

One of the biggest challenges for classification systems is to generalize to different hardware setups. The dataset that we introduced in this work is the only publicly available audio-based terrain classification dataset till date. Therefore, models trained on our dataset should generalize to different hardware setups in order to be easily employable in different robots. DCNN have the ability to learn highly discriminative deep features that generalize effectively to different environments as well as data acquisition setups. In order to evaluate the hardware independence of our models, we collected a supplementary dataset using a mobile phone microphone mounted on the robot. Moreover, in order to further evaluate the boundaries to which the model can generalize, we collected this dataset in a different region than where our main dataset was gathered. Note that we do not train our architecture on this mobile phone microphone dataset, we only test our noise-aware model that was trained on the main shotgun microphone dataset. Each sample in this dataset was tagged with a GPS location to visualize the trajectory of the robot in georeferenced maps and to visually correlate terrain. This dataset contains of over 2.5 h of vehicle-terrain interaction sounds. Unlike the shotgun microphone that we used for collecting our main dataset, mobile phones are equipped with a condenser microphone that collects sound from

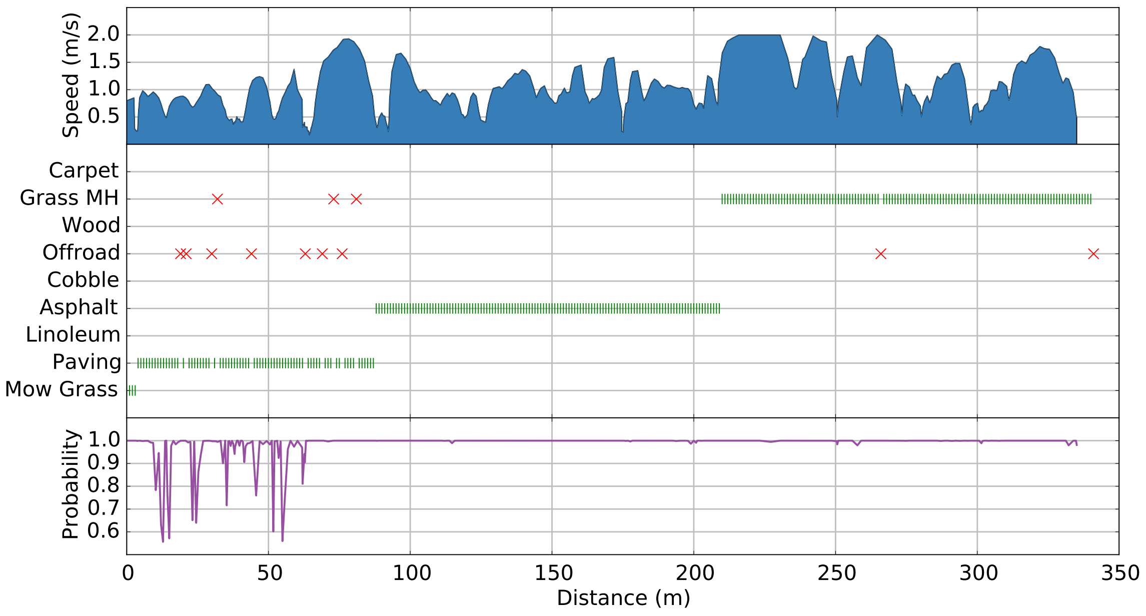


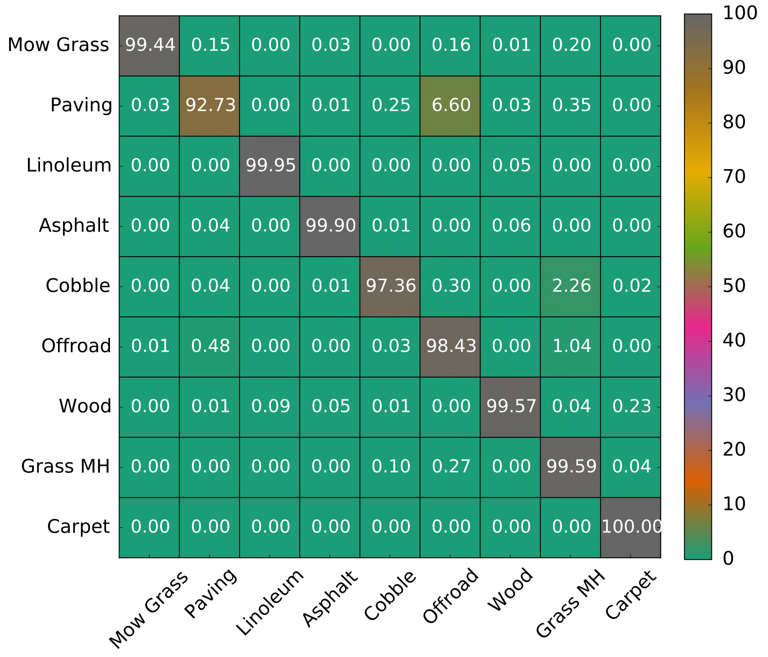
**Figure 3.19:** Terrain predictions made by our model while the robot was traversing the path shown in Figure 3.18. The model effectively generalizes to new environments and demonstrates substantial robustness.

every direction, including a considerable amount of ambient noise. The main purpose of evaluating on this dataset is threefold:

1. to quantify the performance of the model using a new hardware setup;
2. to verify the adaptability of the noise-aware model to a new environment; and
3. to quantify the performance in the presence of substantial amount of real-world ambient noise.

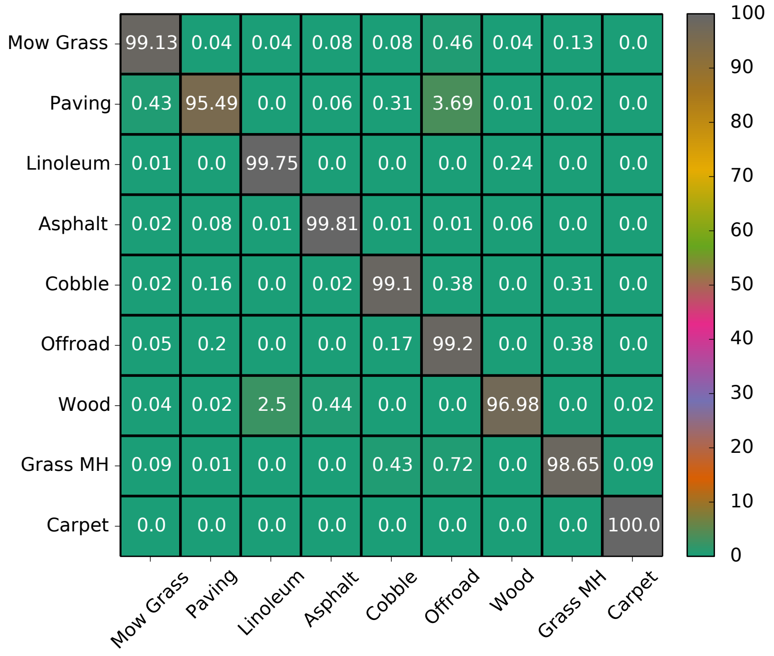
Figure 3.18 shows an example trajectory that the robot traversed during a part of this data collection run. The figure shows the variation in speed 0 - 2 $\mathrm{m\,s^{-1}}$ along the path as a heatmap on the trajectory, where thicker red lines indicate slower speed. Our noise-aware TerrainNet++ model achieves an accuracy of 98.68% on the mobile phone microphone dataset. In addition, the plot presented in Figure 3.19 shows the true positives and the false positives along the path traversed by the robot shown in Figure 3.18. Interestingly, most of the false positives occur when the robot is traversing on *Paving* and when the speed of the robot is above 1 $\mathrm{m\,s^{-1}}$. This can be attributed to the fact that the height of the *Paving* was often irregular in this case and when coupled with the higher speed caused it to be misclassified as the *Offroad* terrain.

In order to further investigate the performance on the mobile phone dataset, we present the confusion matrices of TerrainNet and TerrainNet++ models in Figure 3.20. *Paving, Wood* and *Linoleum* terrains demonstrate the highest misclasifications using the TerrainNet++ model. Nevertheless, compared to the TerrainNet model, it achieves a reduction



| | Mow Grass | Paving | Linoleum | Asphalt | Cobble | Offroad | Wood | Grass MH | Carpet |
|---|---|---|---|---|---|---|---|---|---|
| Mow Grass | 99.44 | 0.15 | 0.00 | 0.03 | 0.00 | 0.16 | 0.01 | 0.20 | 0.00 |
| Paving | 0.03 | 92.73 | 0.00 | 0.01 | 0.25 | 6.60 | 0.03 | 0.35 | 0.00 |
| Linoleum | 0.00 | 0.00 | 99.95 | 0.00 | 0.00 | 0.00 | 0.05 | 0.00 | 0.00 |
| Asphalt | 0.00 | 0.04 | 0.00 | 99.90 | 0.01 | 0.00 | 0.06 | 0.00 | 0.00 |
| Cobble | 0.00 | 0.04 | 0.00 | 0.01 | 97.36 | 0.30 | 0.00 | 2.26 | 0.02 |
| Offroad | 0.01 | 0.48 | 0.00 | 0.00 | 0.03 | 98.43 | 0.00 | 1.04 | 0.00 |
| Wood | 0.00 | 0.01 | 0.09 | 0.05 | 0.01 | 0.00 | 99.57 | 0.04 | 0.23 |
| Grass MH | 0.00 | 0.00 | 0.00 | 0.00 | 0.10 | 0.27 | 0.00 | 99.59 | 0.04 |
| Carpet | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 100.00 |

(a) Confusion matrix of our noise-aware TerrainNet model for a clip length of 300 ms



| | Mow Grass | Paving | Linoleum | Asphalt | Cobble | Offroad | Wood | Grass MH | Carpet |
|---|---|---|---|---|---|---|---|---|---|
| Mow Grass | 99.13 | 0.04 | 0.04 | 0.08 | 0.08 | 0.46 | 0.04 | 0.13 | 0.0 |
| Paving | 0.43 | 95.49 | 0.0 | 0.06 | 0.31 | 3.69 | 0.01 | 0.02 | 0.0 |
| Linoleum | 0.01 | 0.0 | 99.75 | 0.0 | 0.0 | 0.0 | 0.24 | 0.0 | 0.0 |
| Asphalt | 0.02 | 0.08 | 0.01 | 99.81 | 0.01 | 0.01 | 0.06 | 0.0 | 0.0 |
| Cobble | 0.02 | 0.16 | 0.0 | 0.02 | 99.1 | 0.38 | 0.0 | 0.31 | 0.0 |
| Offroad | 0.05 | 0.2 | 0.0 | 0.0 | 0.17 | 99.2 | 0.0 | 0.38 | 0.0 |
| Wood | 0.04 | 0.02 | 2.5 | 0.44 | 0.0 | 0.0 | 96.98 | 0.0 | 0.02 |
| Grass MH | 0.09 | 0.01 | 0.0 | 0.0 | 0.43 | 0.72 | 0.0 | 98.65 | 0.09 |
| Carpet | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 100.0 |

(b) Confusion matrix of our noise-aware TerrainNet++ model for a clip length of 200 ms and a LSTM window length of five.

**Figure 3.20:** Performance comparison on the audio data collected using a mobile phone microphone. There is a significant decrease in misclassifications between the *Paving* and *Ofroad* terrains, as well as *Cobble* and *Grass Medium-High* terrains.

of 3% in the misclassifications between the *Paving* and *Offroad* terrains, as well as a similar reduction of misclassifications between the *Cobble* and *Offroad* terrains. These experiments demonstrate that our models trained using our noise-aware training scheme not only generalize effectively to different environments but also to different hardware setups used for acquiring the data, thereby making our model effectively employable for robust audio-based terrain classification in different real-world environments.

## 3.5 Related Work

In this chapter, we addressed the problem of audio-based terrain classification using vehicle-terrain interaction sounds. Terrain classification is a critical and essential component of a robot's autonomy system that enables safe and efficient navigation on unknown terrains. Unlike conventional approaches that employ cameras or Lidars for terrain classification, using vehicle-terrain interaction sounds as a proprioceptive modality enables our system to be robust to visual appearance changes and allows us to perform classification of terrains at finer level. The techniques that we presented in this chapter enables a robot to utilize a low-cost microphone to robustly classify a wide range of indoor and outdoor terrains. To the best of our knowledge, our contribution is the first successful proprioceptive terrain classification system to not require any manual feature engineering. In this section, we discuss prior work in the literature on terrain classification using proprioceptive modalities.

Terrain classification using proprioceptive modalities has not been explored in the same depth as vision based approaches, yet there is a sizable amount of work in this area. The most researched proprioceptive terrain classification technique is using accelerometer data [80, 113, 114]. Classification of terrains is performed by extracting features such as power spectral density, discrete fourier transform and other statistical measures from the vibrations induced on the vehicles body. Such approaches demonstrate a substantial amount of false positives for finer terrain classes such as asphalt and carpet. However, accuracies as high as 91.8% has been reported for a seven class problem using SVMs [80]. Subsequently, accelerometer data from a mobile sensor network system was used to detect potholes and other road anomalies [79]. Hand-engineered features were used and the system achieves an average accuracy of 90% in real-world experiments.

There is a body of work in terrain classification tailored to legged robots. Unlike in wheeled mobile robots, proprioceptive terrain classification can enable safe foothold placement which is critical to ensure the stability of such legged systems. In one of the initial works [115], Hoepflinger *et al.* extracted features from ground contact forces and joint motor current measurements to train a multiclass AdaBoost classifier. Although they did not test the efficacy of their approach on real-world terrains, they demonstrated the capability of the classifier to identify different coarseness and curvatures of surfaces as a first step towards real-world proprioceptive terrain classification. In another approach [116], Best *et al.* demonstrate the ability to classify four different outdoor terrains using position measurements from leg servos of a hexapod robot. They extract a 600-dimensional feature vector consisting of gait-phase domain as well as frequency domain features. Their approach utilizes a 2.7 s window of data and a SVM is trained to classify the terrains.

Vibration data from contact microphones has also been successfully used for terrain classification. The vibrations captured from contact microphones are similar to accelerometer data than that of air microphones that we use in this work. Contact microphones pick up only structure-borne sound and minimal environmental noise. Brooks *et al.* equip the wheel of an analog rover with a contact microphone to capture the vibrations induced on the vehicles body and classify the terrain [75]. The approach extracts log-scale power spectral density features and uses a pairwise classifier. Their approach was evaluated on three terrain classes and achieves an average accuracy of 74% on a wheel-terrain testbed and 85.3% in the real-world experiments using a rover. In a subsequent work, they present a self-supervised classification approach where a visual classifier is trained using labels provided by a vibration-based terrain classifier [117].

The use of vehicle-terrain interaction sounds for terrain classification has been the most sparsely explored among all the proprioceptive modalities. In one of the early works, Durst *et al.* [118] proposed an approach for classifying terrain surfaces from single impact sounds that were produced when the surfaces were struck by an aluminum cane. The approach extracts the most significant spikes in the frequency domain as features and employs a decision-tree classifier to identify the surface type. In the works that followed, typically, combinations of state-of-the-art classical audio features are used with traditional machine learning algorithms. Recently, a multiclass audio-based terrain classification system [76] for mobile robots was proposed in which several audio features were incorporated including spectral coefficients, spectral moments and various other spectral and temporal statistics. The approach employs a SVM-based classifier that achieves an average accuracy of 78% over three terrain classes and three hazardous vehicle-terrain interaction classes. Furthermore, they also show that smoothing over larger temporal window of about 2 s yields an improved accuracy of 92%. Ojeda *et al.* propose an approach [113] in which a suite of sensors including microphones, accelerometers, motor current and voltage sensors, infrared, ultrasonics and encoders were used along with a feedforward neural network classifier. However, their audio-based classifier only achieves an average accuracy of 60.3% for five terrain classes. They concluded that the overall performance was poor and such an approach was promising only for terrains such as grass.

Microphones have also been used for terrain classification with legged robots. Ozkul *et al.* [78] propose an approach in which the terrain is classified from the steps taken by a hexapod robot. They use a feature set that includes zero-crossing rate, frequency band coefficients and delta-features along with a functional trees classifier. They also experiment with different noise elimination techniques to remove motor/gear-head noise, but concluded that the performance was worse after the noise elimination. Their seven class terrain classifier achieves an average accuracy of 90%. More recently, Christe *et al.* [77] introduced an approach that employs an SVM-based classifier using statistics of spectral and band features to classify terrains from the sounds produced during the locomotion of hexapod robot. Their classifier was trained on a dataset that contains 5 min audio clips from each terrain and uses 1 s windows for classification while operating at 1 Hz. Their approach achieves an average accuracy of 89.08% for seven terrain classes. They also investigated the use of spectral subtraction to eliminate the servo noise and report an improvement of 2.2% in the recall of the classifier.

In all of the previous works, manually handcrafted features were extracted after specialized preprocessing steps. The approaches were evaluated on comparatively limited data and only in one environment. Most importantly, these approaches do not model the temporally discriminative information in proprioceptive data. As we demonstrated in this chapter, modeling the temporal information substantially improves the classification performance, increases robustness and reduces false detections. Furthermore, they do not address the robustness of their models to ambient noise, which is one of the most critical requirements for real-world deployment. Whereas, in this chapter, we presented two DCNN architectures that learn to classify terrains from vehicle-terrain interaction sounds without the need for manually designing the features. Our recurrent DCNN architecture exploits the complex temporal dynamics in the vehicle-terrain interaction signal and has a classification rate of 81 Hz. We also introduced a noise-aware training scheme to improve the robustness of our model to different ambient noises, better regularize the network and to enable our model to effectively generalize to different real-world environments.

## 3.6 Conclusions

In this chapter, we presented a robust proprioceptive terrain classification approach that uses sound from vehicle terrain-interactions to classify a wide range of indoor and outdoor terrains. We proposed two novel deep convolutional neural network architectures that learn features from spectrograms of vehicle-terrain interaction signals. Both our architectures employ one dimensional convolution and pooling layers to learn spectrogram features across the temporal dimension. Additionally, our networks incorporate our proposed GSP module that aggregates the statistics of pooled features across time. In order to further learn the complex temporal dynamics in the vehicle-terrain interaction signal, our recurrent TerrainNet++ architecture incorporates long short-term memory units that exploit temporal relationships in the sequence of input audio frames. Furthermore, we introduced a noise-aware training scheme that injects ambient environmental noise from the diverse environments multichannel acoustic noise database into the pure vehicle-terrain interaction signals while training to improve the robustness of the model and to regularize the network. In order to facilitate this work, we collected an extensive vehicle-terrain interaction dataset consisting of over 15 h of audio data using two different hardware setups. Our dataset was collected on a total of nine different indoor as well as outdoor terrains and we made the dataset publicly available to foster future progress in proprioceptive terrain classification.

We presented extensive empirical evaluations that demonstrate that both our TerrainNet and recurrent TerrainNet++ architectures outperform existing approaches that employ classical audio features and achieve state-of-the-art results for proprioceptive terrain classification. In depth analysis of our TerrainNet++ architecture shows that exploiting the temporal dynamics by incorporating both our proposed GSP module and the LSTM unit substantially improves the classification performance while simultaneously increasing the classification rate. We presented thorough ablation studies on the influence of various hyperparameters on the classification performance of our network. Our best performing TerrainNet++ model achieves an overall accuracy of 99.03% with a classification rate of 81.7 Hz. Additionally, we presented extensive robustness evaluations of our models that demonstrate high ambient noise tolerance in various real-world environments. Furthermore, we presented thorough evaluations of our proposed noise-aware training scheme that enables our model to further improve its performance and achieve an accuracy of 99.72% while demonstrating exceptional robustness to a variety of ambient noises, even for very low signal-to-noise ratios. We also presented experiments using an inexpensive low-quality microphone in an unknown environment and demonstrated the generalization capability as well as the hardware independence of our approach. To the best of our knowledge, the approach presented in this chapter is the first proprioceptive terrain classification technique that does not require manual hand-engineering of features and achieves an accuracy of over 98% on a wide range of indoor as well as outdoor terrains while being over twice as fast as real-time.

# Chapter 4

# Semantic Scene Segmentation

**Semantic segmentation is an essential task for autonomous robots, as accurate understanding of the surroundings is a precursor for navigation and action planning. Recently, deep learning-based approaches have achieved unprecedented performance for various semantic segmentation problems, yet they face an immense challenges in terms of encoding multiscale information, incorporating global context, accurately capturing object boundaries and in achieving computational efficiency. In this chapter, we propose two CNN architectures that address the aforementioned challenges. Our first architecture termed AdapNet incorporates our multiscale residual units with atrous convolutions that effectively enlarges the receptive field of filters to incorporate a larger context. In addition, we employ modules with parallel atrous convolutions with different dilation rates to learn multiscale features without increasing the parameters. We also propose the AdapNet++ architecture that builds upon AdapNet and integrates our efficient atrous spatial pyramid pooling for aggregating multiscale information as well as capturing long range context, complemented with a novel decoder with a multi-resolution supervision scheme that recovers high-resolution details. Furthermore, we propose a network-wide holistic pruning approach invariant to shortcut connections. Extensive experiments on multiple indoor and outdoor benchmarks demonstrate that our architectures achieve state-of-the-art performance while being efficient in terms of parameters and inference time.**

## 4.1 Introduction

In the previous chapter, we proposed an approach to robustly classify different indoor and outdoor terrains in order to enable the robot to adapt its navigation policy so that it

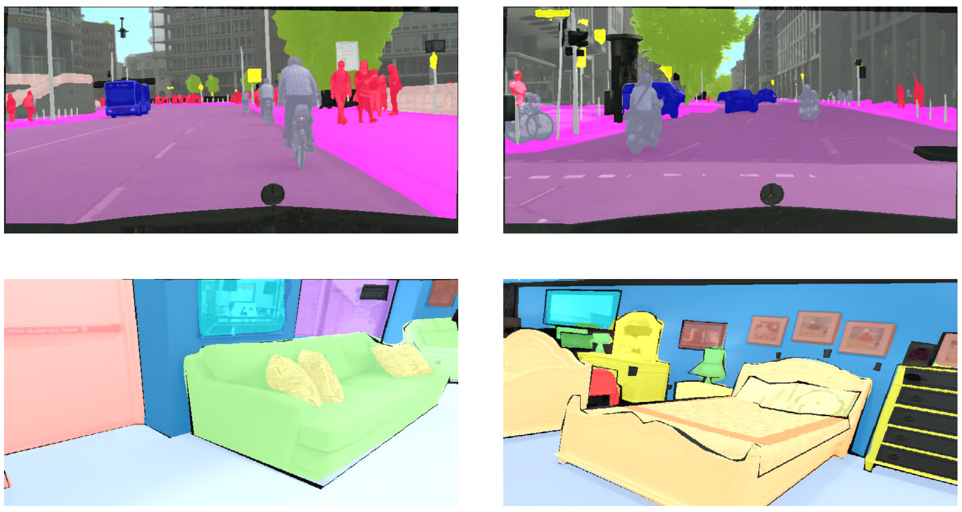


**Figure 4.1:** Example segmentation output from our proposed AdapNet++ model overlaid on the input image. Top row shows images from the Cityscapes dataset in a driving scenario and the bottom row shows images from the indoor SUN RGB-D dataset. Each color corresponds to a specific semantic object category in the dataset. The images illustrate the difficulty in segmentation due to the different scales of objects, substantial clutter and very thin structures that are hard to capture.

can traverse the terrain most effectively. In addition to being able to identify the different types of terrains, the robot simultaneously needs to acquire a comprehensive understanding of its surroundings before it can plan for future actions. Humans have the remarkable ability to instantaneously recognize and understand a complex visual scene which has piqued the interest of researches to model this ability since the 1960s [119]. Semantic segmentation with the goal of assigning semantic class labels to each pixel in an image is one of the fundamental tasks for modeling the complex relationship of semantic entities in the environment that pave the way towards complete scene understanding. There are numerous ever-expanding applications to this capability ranging from robotics [120] and remote sensing [121] to medical diagnostics [122] and content-based image retrieval [123]. Moreover, it has several applications in robot perception including inferring support relationships among objects [124], improving object detection [125], discarding regions that do not contain objects of interest [126], autonomous driving [127] and in combination with 3D scene geometry [128]. However, there are several challenges imposed by the multifaceted nature of this problem that makes this task extremely challenging including the large variation in types and scales of objects, clutter and occlusions in the scene as well as thin structures that are difficult to capture due to the inherent downsampling in the network. Example scenarios depicting these challenges are shown in Figure 4.1.

Classical techniques that rely on low-level vision cues [129, 130, 131] have recently been superseded by deep CNN based methods modeled as a Full Convolutional Network

(FCN) [24, 53, 132] that are trained in an end-to-end manner. This success can be attributed to the ability of CNNs to learn highly discriminative hierarchical deep features. The general structure of a FCN follows an encoder-decoder design principle, in which the encoder resembles the topology of a classification network and the decoder semantically projects the low-resolution discriminative features learned by the encoder onto the high-resolution pixel space to obtain a dense classification. However, unlike in pure classification networks where the built-in invariance of CNNs to local image transformations is beneficial, it is highly undesirable in dense prediction tasks where abstraction of spatial information affects the performance. Concretely, the following are the major challenges for employing CNNs for semantic segmentation:

- Decrease in feature resolution caused by pooling and striding.
- Existence of objects at multiple scales.
- Reduced localization accuracy due to invariance.
- Large model size and runtime.

In the rest of this section, we discuss each of the aforementioned challenges and describe the techniques that we employ to overcome them in our proposed architectures. CNNs designed for classification tasks that are often employed as the encoder in FCNs have multiple pooling or convolution layers with striding operations to learn increasingly abstract feature representations that are invariant to local image transformation. However, this significantly reduces the spatial resolution of feature maps at the output of the encoder, which have rich semantic information. Due to this factor, details on the object boundaries and thin structures are lost and cannot be recovered even using deconvolutional layers that learn the up-sampling [24]. In order to alleviate this problem, we reduce the number of downsampling operations in the encoder section of our network and incorporate our proposed multiscale residual units that introduce atrous convolutions [67] in parallel to the standard convolutions to extract denser feature maps. This enables our multiscale residual units to enlarge the receptive field of the filters thereby encoding higher level semantics without increasing the number of parameters or the computational operations.

The second challenge arises due to the presence of objects in different scales in the real-world. Over the years, several techniques have been proposed to address this problem including using multiple rescaled versions of the same image as input to the network [133, 134], using spatial pyramid pooling [135], exploiting features from each downsampling stage in the encoder-decoder topology [136] and employing multiple encoder branches with each taking a different resolution of the image as input [137]. Though these approaches have been reasonably effective, they substantially increase the computational overhead and memory requirement. Alternatively, as we employ atrous convolutions in our proposed multiscale residual units, incorporating multiple such units into the architecture with different dilation rates can enable effective multiscale feature learning throughout the network. DeepLab [138] proposed a similar module called Atrous Spatial Pyramid Pooling

(ASPP) that concatenates feature maps from atrous convolutions with different dilation rates to encode multiscale information. However, input images of large resolutions cannot be employed in order for it be effective, as it requires correspondingly increasing the dilation rate which causes atrous convolutions to become more ineffective due to image boundary effects [132]. Moreover, ASPP consumes 15.53 M parameters and 34.58 B floating point operations per second (FLOPS) which is prohibitively expensive to employ in architectures that are used in robotics applications for which computational efficiency is paramount. In addition, as ASPP uses a large number of high-dimensional and high-resolution feature maps, it requires an enormous amount of GPU memory while training which also restricts the incorporation of modules to address the other major challenges. As a solution to this problem we propose an efficient variant of the ASPP termed eASPP that employs both parallel and cascaded atrous convolutions in a bottleneck fashion to increase the resolution of the effective receptive field while significantly reducing the number of parameters it consumes.

Another challenge lies in the inherent invariance to spatial transformations in CNNs that limits the spatial localization accuracy of the features. Moreover, atrous convolutions coarsely sub-sample the features which leads to the loss of important details along object boundaries. In order to generate high-resolution predictions in the decoder, networks often utilize skip connections to leverage features from multiple intermediate network stages that retain spatial information and describe mid-level representations of objects [24, 139]. These features are complementary to the high-level semantic information at the end of the decoder that lack strong spatial information as well as to low-level features from early network layers that represent edges and corners but also maintain the spatial information. In our proposed architectures we employ such skip connections to leverage both low-level and mid-level features to improve the resolution of the segmentation output to generate sharp boundaries along the edges of objects. In addition, we also propose a multi-resolution supervision strategy that introduces weighted auxiliary losses after each upsampling stage in the decoder to deeply supervise the training and to further help in generating a high-resolution segmentation output. This also enables faster convergence, in addition to improving the performance of the model along the object boundaries.

State-of-the-art semantic segmentation architectures such as DeepLab v3 [132] and PSPnet [135] employ the ResNet-101 [27] architecture which consumes 42.39 M parameters and 113.96 B FLOPS, as the encoder backbone. Training such architectures requires a large amount of memory and synchronized training across multiple GPUs. Moreover, they have a slow runtime rendering them impractical for resource constrained applications such as robotics and augmented reality. With the goal of achieving the right trade-off between performance and computational complexity, we propose the AdapNet and AdapNet++ architectures for semantic segmentation. Our AdapNet architecture employs the ResNet-50 [27] architecture that consumes 25.61 M parameters for the encoder backbone and incorporates our proposed multiscale residual units to aggregate multiscale features throughout the network without increasing the number of parameters. The proposed units are more effective in learning multiscale features than the commonly employed multigrid approach introduced in DeepLab v3 [132]. AdapNet uses deconvolution layers and a single stage skip refinement to yield a segmentation output with the same resolution as the input image. In order to further improve the performance, we propose the AdapNet++ architecture that builds upon AdapNet but uses the full pre-activation ResNet-50 [72] architecture for the encoder backbone in order to reduce overfitting and improve convergence. In addition, we incorporate our proposed eASPP at the end of the encoder to capture long range context with a larger effective receptive field, while simultaneously reducing the number of parameters by 87% in comparison to the originally proposed ASPP. We also employ a new decoder that integrates low and mid-level features from the encoder using multiple skip refinement stages and employs a multi-resolution supervision strategy for high-resolution segmentation. Both our proposed architectures are compact and end-to-end trainable with a large mini-batch size on a single consumer grade GPU.

Motivated by the recent success of compressing DCNNs by pruning unimportant neurons [140, 141, 142], we explore pruning entire convolutional feature maps of our model to further reduce the number of parameters. Network pruning approaches utilize a cost function to first rank the importance of neurons, followed by removing the least important neurons and fine-tuning the network to recover any loss in accuracy. Thus far, these approaches have only been employed for pruning convolutional layers that do not have an identity or a projection shortcut connection. Pruning residual feature maps (third convolutional layer of a residual unit) also necessitates pruning the projected feature maps in the same configuration in order to maintain the shortcut connection. This leads to a significant drop in accuracy, therefore current approaches omit pruning convolutional filters with shortcut connections. As a solution to this problem, we propose a network-wide holistic pruning approach that employs a simple and yet effective strategy for pruning convolutional filters invariant to the presence of shortcut connections. This enables our network to further reduce the number of parameters and computing operations, making our model efficiently deployable even in resource constrained applications.

Most work in semantic segmentation thus far has been focused on indoor and outdoor urban environments where the scene is highly structured. However, there exists an increasing number of applications where mobile robots are tasked to operate in unstructured outdoor environments that have significantly more variation in semantic object classes and a substantial amount of perceptual disturbances due to changes in weather conditions. In order to also evaluate the performance our proposed architectures in such scenarios, we introduce a first-of-a-kind dataset collected in unstructured forested environments that we made publicly available. We present extensive experimental evaluations of our architectures on benchmark scene understanding datasets including Cityscapes [143], Synthia [144], SUN RGB-D [145] and ScanNet [146] as well as on our Freiburg Forest [50] dataset. The results demonstrate that our architectures set the new state-of-the-art on all these benchmarks while being computationally more efficient and with a faster inference time.

In summary, the following are the main contributions that we make in this chapter:

- The novel AdapNet architecture for semantic segmentation that incorporates our multiscale residual units to effectively learn multiscale features throughout the network and a decoder with skip refinement for high-resolution segmentation.
- The AdapNet++ architecture that further improves the performance by integrating a new efficient ASPP (eASPP) to aggregate multiscale information, a new decoder with multiple skip refinement stages and a multi-resolution supervision strategy for finer segmentation along the object boundaries.
- The eASPP for efficiently aggregating multiscale features and capturing long range context, while having a larger effective receptive field and over 10 times reduction in parameters compared to the standard ASPP.
- A holistic network-wide pruning approach that enables pruning of convolutional filters invariant to the presence of identity or projection shortcuts.
- A novel semantic segmentation dataset of unstructured forested environments consisting of multiple modalities and spectra with semantically annotated pixel-level groundtruth labels.
- Extensive benchmarking of existing approaches with the same input image size and evaluation setting along with quantitative and qualitative evaluations of our proposed architectures on five different benchmark datasets consisting of indoor environments, outdoor driving scenarios and unstructured forested environments.
- Implementations of our proposed models are made publicly available at `http://deepscene.cs.uni-freiburg.de`.

The remainder of this chapter is organized as follows. In Section 4.2, we describe our semantic segmentation architectures and our technique for pruning convolutional filters. We then describe the methodology that we employed for collecting the Freiburg Forest dataset in Section 4.3. In Section 4.4, we present extensive empirical evaluations, detailed ablation studies on the design choices and model variants, followed by qualitative evaluations on each of the datasets that we benchmark on. Finally, we discuss recent related work on semantic segmentation in Section 4.5 and conclude with a discussion in Section 4.6.

## 4.2 Technical Approach

In this section, we first describe the topology of our proposed AdapNet architecture and then detail the structure of our AdapNet++ architecture as well as its constituting components. Subsequently, we present our pruning strategy for network compression that is invariant to the presence of identity or projection shortcuts in the residual units.

### 4.2.1 AdapNet Architecture

Our architecture shown in Figure 4.2 follows the general fully convolutional encoder-decoder design principle. The encoder has a topology similar to classification networks and downsamples the input image, while the decoder upsamples the feature maps back to the input image resolution. In the following sections, we describe the components of our proposed AdapNet architecture.

#### 4.2.1.1 AdapNet Encoder

Encoders are the foundation of FCN architectures. Therefore, it is essential to build upon a good baseline that has a high representational ability conforming with the computational budget. Our critical requirement is to achieve the right trade-off between the accuracy of segmentation and inference time on a consumer grade GPU, while keeping the number of parameters low. Therefore, in contrast to the previous approaches, we build upon the ResNet-50 [27] architecture as it offers a good trade-off between learning highly discriminative deep features and the computational complexity required. It includes batch normalization and convolution layers with skip connections. This allows the design of much deeper networks without facing degradation of the gradient and therefore leads to very large receptive fields. The ResNet architecture has four computational blocks with varying number of residual units. We use the bottleneck residual units in our encoder as they are computationally more efficient than the baseline residual units and they enable us to build more complex models that are easily trainable.

In the ResNet architecture, the resolution of the feature maps drops down to a fourth of the input resolution after passing through the first 3 layers. On the one hand, this allows for context aggregation and speed-up due to smaller feature maps, but on the other hand, this restricts the learning of high-resolution features, which could potentially be useful in the later stages. In our AdapNet architecture, we introduced an additional convolution with a kernel size of $3 \times 3$ before the first convolution layer in the ResNet-50 architecture. This enables the network to learn more high-resolution features without significantly increasing the inference time. The output of the last residual block of the ResNet-50 architecture is 32-times downsampled with respect to the input image resolution. In order to increase the spatial density of the feature responses and to prevent signal decimation, we set the stride of the convolution layer in the last block (*Res4a*) from two to one which makes the resolution of the output feature maps 1/16-times the input image resolution. This enables the network to retain the higher resolution details while aggregating the same amount of context as before. We then replace the residual blocks that follow this last downsampling stage with our proposed multiscale residual units.

**Multiscale Residual Units:** A naive approach to compute the feature responses at the full image resolution would be to remove the downsampling and replace all the convolution

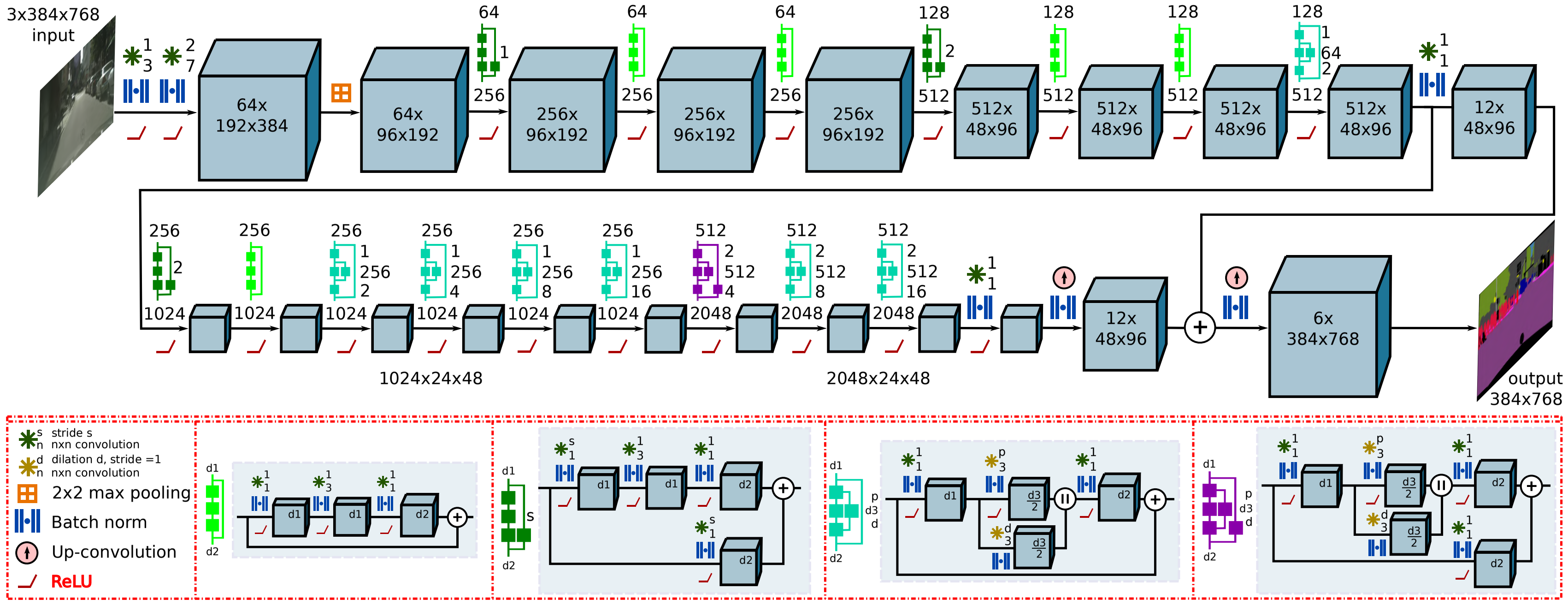


**Figure 4.2:** Depiction of our proposed AdapNet architecture for semantic segmentation. We employ the ResNet-50 architecture for the encoder backbone and additionally add a convolution layer at the beginning of the network, followed by changing the stride in the convolution layer of the last residual block from two to one. We also change the convolution layers that follow to an atrous convolution with a dilation of $r = 2$. The legend enclosed in red lines show the original residual units in the bottom left (light green and dark green), while our proposed multiscale residual units are shown in the bottom right (cyan and purple).

layers with atrous convolutions having a dilation rate $r \geq 2$ but this would be both computationally and memory intensive. Therefore, we propose the novel multiscale residual unit to efficiently enlarge the receptive field and aggregate multiscale features without increasing the number of parameters and the computational burden. Specifically, we replace the $3 \times 3$ convolution in the standard residual unit with two parallel $3 \times 3$ atrous convolutions with different dilation rates and half the number of feature maps each. We then concatenate their outputs before the following $1 \times 1$ convolution. By concatenating their outputs, the network additionally learns to combine the feature maps of different scales. Now, by setting the dilation rate in one of the $3 \times 3$ convolutional layers to $r = 1$ and another to a rate $r \geq 2$, we can preserve the original scale of the features within the block and simultaneously add larger context. While, by varying the dilation rates in each of the parallel $3 \times 3$ convolutions, we can enable the network to effectively learn multiscale representations at different stages of the network. The topology of the proposed multiscale residual units and the corresponding original residual units are shown in the legend in Figure 4.2. The lower left two units show the original configuration, while the lower right two units show the proposed configuration.

We incorporate the first multiscale residual unit with $r_1 = 1$ and $r_2 = 2$ before the third block at *Res3d* (unit before the block where we remove the downsampling as mentioned earlier). Subsequently, we replace the units *Res4c, Res4d, Res4e, and Res4f* with our proposed multiscale units with rates $r_1 = 1$ in all the units and $r_2 = 2, 4, 8, 16$ correspondingly. In addition, we replace the last three units of block four *Res5a, Res5b, and Res5c* with the multiscale units with increasing rates in both the $3 \times 3$ convolutions, as $(r_1 = 2, r_2 = 4)$, $(r_1 = 2, r_2 = 8)$, and $(r_1 = 2, r_2 = 16)$ correspondingly. We evaluate our proposed configuration in comparison to the multigrid method of DeepLab v3 [132] in the ablation study presented in Section 4.4.6.1.

#### 4.2.1.2 AdapNet Decoder

The output of the encoder is 16-times downsampled with respect to the input image and it contains 2, 048 feature channels. In order to obtain the segmented image, we first employ a $1 \times 1$ convolution layer to reduce the number of feature channels to the number of semantic object categories in the dataset. We then upsample feature maps by two times using a deconvolution layer and perform refinement by fusing mid-level features from *Res3d* of the encoder. Subsequently, we employ a second deconvolution layer to upsample the predictions by eight times to yield an output with the same dimensions as the input image. In comparison to the initial FCN architecture [24], our proposed AdapNet architecture has a significantly lesser number of parameters and a substantially faster inference time, which are both critical requirements for models to be employed in real-world robotic perception applications. Moreover, the network converges in 12 hours while training on an NVIDIA Titan X GPU, whereas FCNs take about three days for the model to converge.

### 4.2.2 AdapNet++ Architecture

In an effort to further improve the segmentation performance, we propose the AdapNet++ architecture that builds upon AdapNet. We first briefly describe the overall topology and our main contributions motivated by our design criteria. We then detail each of the constituting architectural components. Similar to AdapNet, our network follows the general fully convolutional encoder-decoder design principle as shown in Figure 4.3. The encoder (depicted in blue) is based on the full pre-activation ResNet-50 [72] model as it has been demonstrated to reduce overfitting and improve the convergence while compared to the standard ResNet architecture. We incorporate our multiscale residual units at varying dilation rates in the last two blocks of the encoder to effectively compute high-resolution feature responses at different spatial densities. Figure 4.4 shows the topology of the encoder in more detail. In addition, to enable our model to capture long-range context and to further learn multiscale representations, we propose an efficient variant of the atrous spatial pyramid pooling module known as eASPP which has a larger effective receptive field and reduces the number of parameters required by over 87% compared to the originally proposed ASPP in DeepLab v3 [132]. We append the proposed eASPP after the last residual block of the encoder, shown as green blocks in Figure 4.3.

In order to recover the segmentation details from the low spatial resolution output of the encoder section, we propose a new deep decoder consisting of multiple deconvolution and convolution layers. Additionally, we employ multiple skip refinement stages that fuse low and mid-level features from the encoder with the upsampled decoder feature maps for object boundary refinement. Furthermore, we add two auxiliary supervision branches after each upsampling stage to accelerate training and improve the gradient propagation in the network. We depict the decoder as orange blocks and the skip refinement stages as gray blocks in the network architecture shown in Figure 4.3. In the following sections, we discuss each of the aforementioned network components in detail and elaborate on the design choices.

#### 4.2.2.1 Efficient Atrous Spatial Pyramid Pooling

In this section, we first describe the topology of the Atrous Spatial Pyramid Pooling (ASPP) module, followed by the structure of our proposed efficient Atrous Spatial Pyramid Pooling (eASPP). ASPP has become prevalent in most state-of-the-art architectures due to its ability to capture long range context and multiscale information. Inspired by spatial pyramid pooling [147], the initially proposed ASPP in DeepLab v2 [148] employs four parallel atrous convolutions with different dilation rates. Concatenating the outputs of multiple parallel atrous convolutions aggregates multiscale context with different receptive field resolutions. However, as illustrated in the subsequent DeepLab v3 [132], applying extremely large dilation rates inhibits capturing long range context due to image boundary effects. Therefore, an improved version of ASPP was proposed [132] to add global context

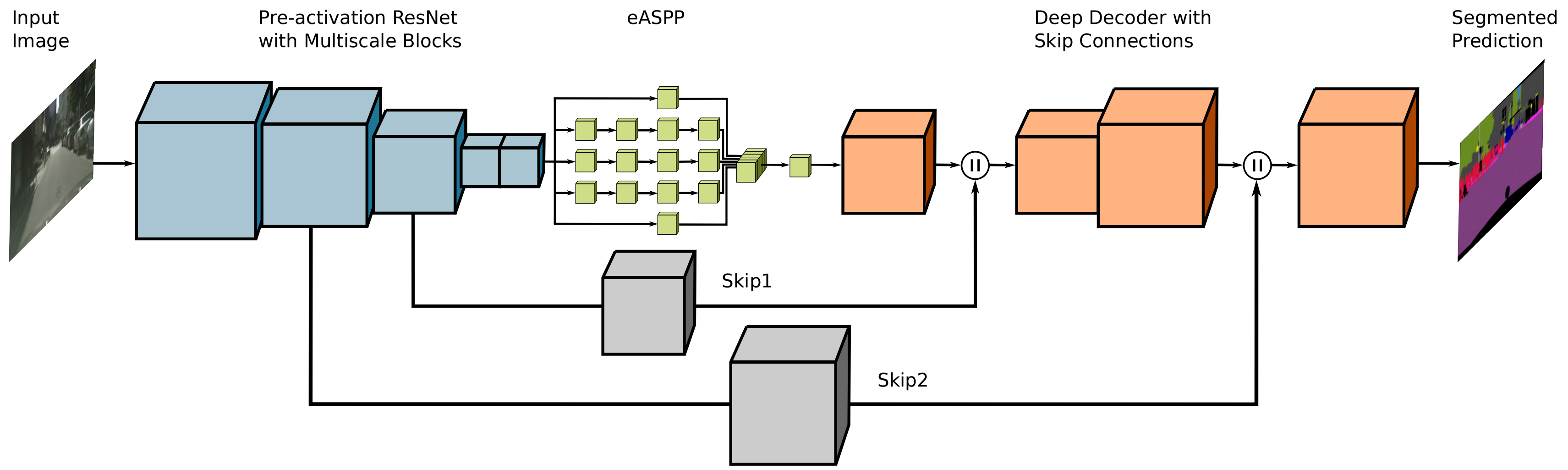


**Figure 4.3:** Overview of our proposed Adapnet++ architecture. Given an input image, we use the full pre-activation ResNet-50 architecture augmented with our proposed multiscale residual blocks to yield a feature map 16-times downsampled with respect to the input image resolution, then our proposed efficient Atrous Spatial Pyramid Pooling (eASPP) module is employed to further learn multiscale features and to capture long range context. Finally, the output of the eASPP is fed into our proposed deep decoder with skip connections for upsampling and refining the semantic pixel-level prediction.

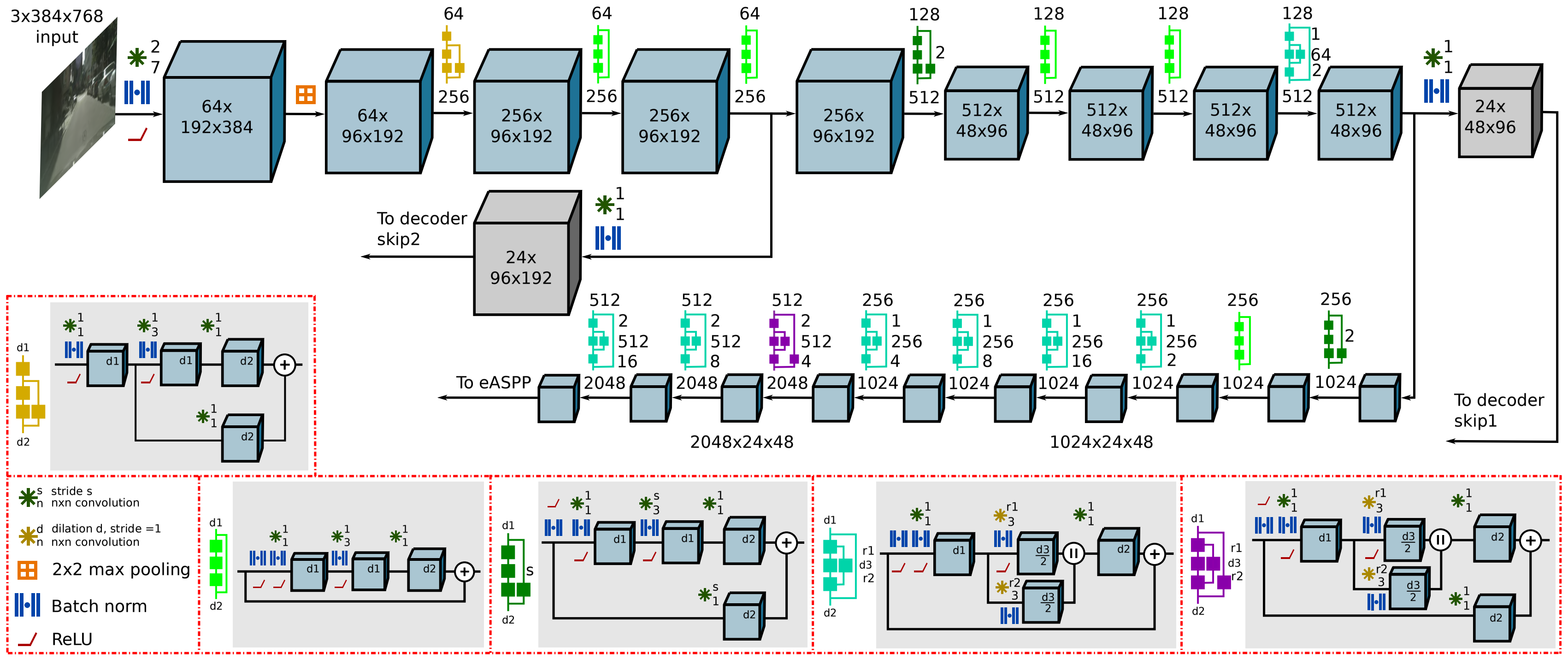


**Figure 4.4:** The proposed encoder is built upon the full pre-activation ResNet-50 architecture. Specifically, we remove the last downsampling stage in ResNet-50 by setting the stride from two to one, therefore the final output of the encoder is 16-times downsampled with respect to the input. We then replace the residual units that follow the last downsampling stage with our proposed multiscale residual units. The legend enclosed in red lines show the original pre-activation residual units in the bottom left (yellow, light green and dark green), while our proposed multiscale residual units are shown in the bottom right (cyan and purple).

information by incorporating image-level features.

The resulting ASPP shown in Figure 4.5 (a) consists of five parallel branches: one $1 \times 1$ convolution and three $3 \times 3$ convolutions with different dilation rates. Additionally, image-level features are introduced by applying global average pooling on the input feature map, followed by a $1 \times 1$ convolution and bilinear upsampling to yield an output with the same dimensions as the input feature map. All the convolutions have 256 filters and batch normalization layers to improve training. Finally, the resulting feature maps from each of the parallel branches are concatenated and passed through another $1 \times 1$ convolution with batch normalization to yield 256 output filters. The ASPP module is appended after the last residual block of the encoder where the feature maps are of dimensions $65 \times 65$ in the DeepLab v3 architecture [132], therefore dilation rates of 6, 12 and 18 were used in the parallel $3 \times 3$ atrous convolution layers. However, as we use a smaller input image, the dimensions of the input feature map to the ASPP is $24 \times 48$, therefore, we reduce the dilation rates to 3, 6 and 12 in the $3 \times 3$ atrous convolution layers respectively.

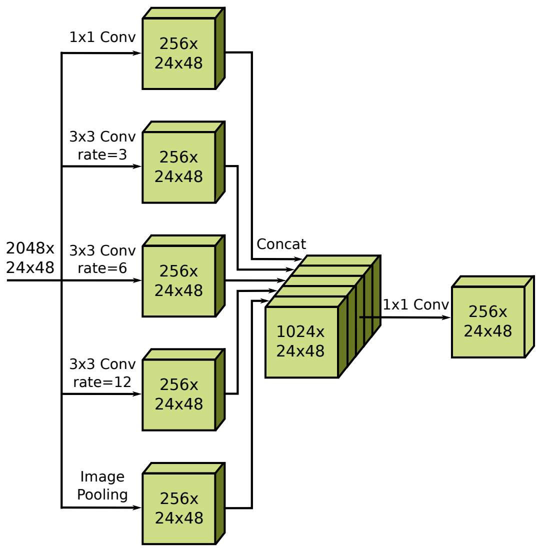
The biggest caveat of employing the ASPP is the extremely large amount of parameters and FLOPS that it consumes. Each of the $3 \times 3$ convolutions have 256 filters, which in total for the entire ASPP amounts to 15.53 M parameters and 34.58 B FLOPS which is prohibitively expensive. To address this problem, we propose an equivalent structure called eASPP that substantially reduces the computational complexity. Our proposed topology is based on two principles: cascading atrous convolutions and the bottleneck structure. Cascading atrous convolutions effectively enlarges the receptive field as the latter atrous convolution takes the output of the former atrous convolution. The receptive field size $F$ of an atrous convolution can be computed as

$$F = (r - 1) \cdot (N - 1) + N, \tag{4.1}$$

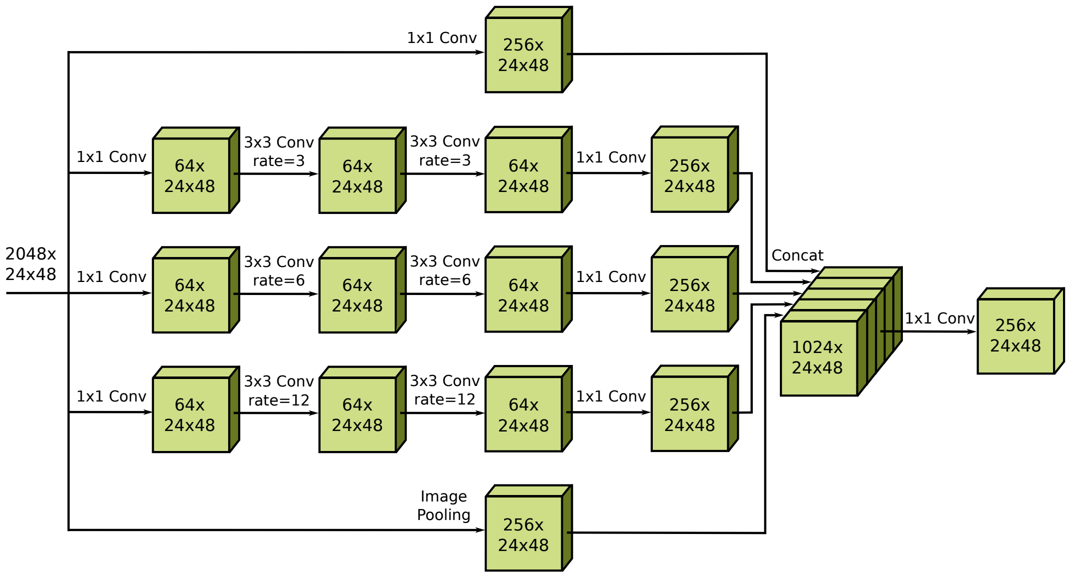
where r is the dilation rate of the atrous convolution and $N$ is the filter size. When two atrous convolutions with the receptive field sizes as $F_1$ and $F_2$ are cascaded, the effective receptive field size is computed as

$$F_{eff} = F_1 + F_2 - 1. \tag{4.2}$$

For example, if two atrous convolutions with filter size $F = 3$ and dilation $r = 3$ are cascaded, then each of the convolutions individually has a receptive field size of 7, while the effective receptive field size of the second atrous convolution is 13. Moreover, cascading atrous convolutions enable denser sampling of pixels in comparison to parallel atrous convolutions. We illustrate this phenomenon in Figure 4.6. Atrous convolutions increase the size of the receptive field by inserting holes between consecutive filter values, but the number of pixels that are sampled for the computation is still sparse. This factor becomes worse in the case of two-dimensional convolutions when large dilation rates are used. However, by concatenating atrous convolutions, the convolution in the upper layer

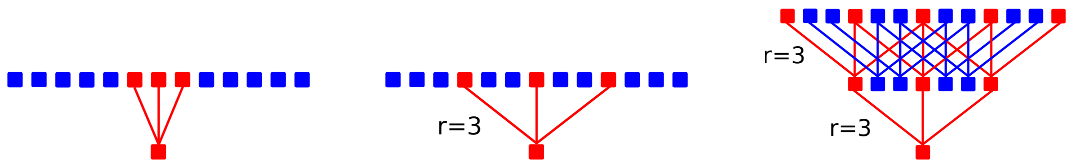


(a) Atrous Spatial Pyramid Pooling (ASPP)



(b) Efficient Atrous Spatial Pyramid Pooling (eASPP)

**Figure 4.5:** Depiction of the ASPP module from DeepLab v3 and our proposed efficient eASPP module. eASPP reduces the number of parameters by 87.87% and the number of FLOPS by 89.88%, while simultaneously achieving an improved performance. Note that all the convolution layers have batch normalization and we change the corresponding dilation rates in the $3 \times 3$ convolutions in ASPP to 3,6,12 as the input feature map to the ASPP is of dimensions $48 \times 23$ in our network architecture.



(a) Standard convolution (b) Atrous convolution (c) Stacking atrous convs

**Figure 4.6:** Illustration of pixel sampling in different 1-D convolutions. Stacking atrous convolutions yields a denser sampling rate with a very large receptive field. *r* denotes the dilation rate of the atrous convolution layer and red pixels denote the information captured by the convolution.

can utilize features from the lower layer and hence sample the pixels in a much denser fashion. Therefore, by using both parallel and cascaded atrous convolutions in the ASPP, we can efficiently aggregate dense multiscale features with very large receptive fields.

In order to reduce the number of parameters in the ASPP topology, we employ a bottleneck structure in the cascaded atrous convolution branches. The topology of our proposed eASPP shown in Figure 4.5 (b) consists of five parallel branches similar to ASPP but the branches with the $3 \times 3$ atrous convolutions are replaced with our cascaded bottleneck branches. If $c$ is the number of channels in the $3 \times 3$ atrous convolution, we add a $1 \times 1$ convolution with $c/4$ filters before the atrous convolution to squeeze only the most relevant information through the bottleneck. We then replace the $3 \times 3$ atrous convolution with two cascaded $3 \times 3$ atrous convolutions with $c/4$ filters, followed by another $1 \times 1$ convolution to restore the number of filters to $c$. The proposed eASPP only has 2.04 M parameters and consumes 3.5 B FLOPS which accounts to a reduction of 87.87% of parameters and 89.88% of FLOPS in comparison to the ASPP. We evaluate our proposed eASPP in comparison to ASPP in the ablation study presented in Section 4.4.6.2 and show that it achieves improved performance while being more than 10 times efficient in the number of parameters.

#### 4.2.2.2 AdapNet++ Decoder

The output of the eASPP in our network is 16-times downsampled with respect to the input image and therefore it has to be upsampled back to the full input resolution. As described in Section 4.2.1.2, AdapNet employs a simple decoder with two deconvolution layers and one skip refinement connection. Although the decoder is more effective in recovering the segmentation details in comparison to direct bilinear upsampling, it often produces disconnected segments while recovering the structure of thin objects such as poles and fences. In order to overcome this impediment, we propose a more effective decoder that we integrate into the AdapNet++ architecture.

The decoder shown in Figure 4.7 consists of three stages. In the first stage, the output of the eASPP is upsampled by a factor of two using a deconvolution layer to obtain a coarse

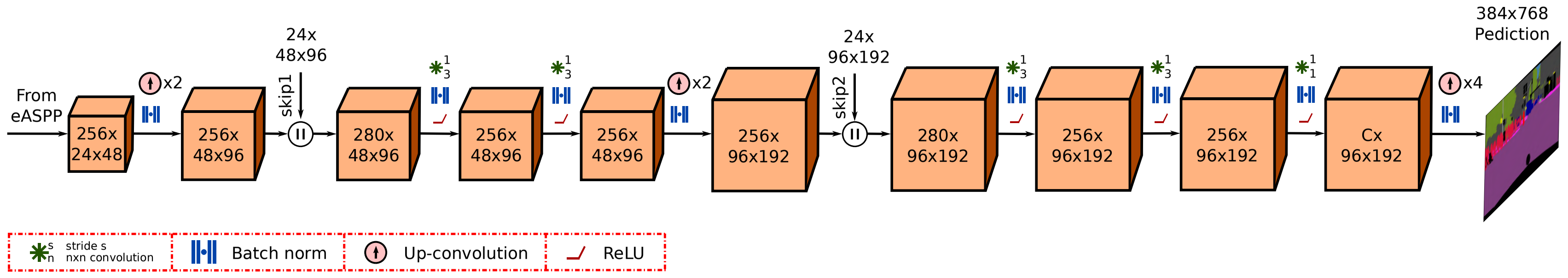


**Figure 4.7:** Our decoder consists of three upsampling stages that recover segmentation details using deconvolution layers and two skip refinement stages that fuse mid-level features from the encoder to improve the segmentation along object boundaries. Each skip refinement stage consists of concatenation of mid-level features with the upsampled decoder feature maps, followed by two $3 \times 3$ convolutions to improve the discriminability of the high-level features and the resolution of the refinement.

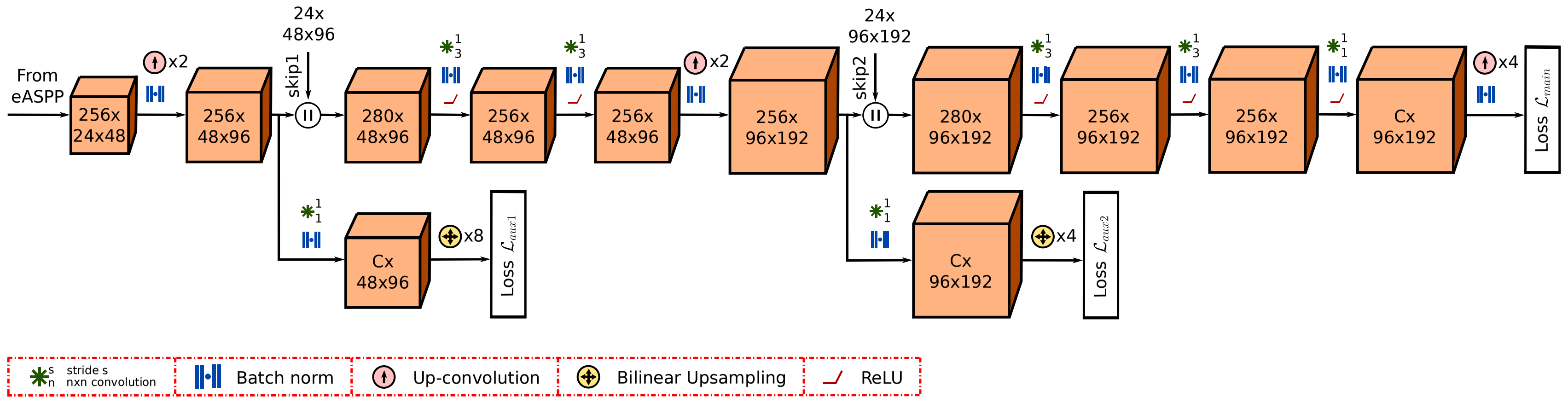


**Figure 4.8:** Depiction of the two auxiliary softmax losses that we add before each skip refinement stage in the decoder in addition to the main softmax loss in the end of the decoder. The two auxiliary losses are weighed for balancing the gradient flow through all the previous layers. While testing the auxiliary branches are removed and only the main stream as shown in Figure 4.7 is used.

segmentation mask. The upsampled coarse mask is then passed through the second stage, where the feature maps are concatenated with the first skip refinement from *Res3d*. The skip refinement consists of a $1 \times 1$ convolution layer to reduce the feature depth in order to not outweigh the encoder features. We experiment with varying number of feature channels in the skip refinement in the ablation study presented in Section 4.4.6.3. The concatenated feature maps are then passed through two $3 \times 3$ convolutions to improve the resolution of the refinement, followed by a deconvolution layer that again upsamples the feature maps by a factor of two. This upsampled output is fed to the last decoder stage which resembles the previous stage consisting of concatenation with the feature maps from the second skip refinement from *Res2c*, followed by two $3 \times 3$ convolution layers. All the convolutional and deconvolutional layers until this stage have 256 feature channels, therefore the output from the two $3 \times 3$ convolutions in the last stage is fed to a $1 \times 1$ convolution layer to reduce the number of feature channels to the number of object categories $C$. This output is finally fed to the last deconvolution layer which upsamples the feature maps by a factor of four to recover the original input resolution.

#### 4.2.2.3 Multiresolution Supervision

Deep networks often have difficulty in training due to the intrinsic instability associated with learning using gradient descent which leads to exploding or vanishing gradient problems. As our encoder is based on the residual learning framework, shortcut connections in each unit help propagating the gradient more effectively. Another technique that can be used to mitigate this problem to a certain extent is by initializing the layers with pretrained weights, however our proposed eASPP and decoder layers still have to be trained from scratch which could lead to optimization difficulties. Recent deep architectures have proposed employing an auxiliary loss in the middle of encoder network [135, 149], in addition to the main loss towards the end of the network. However, as shown in the ablation study presented in Section 4.4.6.1 this does not improve the performance of our network although it helps the optimization to converge faster.

Unlike previous approaches, we propose a multi-resolution supervision strategy to both accelerate the training and improve the resolution of the segmentation. As described in the previous section, our decoder consists of three upsampling stages. We add two auxiliary loss branches at the end of the first and second stage after the deconvolution layer in addition to the main $\mathsf{softmax}$ loss $\mathcal{L}_{main}$ at the end of the decoder as shown in Figure 4.8. Each auxiliary loss branch decreases the feature channels to the number of category labels $C$ using a $1 \times 1$ convolution with batch normalization and upsamples the feature maps to the input resolution using bilinear upsampling. We only use simple bilinear upsampling which does not contain any weights instead of a deconvolution layer in the auxiliary loss branches as our aim is to force the main decoder stream to improve its discriminativeness at each upsampling resolution so that it embeds multi-resolution information while learning to upsample. We weigh the two auxiliary losses $\mathcal{L}_{aux1}$ and $\mathcal{L}_{aux2}$ to balance the gradient flow through all the previous layers. While testing, the auxiliary loss branches are discarded and only the main decoder stream is used. We experiment with different loss weightings in the ablation study presented in Section 4.4.6.3, and we show that each of the auxiliary loss branches improves the segmentation performance in addition to speeding-up the training in Section 4.4.6.1.

### 4.2.3 Network Compression

As we strive to design an efficient and compact semantic segmentation architecture that can be employed in resource constrained applications, we must ensure that the utilization of convolutional filters in our network is thoroughly optimized. Often, even the most compact networks have abundant neurons in deeper layers that do not significantly contribute to the overall performance of the model. Excessive convolutional filters not only increase the model size but also the inference time and the number of computing operations. These factors critically hinder the deployment of models in resource constrained real-world applications. Pruning of neural networks can be traced back to the 80s when LeCun *et al.* [150] introduced a technique called Optimal Brain Damage for selectively pruning weights with a theoretically justified measure. Recently, several new techniques have been proposed for pruning weight matrices [141, 142, 151, 152] of convolutional layers as most of the computation during inference is consumed by them.

These approaches rank neurons based on their contribution and remove the low ranking neurons from the network, followed by fine-tuning of the pruned network. While the simplest neuron ranking method computes the $\ell^1$-norm of each convolutional filter [152], more sophisticated techniques have recently been proposed [140, 141, 142]. Some of these approaches are based on sparsity based regularization of network parameters which additionally increases the computational overhead during training [141, 151]. Techniques [142] have also been proposed for structured pruning of entire kernels with strided sparsity that demonstrate impressive results for pruning small networks. However, their applicability to complex networks that are to be evaluated on large validation sets has not been explored due its heavy computational processing. Moreover, until a year ago these techniques were only applied to simpler architectures such as VGG [153] and AlexNet [22], as pruning complex deep architectures such as ResNets requires a holistic approach. Thus far, pruning of residual units has only been performed on convolutional layers that do not have an identity or shortcut connection as pruning them additionally requires pruning the added residual maps in the exact same configuration. Attempts to prune them in the same configuration have resulted in a significant drop in performance [152]. Therefore, often only the first and the second convolutional layers of a residual unit are pruned.

Our proposed architectures have shortcut and skip connections both in the encoder as well the decoder. Therefore, in order to efficiently maximize the pruning of our network, we propose a holistic network-wide pruning technique that is invariant to the presence of skip or shortcut connections. Our proposed technique first involves pruning all the convolutional layers of a residual unit, followed by masking out the pruned indices of the last convolutional layer of a residual unit with zeros before the addition of the residual maps from the shortcut connection. As masking is performed after the pruning, we efficiently reduce the parameters and computing operations in a holistic fashion, while optimally pruning all the convolutional layers and preserving the shortcut or skip connections. After each pruning iteration, we fine-tune the network to recover any loss in accuracy. We illustrate this strategy adopting a recently proposed greedy criteria-based oracle pruning technique that incorporates a novel ranking method based on a first order Taylor expansion of the network cost function [140]. The pruning problem is framed as a combinatorial optimization problem such that when the weights $B$ of the network are pruned, the change in cost value will be minimal.

$$\min_{\mathcal{W}'} |\mathcal{C}(\mathcal{T}|\mathcal{W}') - \mathcal{C}(\mathcal{T}|\mathcal{W})| \quad \text{s.t.} \quad ||\mathcal{W}'||_0 \leq B, \tag{4.3}$$

where $\mathcal{T}$ is the training set, $\mathcal{W}$ is the network parameters and $\mathcal{C}(\cdot)$ is the negative log-likelihood function. Based on Taylor expansion, the change in the loss function from removing a specific parameter can be approximated. Let $h_i$ be the output feature maps produced by parameter $i$ and $h_i = \{z_0^1, z_0^2, \cdots, z_L^{C_l}\}$. The output $h_i$ can be pruned by setting it to zero and the ranking can be given by

$$|\Delta\mathcal{C}(h_i)| = |\mathcal{C}(\mathcal{T}, h_i = 0) - \mathcal{C}(\mathcal{T}, h_i)|, \tag{4.4}$$

Approximating with Taylor expansion, we can write

$$\begin{aligned} \Theta_{TE}(h_i) &= |\Delta\mathcal{C}(h_i)| = |\mathcal{C}(\mathcal{T}, h_i) - \frac{\delta\mathcal{C}}{\delta h_i} h_i - \mathcal{C}(\mathcal{T}, h_i)| \\ &= \left| \frac{\delta\mathcal{C}}{\delta h_i} h_i \right|, \qquad (4.5) \\ \Theta_{TE}(z_l^{(k)}) &= \left| \frac{1}{M} \sum_m \frac{\delta\mathcal{C}}{\delta z_{l,m}^{(k)}} z_{l,m}^{(k)} \right|, \qquad (4.6) \end{aligned}$$

where $M$ is the length of the vectorized feature map. This ranking can be easily computed using the standard back-propagation computation as it requires the gradient of the cost function with respect to the activation and the product of the activation. Furthermore, in order to achieve adequate rescaling across layers, a layer-wise $\ell^2$-norm of the rankings is computed as

$$\hat{\Theta}(z_l^{(k)}) = \frac{\Theta(z_l^{(k)})}{\sqrt{\sum_j \Theta^2(z_l^{(j)})}}. \tag{4.7}$$

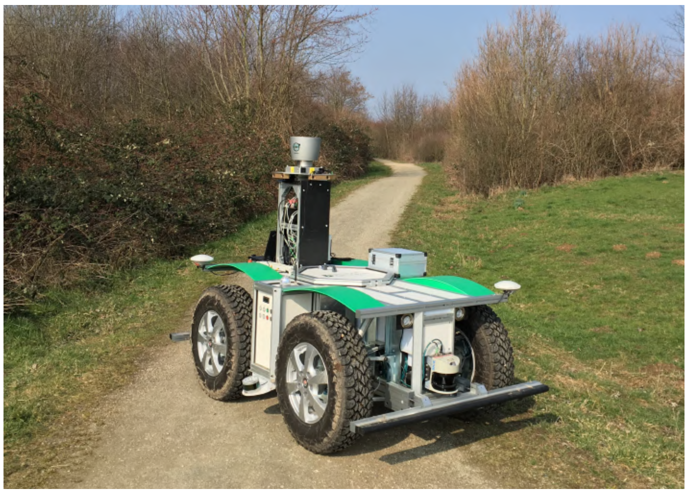
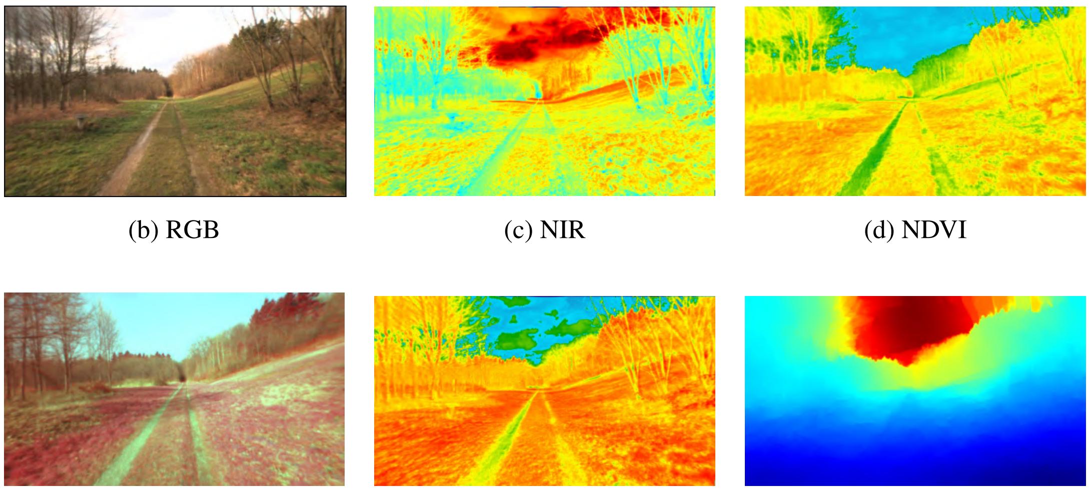


**Figure 4.9:** The Viona robot platform that we used for data collection equipped with a Bumblebee2 stereo vision camera and a self-made NIR camera. Both the cameras were time synchronized and the images were captured at 20 Hz.

The entire pruning procedure can be summarized as follows: first the network is trained until convergence using the training protocol described in Section 4.4.3. Then the importance of the feature maps is evaluated using the aforementioned ranking method and subsequently the unimportant feature maps are removed. The pruned convolution layers that have shortcut connections are then masked at the indices where the unimportant feature maps are removed to maintain the shortcut connections. The network is then fine-tuned and the pruning process is reiterated until the desired trade-off between accuracy and the number of parameters has been achieved. We present results from pruning our AdapNet++ architecture in Section 4.4.5, where we perform pruning of both the convolutional and deconvolutional layers of our network in five stages by varying the threshold for the rankings. For each of these stages, we quantitatively evaluate the performance versus number of parameters trade-off obtained using our proposed pruning strategy in comparison to the standard approach.

## 4.3 Freiburg Forest Dataset

We introduce the Freiburg Multispectral Segmentation benchmark, which is a first-of-a-kind semantic segmentation dataset of unstructured forested environments. Unlike urban and indoor scenes which are highly structured with rigid objects that have distinct geometric properties, objects in unstructured forested environments are extremely diverse and moreover, their appearance completely changes from month to month due to seasonal variations. The primary motivation for the introduction of this dataset is to enable robots to discern obstacles that can be driven over such as tall grass and bushes to obstacles



(b) RGB (c) NIR (d) NDVI

(e) NRG (f) EVI (g) Depth

**Figure 4.10:** Example images from our Freiburg Forest dataset showing the various spectra and modalities contained in our benchmark. NIR, NDVI, EVI and depth images are converted to a three channel colorized image by normalizing and applying the standard jet color map. NIR = Near-InfraRed. NDVI = Normalized Difference Vegetation Index. NRG = Near-InfraRed, Red-channel, Green-channel. EVI = Enhanced Vegetation Index.

that should be avoided such as tall trees and boulders. Therefore, we propose to exploit the presence of chlorophyll in these objects which can be detected in the Near-InfraRed (NIR) wavelength. NIR images provide a high fidelity description on the presence of vegetation in the scene and it enhances border accuracy for segmentation. In this chapter, we only use the RGB images from this dataset for segmentation. We use the NIR images for multimodal semantic segmentation presented in Chapter 5.

We collected the dataset over an extended period of time using our Viona autonomous mobile robot platform shown in Figure 4.9. The robot was equipped with a Bumblebee2 stereo vision camera and a modified camera with the NIR-cut filter replaced with a Wratten 25A filter for capturing the NIR wavelength in the blue and green channels. Both cameras are time synchronized and frames were captured at 20 Hz. In order to match the images captured by both cameras, we first compute SIFT [154] correspondences between the images using the Difference-of-Gaussian detector to provide similarity-invariance. We then filter the detected keypoints with the nearest neighbors test, followed by requiring consistency between the matches with respect to an affine transformation. The matches are further filtered using Random Sample Consensus (RANSAC) [155] and the transformation is estimated using the Moving Least Squares method by rendering through a mesh of triangles. We then transform the RGB image with respect to the NIR image and crop to the intersecting regions of interest. Although our implementation uses two cameras, it is the most cost-effective solution compared to commercial integrated multispectral cameras.

Figure 4.10 shows examples of each of the spectra and modalities that we captured.

We collected data on three different days to have enough variability in lighting conditions as shadows and sun angles play a crucial role in the quality of acquired images. Our raw dataset contains over $15,000$ images that were sub-sampled at 1 Hz, corresponding to traversing over 4.7 km each day. Our benchmark contains 366 images with pixel-level groundtruth annotations which were hand-annotated for six classes: *sky, trail, grass, vegetation, obstacle* and *void*. As there is an abundant presence of vegetation in our environment, we compute global-based vegetation indices such as Normalized Difference Vegetation Index (NDVI) and Enhanced Vegetation Index (EVI) to extract consistent spatial and global vegetation information. NDVI is resistant to noise caused due to changing sun angles, topography and shadows, but is susceptible to error due to variable atmospheric and canopy background conditions [156]. EVI was proposed to compensate for these defects with improved sensitivity to high biomass regions and improved detection though decoupling of canopy background signal and reduction in atmospheric influences. For all the images in our dataset, we calculate NDVI and EVI using the following equations.

$$NDVI = \frac{\rho_{nir} - \rho_{red}}{\rho_{nir} + \rho_{red}}, \tag{4.8}$$

where $\rho_{nir}$ is the reflectance at the NIR wavelength (0.7 – 1.1µm) and $\rho_{red}$ is the reflectance at the red wavelength (0.6 – 0.7µm).

$$EVI = G \times \frac{\rho_{nir} - \rho_{red}}{\rho_{nir} + (C_1 \times \rho_{red} - C_2 \times \rho_{blue}) + L}, \tag{4.9}$$

where $\rho_{blue}$ is the reflectance at the blue wavelength (0.45 – 0.52µm), $G$ is the gain factor, $L$ is a soil adjustment factor, $C_1$ and $C_2$ are coefficients used to correct for aerosol scattering in the red band by the use of the blue band.

Although our dataset contains images from the Bumblebee stereo camera, the processed disparity images were substantially noisy due to several factors including rectification artifacts and motion blur. We compared the results from semi-global matching [157] with the depth map obtained from a CNN-based depth prediction network and found that for an unstructured forested environment, the CNN-based approach provides denser and more reliable estimates. In this work, we use the approach from Liu *et al.* [158] that employs a deep convolutional neural field model for depth estimation by constructing unary and pairwise potentials of conditional random fields. We convert the depth, NIR, NDVI and EVI images to a three-channel colorized image by normalizing and applying the standard jet color map. We made the raw dataset and the semantic annotations publicly available at `http://deepscene.cs.uni-freiburg.de/#datasets`.

## 4.4 Experimental Evaluation

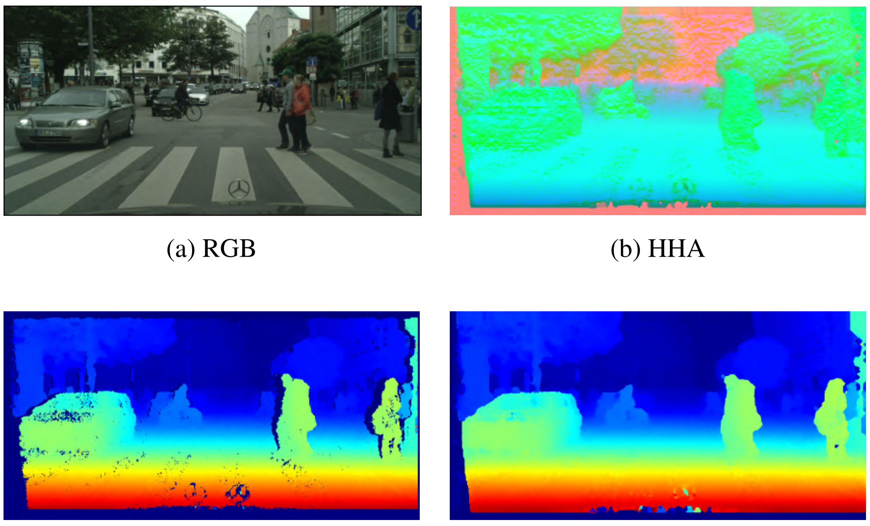
In this section, we first describe the datasets that we benchmark on, followed by comprehensive quantitative results for semantic segmentation using our proposed architectures in Section 4.4.1 and the results for model compression in Section 4.4.5. We then present detailed ablation studies that describe our architectural decisions in Section 4.4.6, followed by the qualitative segmentation results in Section 4.4.7.

All our models were implemented using the TensorFlow [159] deep learning library and the experiments were carried out on a system with an Intel Xeon E5, 2.4 GHz and an NVIDIA TITAN X GPU. We primarily use the standard Jaccard Index, also known as the intersection-over-union (IoU) metric to quantify the performance. It can be computed for each object class as $IoU = TP/(TP + FP + FN)$, where $TP, FP$ and $FN$ correspond to true positives, false positives and false negatives respectively. We also report the mean intersection-over-union (mIoU) metric for all the models and also the pixel-wise accuracy (Acc), average precision (AP), global intersection-over-union (gIoU) metric, false positive rate (FPR), false negative rate (FNR) in the detailed analysis.

### 4.4.1 Benchmark Datasets

We evaluate our proposed AdapNet and AdapNet++ architectures on five publicly available diverse scene understanding benchmarks ranging from urban driving scenarios to unstructured forested scenes and cluttered indoor environments. The datasets were particularly chosen based on the criteria of containing scenes with challenging perceptual conditions including rain, snow, fog, night-time, glare, motion blur and other seasonal appearance changes. Each of the datasets contain multiple modalities. For the experiments presented in this chapter, we only use the visual RGB images from these datasets. The other modalities and spectra are used for benchmarking the multimodal semantic segmentation presented in Chapter 5. In addition to our Freiburg Forest dataset, we benchmark on the following standard semantic scene segmentation datasets described in this section.

**Cityscapes:** The Cityscapes dataset [143] is one of the largest labeled RGB-D dataset for urban scene understanding. Being one of the standard benchmarks, it is highly challenging as it contains images of complex urban scenes, collected from over 50 cities during varying seasons, lighting and weather conditions. The images were captured using a automotive-grade 22 cm baseline stereo camera at a resolution of $2048 \times 1024$ pixels. The dataset contains 5,000 finely annotated images with 30 categories, of which 2,875 are provided for training, 500 are provided for validation and 1,525 are used for testing. Additionally 20,000 coarse annotations are provided. The testing images are not publicly released, they are used by the evaluation server for benchmarking, excluding the rarely appearing categories. In order to facilitate comparison with previous approaches, we benchmark on the label set



(a) RGB (b) HHA

(c) Depth (d) Depth Filled

**Figure 4.11:** Example image from the Cityscapes dataset showing a complex urban scene with many dynamic objects and the corresponding depth map representations.

consisting of: *sky, building, road, sidewalk, fence, vegetation, pole, car/truck/bus, traffic sign, person, rider/bicycle/motorbike* and *background*.

We convert the stereo disparity map to a three-channel colorized depth image by normalizing and applying the standard jet color map. Figures 4.11 (a) and (c) show an example image and the corresponding colorized depth map from the dataset. However, as seen in the figure, the depth maps have considerable amount of noise and missing depth values due to occlusion, which are undesirable especially when utilizing depth maps as an input modality for pixel-wise segmentation. Therefore, we employ a recently proposed state-of-the-art fast depth completion technique [160] to fill any holes that may be present. The resulting filled depth map is shown in Figure 4.11 (d). The depth completion algorithm can easily be incorporated into our pipeline as a preprocessing step as it only requires 11 ms while running on the CPU and it can be further parallelized using a GPU implementation.

Additionally, Gupta *et al.* [161] proposed an alternate representation of the depth map known as the HHA encoding to enable DCNNs to learn more effectively. The authors demonstrate that the HHA representation encodes properties of geocentric pose that emphasizes on complementary discontinuities in the image which are extremely hard for the network to learn, especially from limited training data. This representation also yields a three-channel image consisting of: horizontal disparity, height above ground, and the angle between the local surface normal of a pixel and the inferred gravity direction. The resulting channels are then linearly scaled and mapped to the 0 to 255 range. However, it is still unclear if this representation enables the network to learn features complementary to that learned from visual RGB images as different works show contradicting results

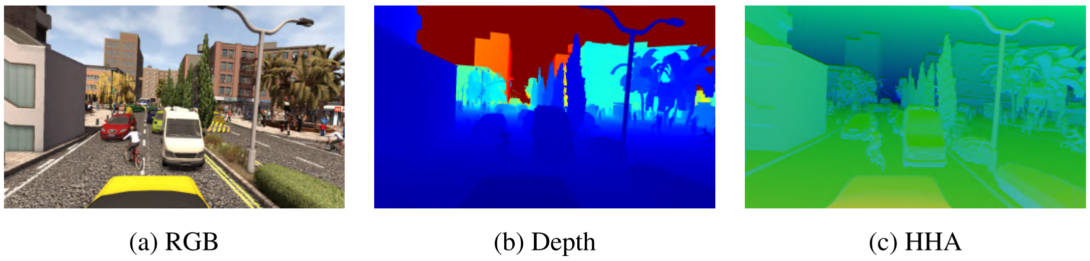


**Figure 4.12:** Example image from the Synthia dataset showing an outdoor urban scene and the corresponding depth map representations.

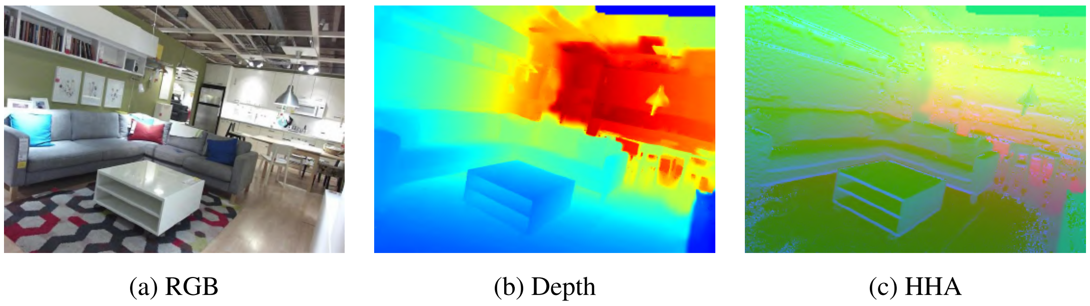


**Figure 4.13:** Example image from the SUN RGB-D dataset showing an indoor scene and the corresponding depth map representations.

[161, 162, 163]. In Chapter 5, we perform in-depth experiments with both the jet colorized and the HHA encoded depth map on a larger and more challenging dataset than previous works to investigate the utility of these encodings.

**Synthia:** The Synthia dataset [144] is a large-scale urban outdoor dataset that contains photo realistic images and depth data rendered from a virtual city built using the Unity engine. It consists of several annotated label sets. We use the Synthia-Rand-Cityscapes and the video sequences which have images of resolution $1280 \times 760$ with a $100^\circ$ horizontal field of view. This dataset is of particular interest for benchmarking multimodal semantic segmentation as it contains diverse traffic situations under different weather conditions. Synthia-Rand-Cityscapes consists of 9,000 images and the sequences contain 8000 images with groundtruth labels for 12 classes. The categories of object labels are the same as the aforementioned Cityscapes label set.

**SUN RGB-D:** The SUN RGB-D dataset [145] is one of the most challenging indoor scene understanding benchmarks to date. It contains 10,335 RGB-D images that were captured with four different types of RGB-D cameras (Kinect V1, Kinect V2, Xtion and RealSense) with different resolutions and fields of view. This benchmark also combines several other datasets including 1,449 images from the NYU Depth v2 [164], 554 images

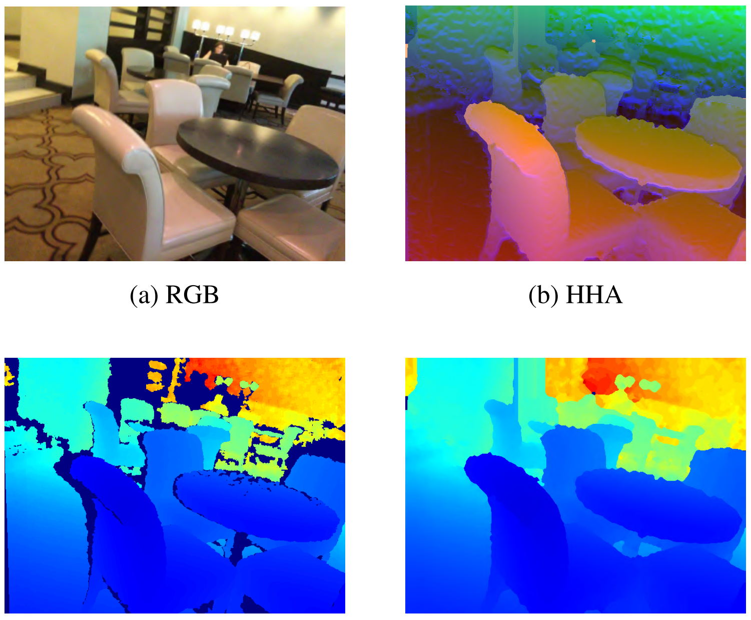
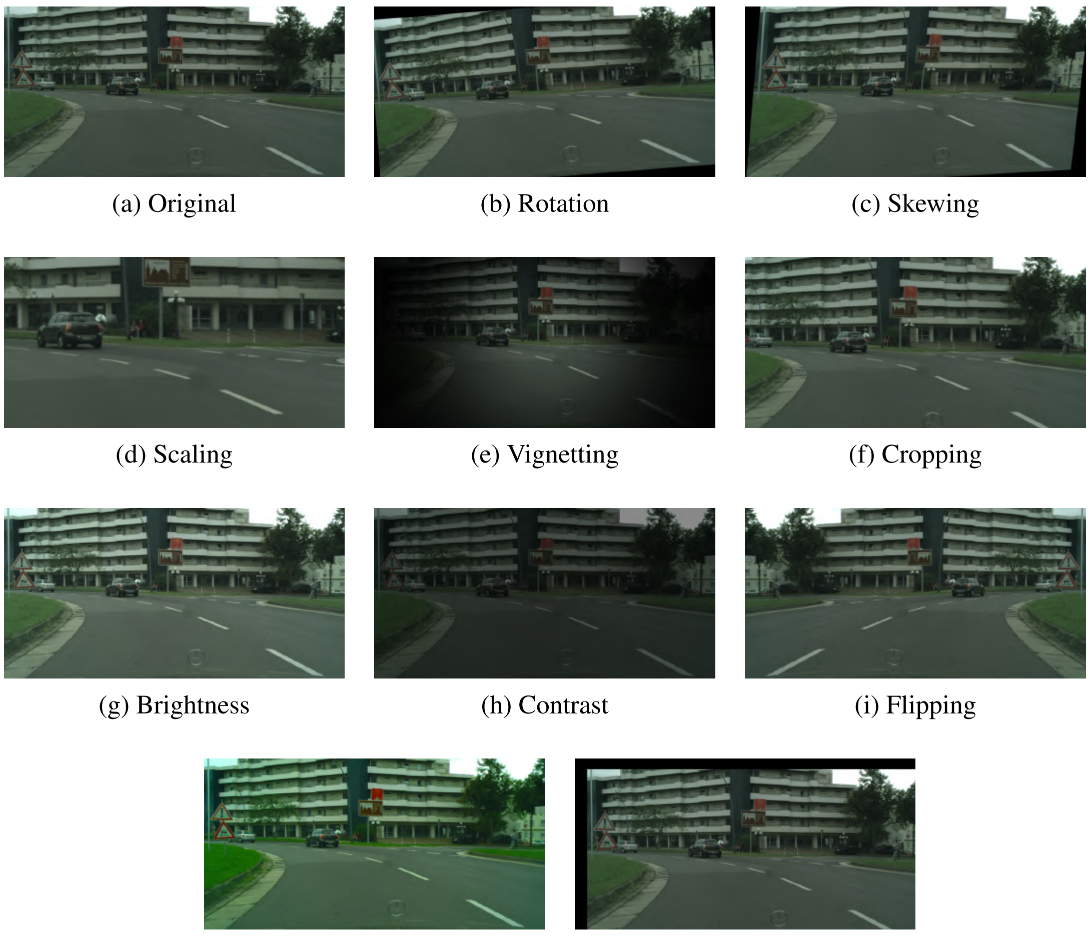


**Figure 4.14:** Example image from the ScanNet dataset showing a complex indoor scene and the corresponding depth map representations.

from the Berkeley B3DO [165] and 3,389 images from the SUN3D [166]. We use the original train-val split consisting of 5,285 images for training and 5,050 images for testing. We use the refined in-painted depth images from the dataset that were processed using a multi-view fusion technique. However, some refined depth images still have missing depth values at distances larger than a few meters. Therefore, as mentioned in previous works [162], we exclude the 587 training images that were captured using the RealSense RGB-D camera as they contain a significant amount of invalid depth measurements that are further intensified due to the in-painting process.

This dataset provides pixel-level semantic annotations for 37 categories, namely: *wall, floor, cabinet, bed, chair, sofa, table, door, window, bookshelf, picture, counter, blinds, desk, shelves, curtain, dresser, pillow, mirror, floor mat, clothes, ceiling, books, fridge, tv, paper, towel, shower curtain, box, whiteboard, person, night stand, toilet, sink, lamp, bathtub* and *bag*. Although we benchmark on all the object categories, 16 out of the 37 classes are rarely present in the images and about 0.25% of the pixels are not assigned to any of the classes, making it extremely unbalanced. Moreover, as each scene contains many different types of objects, they are often partially occluded and may appear completely different in the test images.

**ScanNet:** The ScanNet RGB-D video dataset [146] is a recently introduced large-scale indoor scene understanding benchmark. It contains 2.5 M RGB-D images accounting to



(a) Original (b) Rotation (c) Skewing

(d) Scaling (e) Vignetting (f) Cropping

(g) Brightness (h) Contrast (i) Flipping

(j) Color (k) Translate

**Figure 4.15:** Example data augmentation strategies that we employ on the images randomly while training to increase the amount of training data and to improve the robustness of the network to overfitting.

1512 scans acquired in 707 distinct spaces. The data was collected using an iPad Air2 mounted with a depth camera similar to the Microsoft Kinect v1. Both the iPad camera and the depth camera were hardware synchronized and frames were captured at 30 Hz. The RGB images were captured at a resolution of $1296 \times 968$ pixels and the depth frames were captured at $640 \times 480$ pixels. The semantic segmentation benchmark contains 16,506 labelled training images and 2537 testing images. From the example depth image shown in Figure 4.14 (b), we can see that there are a number of missing depth values at the object boundaries and at large distances. Therefore, similar to the preprocessing that we perform on the cityscapes dataset, we use a fast depth completion technique [160] to fill the holes. The corresponding filled depth image is shown in Figure 4.14 (c). We also compute the HHA encoding for all the depth maps as shown in Figure 4.14 (b).

The dataset provides pixel-level semantic annotations for 21 object categories, namely:

*wall, floor, chair, table, desk, bed, bookshelf, sofa, sink, bathtub, toilet, curtain, counter, door, window, shower curtain, refrigerator, picture, cabinet, other furniture* and *void*. Similar to the SUN RGB-D dataset, many object classes are rarely present making the dataset unbalanced. Moreover, the annotations at the object boundaries are often irregular and parts of objects at large distances are unlabelled. These factors make the task even more challenging on this dataset.

### 4.4.2 Data Augmentation

Training of deep networks can be significantly improved by expanding the dataset to introduce more variability in the captured scene. The core idea is to apply a set of data augmentations on the training data, where the applied transformations define the invariance properties that are to be learned by the network. This enables the network to substantially improve its robustness to overfitting and also its generalization ability to different real-world scenarios. In order to achieve this, we apply a series of augmentation strategies randomly on the input data while training. The augmentations [6] that we apply include rotation (–13° to 13°), skewing (0.05 to 0.10), scaling (0.5 to 2.0), vignetting (210 to 300), cropping (0.8 to 0.9), brightness modulation (–40 to 40), contrast modulation (0.5 to 1.5) and left-right flipping.

### 4.4.3 Network Training

We represent the training set for semantic segmentation as $\mathcal{T} = \{(I_n, M_n) \mid n = 1, \ldots, N\}$, where $I_n = \{u_r \mid r = 1, \ldots, \rho\}$ denotes the input image and the corresponding ground truth label mask $M_n = \{m_r^n \mid r = 1, \ldots, \rho\}$, where $m_r^n \in \{1, \ldots, C\}$ is the set of semantic classes. We define $\theta$ as the network parameters consisting of weights and biases, and $s_j(u_r, \theta)$ as the score assigned for labeling pixel $u_r$ with label $j$. We obtain the probabilities $\mathbf{P} = (p_1, \ldots, p_C)$ for all the semantic classes using the $\mathsf{softmax}$ function $\sigma(.)$ as

$$p_j(u_r, \theta \mid I_n) = \sigma\left(s_j\left(u_r, \theta\right)\right) = \frac{exp\left(s_j\left(u_r, \theta\right)\right)}{\sum_k^C exp\left(s_k\left(u_r, \theta\right)\right)}. \tag{4.10}$$

The optimal network parameters are then estimated by minimizing the cross-entropy loss function as

$$\mathcal{L}_{seg}(\mathcal{T}, \theta) = -\sum_{n=1}^{N}\sum_{r=1}^{\rho}\sum_{j=1}^{C} \delta_{m_r^n, j} \log p_j(u_r, \theta \mid I_n), \tag{4.11}$$

for $(I_n, M_n) \in \mathcal{T}$, where $\delta_{m_r^n, j}$ is the Kronecker delta.

We train our network with an input image of resolution $768 \times 384$ pixels, therefore, we use bilinear interpolation for resizing the RGB images and the nearest-neighbor interpolation for resizing the groundtruth labels. We initialize the encoder of the network with weights pre-trained on the ImageNet dataset [167], while we use the He initialization [93] scheme for the other convolutional and deconvolutional layers. We use the Adam solver for optimization with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-10}$. We train our model for a maximum of 150,000 iterations using an initial learning rate of $\lambda_0 = 10^{-3}$ with a mini-batch size of 8 and a dropout probability of 0.5. We set the weights $\lambda_1 = 0.6$ and $\lambda_2 = 0.5$ to balance the auxiliary losses. The final loss function is computed as $\mathcal{L} = \mathcal{L}_{main} + \lambda_1\mathcal{L}_{aux1} + \lambda_2\mathcal{L}_{aux2}$.

### 4.4.4 Comparison with the State-of-the-Art

In this section, we report results comparing the performance of our proposed AdapNet and AdapNet++ architectures for a input image of resolution $768 \times 384$ pixels. We benchmark against several well adopted state-of-the-art models including DeepLab v3 [132], ParseNet [168], FCN-8s [24], SegNet [136], FastNet [48], DeepLab v2 [138] and DeconvNet [169]. For each of the datasets, we report the mIoU score, as well as the per-class IoU score. Note that we report the performance of each model for the same input image resolution of $768 \times 384$ pixels and using the same evaluation setting in order to have a fair comparison. We do not apply multiscale inputs or left-right flips during testing as these techniques require each crop of each image to be evaluated several times which significantly increases the computational complexity and runtime (Note: We do not use crops for testing, we evaluate on the full image in a single forward-pass). Moreover, these techniques do not improve the performance of the model in real-time applications. However, we show the potential gains that can be obtained in the evaluation metric utilizing these techniques and with a higher resolution input image in the ablation study presented in Section 4.4.6.5.

Table 4.1 shows the comparison on the 11 class Cityscapes validation set. Both AdapNet and AdapNet++ outperform all the baselines in each individual object category as well in the mIoU score. AdapNet and AdapNet++ outperform the highest baseline by a margin of 2.35% and 5.59% respectively. Analyzing the individual class IoU scores, we can see that AdapNet++ yields the highest improvement over AdapNet in object categories that contain thin structures such as *poles* for which it gives a large improvement of 5.42%, a similar improvement of 5.05% for *fences* and the highest improvement for 7.29% for *signs*. Most architectures struggle to recover the structure of thin objects due to downsampling by pooling and striding in the network which causes such information to be lost. However, these results show that AdapNet++ efficiently recovers the structure of such objects by learning multiscale features at several stages of the encoder using the proposed multiscale residual units and the eASPP. We further show the improvement in performance due to the incorporation of the multiscale residual units and the eASPP in the ablation study presented in Section 4.4.6.1. In driving scenarios, information of objects such as *pedestrians* and *cyclists* can also be lost when they appear at far away distances. A large improvement can also be seen in categories such as *person* in which AdapNet++ achieves an improvement

**Table 4.1:** Performance comparison of AdapNet and AdapNet++ with baseline models on the Cityscapes dataset (11 classes).

| Network | Sky | Building | Road | Sidewalk | Fence | Veg | Pole | Car | Sign | Person | Cyclist | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | ⬛ | 🟪 | 🟪 | 🟫 | 🟩 | ⬜ | 🟦 | 🟨 | 🟥 | ⬛ | (%) |
| FCN-8s [24] | 76.51 | 83.97 | 93.82 | 67.67 | 24.91 | 86.38 | 31.71 | 84.80 | 50.92 | 59.89 | 59.11 | 59.97 |
| SegNet [136] | 73.74 | 79.29 | 92.70 | 59.88 | 13.63 | 81.89 | 26.18 | 78.83 | 31.44 | 45.03 | 43.46 | 52.17 |
| FastNet [48] | 77.69 | 86.25 | 94.97 | 72.99 | 31.02 | 88.06 | 38.34 | 88.42 | 52.34 | 61.76 | 61.83 | 68.52 |
| ParseNet [168] | 77.57 | 86.81 | 95.27 | 74.02 | 33.31 | 87.37 | 38.24 | 88.99 | 53.34 | 63.25 | 63.87 | 69.28 |
| DeconvNet [169] | 89.38 | 83.08 | 95.26 | 68.07 | 27.58 | 85.80 | 34.20 | 85.01 | 27.62 | 45.11 | 41.11 | 62.02 |
| DeepLab v2 [138] | 74.28 | 81.66 | 90.86 | 63.3 | 26.29 | 84.33 | 27.96 | 86.24 | 44.79 | 58.89 | 60.92 | 63.59 |
| DeepLab v3 [132] | 92.40 | 89.02 | 96.74 | 78.55 | 41.00 | 90.81 | 49.74 | 91.02 | 64.48 | 66.52 | 66.98 | 75.21 |
| AdapNet (Ours) | 92.45 | 89.98 | 97.43 | 81.43 | 49.93 | 91.44 | 53.43 | 92.23 | 65.32 | 69.86 | 69.62 | 77.56 |
| AdapNet++ (Ours) | **94.18** | **91.49** | **97.93** | **84.40** | **54.98** | **92.09** | **58.85** | **93.86** | **72.61** | **75.52** | **72.90** | **80.80** |

**Table 4.2:** Performance comparison of AdapNet and AdapNet++ with baseline models on the Synthia dataset.

| Network | Sky | Building | Road | Sidewalk | Fence | Veg | Pole | Car | Sign | Person | Cyclist | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ⬜ | 🟥 | 🟪 | 🟦 | 🟦 | 🟩 | 🟨 | 🟪 | 🟫 | ⬛ | 🟦 | (%) |
| FCN-8s [24] | 92.36 | 91.92 | 88.94 | 86.46 | 48.22 | 77.41 | 36.02 | 82.63 | 30.37 | 57.10 | 46.84 | 67.11 |
| SegNet [136] | 91.90 | 87.19 | 83.72 | 80.94 | 50.02 | 71.63 | 26.12 | 71.31 | 1.01 | 52.34 | 32.64 | 58.98 |
| FastNet [48] | 92.21 | 92.41 | 91.85 | 89.89 | 56.64 | 78.59 | 51.17 | 84.75 | 32.03 | 69.87 | 55.65 | 72.28 |
| ParseNet [168] | 93.80 | 93.09 | 91.05 | 88.98 | 53.22 | 79.48 | 46.15 | 85.37 | 36.00 | 63.30 | 50.82 | 71.02 |
| DeconvNet [169] | 95.88 | 93.83 | 92.85 | 90.79 | 66.40 | 81.04 | 48.23 | 84.65 | 0.00 | 69.46 | 52.79 | 70.54 |
| DeepLab v2 [138] | 94.07 | 93.34 | 88.07 | 88.93 | 55.57 | 80.22 | 45.97 | 85.87 | 38.73 | 64.40 | 52.54 | 71.61 |
| DeepLab v3 [132] | 95.30 | 92.75 | 93.58 | 91.56 | 73.37 | 80.71 | 55.83 | 88.09 | 44.17 | 75.65 | 60.15 | 77.38 |
| AdapNet (Ours) | 96.95 | 95.88 | 95.60 | 94.46 | 76.30 | 86.59 | 67.14 | 92.20 | 58.85 | 80.18 | 66.89 | 82.83 |
| AdapNet++ (Ours) | **97.77** | **96.98** | **96.60** | **95.70** | **79.87** | **89.63** | **74.94** | **94.22** | **71.99** | **83.64** | **72.31** | **86.70** |

**Table 4.3:** Benchmarking results on the Cityscapes dataset with full resolution evaluation on 19 semantic class labels. Only the eight top performing published models in the leaderboard are listed in this table. The inference time is reported for an input image resolution of 768 × 384 pixels and it was computed on an NVIDIA TITAN X (PASCAL) GPU using the official implementation of each method. SSMA refers to our multimodal segmentation architecture that builds upon AdapNet++ and is detailed in Chapter 5.

| Network | Backbone | mIoU (%) | | Parms. | Time |
|---|---|---|---|---|---|
| | | val | test | (M) | (ms) |
| PSPNet [135] | ResNet-101 | 80.91 | 81.19 | 56.27 | 172.42 |
| DeepLab v3 [132] | ResNet-101 | 79.30 | 81.34 | 58.16 | 79.90 |
| Mapillary [170] | WideResNet-38 | 78.31 | 82.03 | 135.86 | 214.46 |
| DeepLab v3+ [171] | Modified Xception | 79.55 | 82.14 | 43.48 | 127.97 |
| DPC [172] | Modified Xception | 80.85 | 82.66 | 41.82 | 144.41 |
| DRN [173] | WideResNet-38 | 79.69 | 82.82 | 129.16 | 1259.67 |
| AdapNet++ (Ours) | ResNet50 | 81.24 | 81.34 | 30.20 | 72.92 |
| SSMA (Ours) | ResNet50 | 82.19 | 82.31 | 56.44 | 101.95 |

of 5.66%. The improvement in larger object categories such as *cars* and *vegetation* can be attributed to the new decoder which improves the segmentation performance near object boundaries. This is more evident in the qualitative results presented in Section 4.4.7. Note that the colors shown below the object category names serve as a legend for the qualitative results.

We also report results on the full 19 class Cityscapes validation and test sets in Table 4.3. We compare against the top six published models on the leaderboard, namely, PSPNet [135], DeepLab v3 [132], Mapilary [170], DeepLab v3+ [171], DPC [172], and DRN [173]. The results of the competing methods reported in this table are directly taken from the benchmark leaderboard for the test set and from the corresponding manuscripts of the methods for the validation set. We trained our models on 768 × 768 crops from the full image resolution for benchmarking on the leaderboard. Our AdapNet++ model with a much smaller network backbone achieves a comparable performance as other top performing models on the leaderboard. Moreover, our network is the most efficient architecture in terms of both the number of parameters that it consumes as well as the inference time compared to other networks on the entire first page of the Cityscapes leaderboard.

We benchmark on the Synthia dataset largely due to the variety of seasons and adverse perceptual conditions which make the task of semantic segmentation extremely challenging. From the results shown in Table 4.2, it can be seen that both AdapNet and AdapNet++ outperform all the baselines in the overall mIoU score as well as in the score of the individual object categories. AdapNet and AdapNet++ achieve an overall substantial improvement of 5.45% and 9.32% respectively. A similar observation can be made in the

**Table 4.4:** Performance comparison of AdapNet and AdapNet++ with baseline models on the SUN RGB-D dataset.

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Blinds | Desk | Shlves | Crtain | Dresr | Pillow | Mirror |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟦 | 🟧 | 🟧 | 🟩 | 🟩 | 🟥 | 🟥 | 🟪 | 🟪 | 🟫 | 🟦 | 🟪 | 🟪 | ⬛ | ⬜ | 🟨 | 🟨 | 🟦 |
| FCN-8s [24] | 68.57 | 79.94 | 37.27 | 52.85 | 58.42 | 42.79 | 43.69 | 27.58 | 43.49 | 28.46 | 42.21 | 27.08 | 28.34 | 11.38 | 6.01 | 46.96 | 28.80 | 27.73 | 18.97 |
| SegNet [136] | 70.18 | 82.01 | 37.62 | 45.77 | 57.22 | 35.86 | 40.60 | 30.32 | 39.37 | 29.67 | 40.67 | 22.48 | 32.13 | 14.79 | 5.15 | 39.69 | 27.46 | 26.05 | 15.64 |
| FastNet [48] | 68.39 | 79.73 | 34.98 | 44.93 | 55.26 | 36.68 | 38.57 | 26.88 | 38.28 | 26.02 | 39.72 | 20.55 | 22.47 | 11.71 | 5.67 | 41.98 | 25.08 | 22.76 | 11.76 |
| ParseNet [168] | 70.83 | 81.85 | 40.25 | 55.64 | 61.12 | 45.06 | 45.86 | 30.41 | 41.73 | **34.45** | 44.11 | 28.63 | 32.59 | 12.78 | **8.58** | 46.51 | 34.05 | 28.07 | 24.82 |
| DeconvNet [169] | 61.53 | 75.52 | 26.97 | 38.52 | 48.46 | 30.94 | 37.12 | 21.99 | 27.48 | 0.00 | 0.00 | 0.00 | 0.00 | 9.40 | 0.00 | 29.59 | 0.00 | 0.00 | 8.32 |
| DeepLab v2 [138] | 63.68 | 74.07 | 31.15 | 47.00 | 55.46 | 40.81 | 40.60 | 29.10 | 37.95 | 18.67 | 35.67 | 23.20 | 25.08 | 13.52 | 2.55 | 42.91 | 24.56 | 29.81 | **27.16** |
| DeepLab v3 [132] | **74.99** | 85.44 | 39.64 | 54.14 | 64.23 | 45.84 | 47.35 | 38.30 | 45.50 | 30.70 | 45.30 | 15.21 | 24.92 | 18.18 | 66.37 | 48.84 | 28.20 | 32.87 | 31.23 |
| AdapNet (Ours) | 72.82 | **86.61** | 41.90 | 56.63 | 64.99 | 46.14 | 46.74 | 36.14 | 44.43 | 28.40 | **46.53** | 25.17 | 31.07 | 13.41 | 3.98 | 47.90 | 28.58 | **32.90** | 19.25 |
| AdapNet++ (Ours) | 73.81 | 84.79 | **47.45** | **64.31** | **65.76** | **52.15** | **49.97** | **41.66** | **46.99** | 32.83 | 45.19 | **32.71** | **34.72** | **21.16** | 5.03 | **50.32** | **37.45** | 32.35 | 25.89 |

| Network | FMat | Clths | Ceil | Books | Fridge | Tv | Paper | Towel | ShwrC | Box | Wbrd | Person | NStnd | Toilet | Sink | Lamp | Bthtub | Bag | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟩 | 🟨 | 🟪 | 🟥 | 🟦 | 🟧 | 🟩 | 🟪 | ⬜ | 🟪 | 🟩 | 🟨 | 🟥 | 🟦 | 🟩 | 🟪 | 🟧 | (%) |
| FCN-8s [24] | 0.00 | 20.34 | 51.98 | 30.98 | 9.06 | **34.23** | 15.95 | 12.79 | 0.00 | 18.24 | 41.38 | 8.92 | 0.97 | 59.64 | 44.21 | 22.83 | 26.20 | 8.81 | 30.46 |
| SegNet [136] | 0.00 | 16.01 | 53.89 | 26.21 | 12.95 | 24.23 | 14.57 | 10.88 | 0.00 | 13.50 | 38.02 | 3.83 | 0.76 | 60.98 | 45.97 | 22.82 | 31.94 | 8.97 | 29.14 |
| FastNet [48] | 0.10 | 13.37 | 49.14 | 29.38 | 17.49 | 19.80 | 17.95 | 13.55 | 2.93 | 15.33 | 38.12 | 15.82 | 7.93 | 54.08 | 39.59 | 22.20 | 23.71 | 9.80 | 28.15 |
| ParseNet [168] | 0.02 | 19.10 | 52.28 | **34.36** | 27.05 | 31.49 | **21.65** | 21.49 | **6.46** | 20.77 | 44.48 | 34.59 | 15.37 | 66.96 | 47.15 | 31.49 | 28.14 | 13.19 | 34.68 |
| DeconvNet [169] | 0.00 | 0.00 | 46.40 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 4.43 | 32.06 | 0.00 | 0.00 | 47.35 | 32.85 | 0.00 | 0.00 | 0.00 | 15.64 |
| DeepLab v2 [138] | 0.00 | 15.85 | 43.47 | 25.39 | 35.55 | 23.21 | 16.53 | 22.00 | 0.34 | 13.56 | 22.32 | **46.18** | 11.45 | 55.94 | 43.33 | 29.52 | 32.73 | 7.63 | 30.06 |
| DeepLab v3 [132] | 0.00 | 16.98 | 60.09 | 31.74 | 37.86 | 33.80 | 17.84 | 20.51 | 0.00 | 19.58 | 44.10 | 31.36 | 18.15 | 68.76 | 52.01 | 35.36 | **45.97** | 10.37 | 35.74 |
| AdapNet (Ours) | 0.00 | 17.30 | 52.60 | 30.63 | 20.32 | 16.56 | 18.21 | 13.69 | 0.00 | 15.72 | 43.02 | 17.48 | 9.44 | 61.11 | 49.43 | 31.32 | 21.56 | 9.38 | 32.47 |
| AdapNet++ (Ours) | **0.12** | **21.68** | **60.65** | 32.41 | **46.96** | 18.81 | 20.79 | **27.95** | 1.32 | **22.70** | **51.08** | 33.77 | **20.59** | **71.28** | **55.31** | **36.74** | 38.60 | **15.34** | **38.40** |

improvement of scores for thin structures, reinforcing the utility of our proposed multiscale feature learning configuration. AdapNet++ improves over AdapNet by 13.14% for the *sign* class, followed by an improvement of 7.8% for the *pole* class. In addition a significant improvement of 5.42% can also be seen for the *cyclist* class.

Compared to outdoor driving datasets, indoor benchmarks such as SUN RGB-D and ScanNet pose a different challenge. Indoor datasets have vast amounts of object categories in various different configurations and images captured from many different view points compared to driving scenarios where the camera is always parallel to the ground with similar viewpoints from the perspective of the vehicle driving on the road. Moreover, indoor scenes are often extremely cluttered which causes occlusions, in addition to the irregular frequency distribution of the object classes that make the problem even harder. Due to these factors SUN RGB-D is considered one of the hardest datasets to benchmark on. Despite these factors, as shown in Table 4.4, AdapNet++ outperforms all the baseline networks overall by a margin of 2.66% compared to the highest performing DeepLab v3 baseline which took 30,000 iterations more to reach this score. Unlike the performance in the Cityscapes and Synthia datasets where our AdapNet architecture yields the second highest performance, AdapNet is outperformed by DeepLab v3 in the SUN RGB-D dataset. AdapNet++ on the other hand, outperforms the baselines in most categories by a large margin, while it is outperformed in 13 of the 37 classes by small margin. It can also be observed that the classes in which AdapNet++ get outperformed are the most infrequent classes. This can be alleviated by adding supplementary training images containing the low-frequency classes from other datasets or by employing class balancing techniques. However, our initial experiments employing techniques such as median frequency class balancing, inverse median frequency class balancing, normalized inverse frequency balancing, severely affected the performance of our model.

We also show results on the validation set of the recently introduced ScanNet dataset in Table 4.5, which is currently the largest labeled indoor RGB-D dataset till date. AdapNet++ outperforms the state-of-the-art overall by a margin of 2.61%. The large improvement can be attributed to the proposed eASPP which efficiently captures long range context. Context aggregation plays an important role in such cluttered indoor datasets as different parts of an object are occluded from different viewpoints and across scenes. As objects such as the legs of a chair have thin structures, multiscale learning contributes to recovering such structures. We see a similar trend in the performance as in the SUN RGB-D dataset, where our network outperforms the baselines in most of the object categories (11 of the 20 classes) significantly, while yielding a comparable performance for the other categories. The largest improvement of 11.62% is obtained for the *counter* class, followed by an improvement of 8.69% for the *curtain* class which appears as many different variations in the dataset. An interesting observation that can be made is that the highest parametrized network DeconvNet which has 252 M parameters has the lowest performance in both SUN RGB-D and ScanNet datasets, while AdapNet++ which has about 1/9th of the parameters,

**Table 4.5:** Performance comparison of AdapNet and AdapNet++ with baseline models on the ScanNet dataset.

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Desk | Curtain | Fridge | ShwrC | Toilet | Sink | Bthtub | OthrF | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | ■ | (%) |
| FCN-8s [24] | 70.00 | 74.70 | 39.31 | 66.23 | 51.96 | 48.21 | 45.19 | 42.61 | 42.15 | 62.59 | 27.66 | 24.69 | 33.25 | 48.29 | 51.96 | 32.15 | 55.67 | 35.80 | 38.27 | 26.72 | 45.87 |
| SegNet [136] | 63.45 | 62.67 | 28.90 | 40.62 | 37.97 | 37.37 | 24.73 | 28.09 | 35.27 | 40.17 | 14.52 | 17.75 | 16.42 | 36.69 | 0.00 | 1.16 | 31.01 | 14.98 | 26.32 | 17.47 | 28.78 |
| FastNet [48] | 66.31 | 71.35 | 34.22 | 56.32 | 47.86 | 44.87 | 39.34 | 33.20 | 34.32 | 50.11 | 23.27 | 24.48 | 28.87 | 42.56 | 37.77 | 20.27 | 49.55 | 27.27 | 31.99 | 24.39 | 39.42 |
| ParseNet [168] | 71.29 | 76.42 | 39.92 | 71.11 | 52.72 | 51.07 | 48.74 | 43.06 | 45.25 | 58.81 | 31.33 | 27.84 | 38.46 | 54.81 | 52.40 | 28.60 | **58.53** | **36.27** | **41.91** | 25.85 | 47.72 |
| DeconvNet [169] | 69.18 | 74.92 | 26.78 | 51.92 | 43.75 | 41.59 | 42.86 | 24.89 | 0.00 | 53.64 | 0.00 | 14.23 | 18.79 | 36.77 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 11.46 | 25.54 |
| DeepLab v2 [138] | 58.88 | 73.27 | 39.76 | 61.38 | 53.27 | 57.13 | 47.84 | 38.90 | 37.77 | 36.49 | 25.31 | 29.19 | 29.47 | 43.41 | **64.91** | 32.89 | 53.69 | 30.78 | 37.03 | 24.36 | 43.75 |
| DeepLab v3 [132] | 71.90 | **77.99** | **49.00** | 74.51 | 58.79 | 61.54 | 50.04 | 47.08 | 43.35 | **63.34** | 32.22 | 33.57 | 36.44 | 51.78 | 64.41 | 37.15 | 55.01 | 29.18 | 38.08 | 26.02 | 50.07 |
| AdapNet (Ours) | 68.73 | 77.07 | 44.45 | 59.44 | 53.49 | 54.53 | 47.26 | 45.22 | 48.83 | 55.57 | **39.18** | 30.49 | 36.73 | 55.61 | 57.53 | 19.41 | 52.84 | 35.92 | 37.00 | **29.26** | 47.43 |
| AdapNet++ (Ours) | **74.24** | 71.95 | 48.78 | **76.06** | **62.72** | **63.55** | **56.71** | **52.52** | **53.41** | 52.29 | 38.43 | **45.19** | **41.43** | **64.30** | 60.91 | **40.64** | 56.86 | 30.62 | 38.60 | 24.34 | **52.68** |

**Table 4.6:** Performance comparison of AdapNet and AdapNet++ with baseline models on the Freiburg Forest dataset.

| Network | Trail | Grass | Veg. | Sky | Obst. | mIoU | gIoU | FPR | FNR | Param | Time |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | ■ | ■ | ■ | ■ | ■ | (%) | (%) | (%) | (%) | (M) | (ms) |
| FCN-8s [24] | 82.60 | 85.69 | 88.78 | 89.97 | 40.40 | 77.49 | 86.64 | 6.21 | 7.15 | 134.0 | 101.99 |
| SegNet [136] | 82.12 | 84.99 | 88.64 | 89.90 | 27.23 | 74.58 | 86.24 | 6.77 | 6.98 | 29.4 | 79.13 |
| FastNet [48] | 85.70 | 87.53 | 90.23 | 90.73 | 43.76 | 79.67 | 88.18 | 6.09 | 5.72 | 21.0 | 49.31 |
| ParseNet [168] | 85.67 | 87.33 | 89.63 | 89.17 | 43.08 | 78.97 | 87.65 | 6.34 | 6.01 | 20.5 | 286.54 |
| DeconvNet [169] | 86.37 | 87.17 | 89.63 | 92.20 | 34.80 | 78.04 | 89.06 | 6.53 | 6.78 | 252.0 | 168.54 |
| DeepLab v2 [138] | 88.75 | 88.87 | 90.29 | 91.65 | 49.55 | 81.82 | 89.98 | 5.94 | 5.79 | 43.7 | 128.90 |
| DeepLab v3 [132] | 88.62 | 88.84 | 90.55 | 91.75 | 51.61 | 82.28 | 90.08 | 5.21 | 5.82 | 58.16 | 82.83 |
| AdapNet (Ours) | 88.99 | 88.99 | 91.18 | 92.89 | 48.80 | 82.17 | 90.67 | 4.89 | 5.42 | 24.4 | 61.81 |
| AdapNet++ (Ours) | **89.27** | **89.41** | **91.43** | **93.01** | **52.32** | **83.09** | **90.75** | **4.84** | **5.37** | **28.1** | **72.77** |

**Table 4.7:** Bechmarking results on the ScanNet test set with full resolution evaluation. SSMA refers to our multimodal segmentation architecture that builds upon AdapNet++ and is detailed in Chapter 5.

| Network | Multimodal | mIoU (%) |
|---|---|---|
| Enet [174] | - | 37.6 |
| PSPNet [135] | - | 47.5 |
| 3DMV (2d proj) [175] | ✓ | 49.8 |
| FuseNet [162] | ✓ | 52.1 |
| AdapNet++ (Ours) | - | 50.3 |
| SSMA (Ours) | ✓ | 57.7 |

outperforms it by more than twice the margin. However, this is only observed in the indoor datasets, while in the outdoor datasets DeconvNet performs comparable to the other networks. This is primarily due to the fact that indoor datasets have more number of small classes and the predictions of DeconvNet do not retain them.

Table 4.7 shows the results on the ScanNet test set. We compare against the top performing models on the leaderboard, namely, FuseNet [162], 3DMV (2d proj) [175], PSPNet [135], and Enet [174]. Note that 3DMV and FuseNet are multimodal fusion methods. Our proposed AdapNet++ model outperforms all the unimodal networks and achieves state-of-the-art performance for unimodal semantic segmentation on the ScanNet benchmark.

Finally, we also benchmark on our proposed Freiburg Forest dataset as it is the largest dataset to provide semantically labeled training data of unstructured forested environments. We show the results on the Freiburg Forest dataset in Table 4.6, where our proposed AdapNet++ outperforms the state-of-the-art by 0.82%. Note that this dataset contains large objects such trees and it does not contain thin structures or objects in multiple scales. Therefore, the improvement produced by AdapNet++ is mostly due to the proposed decoder which yields an improved resolution of segmentation along the object boundaries. The actual utility of this dataset is seen in the qualitative multimodal semantic segmentation results presented in Chapter 5, where the fusion helps to improve the segmentation in the presence of disturbances such as glare on the optics and snow. Nevertheless, we see the highest improvement of 3.52% in the *obstacle* class, which is the hardest to segment in this dataset as it contains many different types of objects in one category and it has comparatively fewer examples in the dataset

Moreover, we also compare the number of parameters and the inference time with the baseline networks in Table 4.6. Our proposed AdapNet and AdapNet++ architectures perform inference in 61.81 ms and 72.77 ms respectively on an NVIDIA TITAN X GPU

**Table 4.8:** Comparison of network compression approaches on our AdapNet++ model trained on the Cityscapes dataset.

| Technique | mIoU | Param | FLOPS | Reduction % of | |
|---|---|---|---|---|---|
| | (%) | (M) | (B) | Param | FLOPS |
| Original | 80.77 | 30.20 | 138.47 | – | – |
| Baseline | 80.67 | 28.57 | 136.05 | -5.40 | -1.75 |
| Oracle | 80.80 | 28.34 | 135.64 | -6.15 | -2.04 |
| | 80.56 | 23.67 | 125.33 | -21.62 | -9.49 |
| | 80.18 | 21.66 | 83.84 | -28.28 | -39.45 |
| | 79.65 | 19.91 | 81.72 | -34.07 | -40.98 |
| | 77.95 | 17.79 | 79.84 | -41.09 | -42.34 |
| Oracle with skip (Ours) | **80.80** | 28.14 | 135.17 | **-6.82** | **-2.38** |
| | **80.58** | 23.16 | 124.14 | **-23.31** | **-10.34** |
| | **80.21** | 21.11 | 83.01 | **-30.10** | **-40.05** |
| | **79.68** | 19.75 | 81.53 | **-34.60** | **-41.12** |
| | **78.05** | 17.63 | 79.48 | **-41.62** | **-42.60** |

which is substantially faster than the top performing architectures in all the benchmarks. Most of them consume more than twice the amount of time and the number of parameters making them unsuitable for real-world resource constrained applications. Our critical design choices enable AdapNet++ to consume only 10.98 ms more than AdapNet, while exceeding its performance in each of the benchmarks by a large margin. This shows that AdapNet++ achieves the right performance vs. compactness trade-off which enables it to be employed in not only resource critical applications, but also in applications that demand efficiency and a fast inference time.

### 4.4.5 Evaluation of Model Compression

In this section, we present empirical evaluations of our proposed pruning strategy that is invariant to shortcut connections. We experiment with pruning entire convolutional filters which results in the removal of its corresponding feature map and the related kernels in the following layer. Most existing approaches only prune the first and the second convolution layer of each residual block, or in addition, equally prune the third convolution layer similar to the shortcut connection. However, this equal pruning strategy always leads to

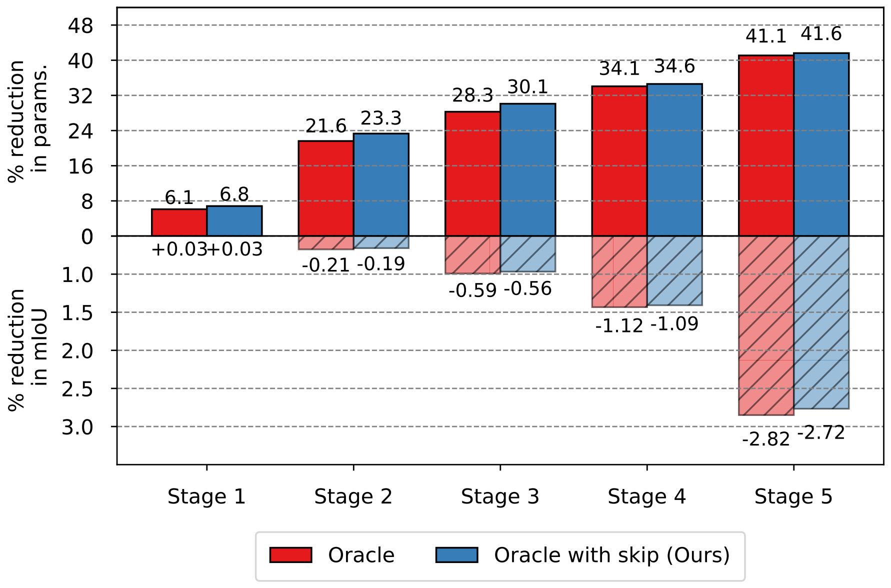


**Figure 4.16:** Evaluation of network compression approaches shown as the percentage of reduction in the number of parameters with the corresponding decrease in the mIoU score for various baseline approaches versus our proposed technique. The results are shown for the AdapNet++ model trained on the Cityscapes dataset.

a significant drop in the accuracy of the model that is not recoverable [152]. Therefore, recent approaches have resorted to omitting pruning of these connections. Contrarily, our proposed technique is invariant to the presence of identity or projection shortcut connections, thereby making the pruning more effective and flexible. We employ a greedy pruning approach but rather than pruning layer by layer and fine-tuning the model after each step, we perform pruning of entire residual blocks at once and then perform the fine-tuning. As our network has a total of 75 convolutional and deconvolutional layers, pruning and fine-tuning each layer will be extremely cumbersome. Nevertheless, we expect a higher performance employing a fully greedy approach.

We compare our strategy with a baseline approach [152] that uses the $\ell^1$-norm of the convolutional filters to compute their importance as well as the approach that we build upon that uses the Taylor expansion criteria [140] for the ranking as described in Section 4.2.3. We denote the approach of [140] as Oracle in our results. In the first stage, we start by pruning only the *Res5* block of our model as it contains the most number of filters, therefore, a substantial amount of parameters can be reduced without any loss in accuracy. As shown in Table 4.8, our approach enables a reduction of 6.82% of the parameters and 3.3 B FLOPS with a slight increase in the mIoU metric. Similar to our approach the original Oracle approach does not cause a drop in the mIoU metric but achieves a lower reduction in parameters. Whereas, the baseline approach achieves a smaller reduction in the parameters and simultaneously causes a drop in the mIoU score.

Our aim for pruning in the first stage was to compress the model without causing a drop in the segmentation performance, while in the following stages, we aggressively prune the model to achieve the best parameter to performance ratio. Results from this experiment are shown as the percentage in reduction of parameters in comparison to the change in mIoU in Figure 4.16. In the second stage, we prune the convolutional feature maps of *Res2*, *Res3*, *Res4* and *Res5* layers. Using our proposed method, we achieve a reduction of 23.31% of parameters with minor drop of 0.19% in the mIoU score. Whereas, the Oracle approach yields a lower reduction in parameters as well as a larger drop in performance. A similar trend can also be seen for the other pruning stages where our proposed approach yields a higher reduction in parameters and FLOPS with a minor reduction in the mIoU score. This shows that pruning convolutional feature maps with regularity leads to a better compression ratio than selectively pruning layers at different stages of the network. In the third stage, we perform pruning of the deconvolutional feature maps, while in the fourth and fifth stages we further prune all the layers of the network by varying the threshold for the rankings. In the final stage we obtain a reduction of 41.62% of the parameters and 42.60% of FLOPS with a drop of 2.72% in the mIoU score. Considering the compression that can be achieved, this minor drop in the mIoU score is acceptable to enable efficient deployment in resource constrained applications.

### 4.4.6 Ablation Study

In order to evaluate the various components of our AdapNet and AdapNet++ architectures, we performed several experiments in different settings. In this section, we study the improvement obtained due to the proposed encoder with the multiscale residual units, a detailed analysis of the proposed eASPP, comparisons with different base encoder network topologies, the improvement that can be obtained by using higher resolution images as input and using multiscale testing. For each of these components, we also study the effect of different parameter configurations. All the ablation studies presented in this section were performed using models trained on the Cityscapes dataset.

#### 4.4.6.1 Detailed Study on the Architectural Components

We first study the major contributions made to the encoder as well as the decoder in our proposed architectures. Table 4.9 shows results from this experiment and subsequent improvement due to each of the configurations. The simple base model M1 consisting of the standard ResNet-50 architecture for the encoder and a single deconvolution layer for upsampling achieves a mIoU of 75.22%. The model M2 that incorporates our multiscale residual units achieves an improvement of 1.7% without any increase in the memory consumption. Whereas, the multigrid approach from DeepLab v3 [132] in the same configuration achieves only 0.38% of improvement in the mIoU score. This shows the

**Table 4.9:** Effect of the various contributions proposed in the AdapNet and AdapNet++ architectures. The performance is shown for the models trained on the Cityscapes dataset. The symbol $\uparrow_f^k$ denotes a deconvolution layer and $c_f^k$ denotes a convolution layer with $f$ number of filters, $k \times k$ kernel size and $n$ is the number of classes. PA ResNet-50 refers to the full pre-activation ResNet-50 architecture. The weights for the two auxilary losses were set to $\mathcal{L}_1 = 0.5$ and $\mathcal{L}_2 = 0.6$ for this experiment.

| Model | Encoder | MS | Decoder | Skip | ASPP | Aux Loss | He | mIoU |
|---|---|---|---|---|---|---|---|---|
| | | Residual | | | dil 3,6,12 | 2x | Init | (%) |
| M1 | ResNet-50 | - | $c_n^1 \uparrow_n^{16}$ | - | - | - | - | 75.22 |
| M2 | ResNet-50 | ✓ | $c_n^1 \uparrow_n^{16}$ | - | - | - | - | 76.92 |
| M3 | ResNet-50 | ✓ | $c_n^1 \uparrow_{2n}^2 \oplus \uparrow_n^8$ | 3d | - | - | - | 77.78 |
| M4 | PA ResNet-50 | ✓ | $c_n^1 \uparrow_{2n}^2 \oplus \uparrow_n^8$ | 3d | - | - | - | 78.44 |
| M5 | PA ResNet-50 | ✓ | $c_n^1 \uparrow_{2n}^2 \oplus \uparrow_n^8$ | 3d | ✓ | - | - | 78.93 |
| M6 | PA ResNet-50 | ✓ | $c_n^1 \uparrow_{2n}^2 \oplus \uparrow_{2n}^2 \oplus \uparrow_n^8$ | 3d,2c | ✓ | - | - | 79.19 |
| M7 | PA ResNet-50 | ✓ | $\uparrow^2\Vert\ c^3\, c^3\ \uparrow^2\Vert\ c^3\, c^3\, c_n^1\ \uparrow_n^8$ | 3d,2c | ✓ | - | - | 79.82 |
| M8 | PA ResNet-50 | ✓ | $\uparrow^2\Vert\ c^3\, c^3\ \uparrow^2\Vert\ c^3\, c^3\, c_n^1\ \uparrow_n^8$ | 3d,2c | ✓ | ✓ | - | 80.34 |
| M9 | PA ResNet-50 | ✓ | $\uparrow^2\Vert\ c^3\, c^3\ \uparrow^2\Vert\ c^3\, c^3\, c_n^1\ \uparrow_n^8$ | 3d,2c | ✓ | ✓ | ✓ | **80.67** |

novelty in employing our multiscale residual units for efficiently learning multiscale features throughout the network. In the M3 model, we study the effect of incorporating skip connections for refinement. Skip connections that were initially introduced in the FCN architecture are still widely used for improving the resolution of the segmentation by incorporating low or mid-level features from the encoder into the decoder while upsampling. The ResNet-50 architecture contains the most discriminative features in the middle of the network. In our M3 model, we first upsample the encoder output by a factor two, followed by fusing the features from *Res3d* block of the encoder to refinement and subsequently using another deconvolutional layer to upsample back to input resolution. This model which resembles the topology of the AdapNet architecture achieves a further improvement of 0.86%.

In the M4 model, we replace the standard residual units with the full pre-activation residual units which yields an improvement of 0.66%. As mentioned in the work by He *et al.* [72], the results corroborate that pre-activation residual units yields a lower error than standard residual units due to the ease of training and improved generalization capability. Aggregating multiscale context using ASPP has become standard practice in most classification and segmentation networks. In the M5 model, we add the ASPP module to the end of the encoder segment. This model demonstrates an improved mIoU of 78.93% due to the ability of the ASPP to capture long range context. In the subsequent M6 model, we study if adding another skip refinement connection from the encoder yields a better performance. This was challenging as most combinations along with the *Res3d* skip connection did not demonstrate any improvement. However, adding a skip connection from *Res2c* showed a slight improvement.

In all the models upto this stage, we fused the low and mid-level encoder features into the decoder using element-wise addition. In order to make the decoder stronger, we experiment with improving the learned decoder representations with additional convolutions after concatenation of the mid-level features. Specifically, the M7 model has three upsampling stages, the first two stages consist of a deconvolution layer that upsamples by a factor of two, followed by concatenation of the mid-level features and two following $3 \times 3$ convolutions that learn highly discriminative fused features. This model shows an improvement of 0.63% which is primarily due to the improved segmentation along the object boundaries as demonstrated in the qualitative results in Section 4.4.7. Our M7 model contains a total of 75 convolutional and deconvolutional layers, making the optimization challenging. In order to accelerate the training and to further improve the segmentation along object boundaries, we propose a multi-resolution supervision scheme in which we add a weighted auxiliary loss to each of the first two upsampling stages. This model denoted as M8 achieves an improved mIoU of 80.34%. In comparison to aforementioned scheme, we also experimented with adding a weighted auxiliary loss at the end of the encoder of the M7 model, however it did not improve the performance, although it accelerated the training. Finally we also experimented with initializing the layers with the He initialization [93]

**Table 4.10:** Evaluation of various atrous spatial pyramid pooling configurations. The performance is shown for the models trained on the Cityscapes dataset. The symbol $\uparrow_f^k$ denotes a deconvolution layer and $c_f^k$ refers to a convolution layer with $f$ number of filters and $k \times k$ kernel size. The weights for the two auxilary losses were set to $\mathcal{L}_1$ = 0.5 and $\mathcal{L}_2$ = 0.6 for this experiment.

| Model | ASPP Conv. | Decoder Conv. | mIoU (%) | Param (M) | FLOPS (B) |
|---|---|---|---|---|---|
| M91 | $[c_{256}^3]$ | $[c_{256}^1]$ | 80.06 | 41.3 | 115.99 |
| M92 | $[c_{256}^3]$ | $[c_{256}^3]$ | 80.27 | 42.5 | 142.42 |
| M93 | $[c_{256}^3]$ | $[c_{256}^3] \times 2$ | 80.67 | 43.7 | 169.62 |
| M94 | $[c_{64}^1\ c_{64}^3\ c_{256}^1]$ | $[c_{256}^3] \times 2$ | 80.42 | 30.1 | 138.21 |
| M95 | $[c_{64}^1\ c_{64}^3\ c_{64}^3\ c_{256}^1]$ | $[c_{256}^3] \times 2$ | **80.77** | 30.2 | 138.47 |

scheme (also known as MSRA) in the M9 model which further boosts the mIoU to 80.67%. The following section further builds upon the M9 model to yield the topology of our proposed AdapNet++ architecture.

#### 4.4.6.2 Detailed study on the eASPP

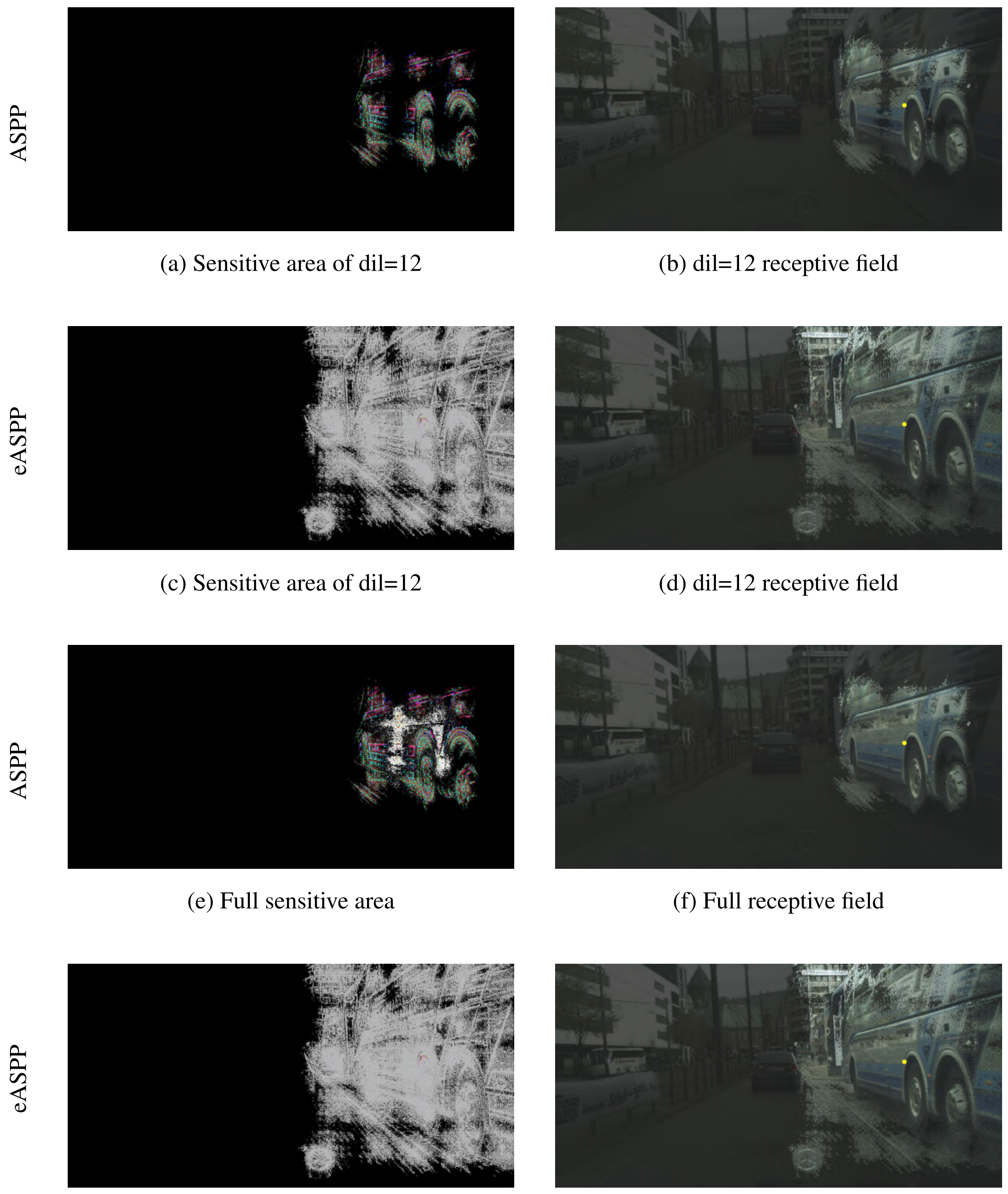
In this section, we quantitatively and qualitatively evaluate the performance of our proposed eASPP configuration and the new decoder topology. We perform all the experiments in this section using the best performing M9 model described in Section 4.4.6.1. In the first configuration of the M91 model, we employ a single 3 × 3 atrous convolution in the ASPP, similar the configuration proposed in DeepLab v3 [132] and use a single 1 × 1 convolution in the place of the two 3 × 3 convolutions in the decoder of the M9 model. This model achieves an mIoU score of 80.06% with 41.3 M parameters and consumes 115.99 B FLOPS. In order to better fuse the concatenated mid-level features with the decoder and to improve its discriminability, we replace the 1 × 1 convolution layer with a 3 × 3 convolution in the M92 model and with two 3 × 3 convolutions in the M93 model. Both these models demonstrate an increase in performance corroborating that a simple 1 × 1 convolution is insufficient for object boundary refinement using fusion of mid-level encoder features.

In an effort to reduce the number of parameters, we employ a bottleneck architecture in the ASPP of the M94 model by replacing the 3 × 3 atrous convolution with a structure consisting of a 1 × 1 convolution with half the number of filters, followed by a 3 × 3 atrous convolution with half the number of filters and another 1 × 1 convolution with the original amount of filters. This model achieves an mIoU score of 80.42% which accounts to a reduction of 0.25% in comparison to the M93 model, however, it reduces computational requirement by 13.6 M parameters and 31.41 B FLOPS which makes the model very efficient. Nevertheless, this drop in performance is not ideal. Therefore, in order to compensate for this drop, we leverage the idea of cascading atrous convolutions that enables an increase in the size of the effective receptive field and the density of the pixel sampling. Specifically, in the M95 model, we add a cascaded $3 \times 3$ atrous convolution in place of the single $3 \times 3$ atrous convolution in the M94 model. This model achieves a mIoU score of 80.77% which is an increase of 0.35% in the mIoU with only a minor increase of 0.1 M parameters in comparison to our M94 model. The originally proposed ASPP module consumes 15.53 M parameters and 34.58 B FLOPS, where the cascaded bottleneck structure in the M95 model only consumes 2.04 M parameters and 3.5 B FLOPS which is over 10 times more computationally efficient. We denote this cascaded bottleneck structure as eASPP.

In order to illustrate the phenomenon caused by cascading atrous convolutions, we visualize the empirical receptive field using the approach proposed by Zhou *et al.* [176]. First, for each feature vector representing an image patch, we use a $8 \times 8$ mean image to occlude the patch at different locations using a sliding window. We then record the change in the activation by measuring the Euclidean distances as a heat map which indicates which regions are sensitive to the feature vector. Although the size of the empirical receptive fields is smaller than theoretical receptive fields, they are better localized and more representative of the information they capture [176]. In Figure 4.17, we show visualizations of the empirical receptive field size of the convolution layer of the ASPP that has one $3 \times 3$ atrous convolution in each branch in comparison to our M95 model that has cascaded $3 \times 3$ atrous convolutions. Figures 4.17 (b) and (d) show the receptive field at the annotated yellow dot for the atrous convolution with the largest dilation rate in ASPP and in our eASPP correspondingly. It can be seen that our eASPP has a much larger receptive field that enables capturing large contexts. Moreover, it can be seen that the pixels are sampled much denser in our eASPP in comparison to the ASPP. In Figures 4.17 (f) and (h), we show the aggregated receptive fields of the entire module in which it can be observed that our eASPP has much lesser isolated points of focus and a cleaner sensitive area than the ASPP. We evaluated the generalization of our proposed eASPP by incorporating the module into our AdapNet++ architecture and benchmarking its performance in comparison to DeepLab which incorporates the ASPP. The results presented in Section 4.4.1 demonstrate that our eASPP effectively generalizes to a wide range of datasets containing diverse environments.

#### 4.4.6.3 Improving the Granularity of Segmentation

In our AdapNet++ architecture, we propose two strategies to improve the segmentation along object boundaries in addition to the new decoder architecture. The first being the two skip refinement stages that fuse low and mid-level encoder features from *Res3d* and *Res2c* into the decoder for object boundary refinement. However, as the low and mid-level features have a large number of filters (512 in *Res3d* and 256 in *Res2c*) in comparison to



(a) Sensitive area of dil=12

(b) dil=12 receptive field

(c) Sensitive area of dil=12

(d) dil=12 receptive field

(e) Full sensitive area

(f) Full receptive field

(g) Full sensitive area

(h) Full receptive field

**Figure 4.17:** Comparison of the receptive field of ASPP and our proposed eASPP. The receptive field is visualized for the annotated yellow dot shown in Figures b, d, f, and h. Our proposed eASPP has larger receptive field size and denser pixel sampling in comparison to the ASPP.

**Table 4.11:** Effect on varying the number of filters in the skip refinement connections in the M95 model. The performance is shown for the models trained on the Cityscapes dataset.

| Skip Channels | 12 | 24 | 36 | 48 | 60 |
|---|---|---|---|---|---|
| mIoU (%) | 80.50 | **80.77** | 80.67 | 80.59 | 80.56 |

**Table 4.12:** Effect on varying the weighting factor of the auxiliary losses in the M95 model. The performance is shown for the models trained on the Cityscapes dataset.

| Aux 1 Weight | 0.4 | 0.4 | 0.5 | 0.5 | 0.6 | 0.6 |
|---|---|---|---|---|---|---|
| Aux 2 Weight | 0.2 | 0.3 | 0.4 | 0.6 | 0.4 | 0.5 |
| mIoU (%) | 80.55 | 80.68 | 80.60 | 80.55 | 80.53 | **80.77** |

the decoder filters that only have 256 feature channels, they will outweigh the high-level features and decrease the performance. Therefore, we employ a $1 \times 1$ convolution to reduce the number of feature channels in the low and mid-level representations before fusing them into the decoder. In Table 4.11, we show results varying the number of feature channels in the $1 \times 1$ skip refinement convolutions in the M95 model from Section 4.4.6.2. We obtain the best results by reducing the number of low and mid-level encoder feature channels to 24 using the $1 \times 1$ convolution layer.

The second strategy that we employ for improving the segmentation along object boundaries is using our proposed multi-resolution supervision scheme. As described in Section 4.2.2.3, we employ auxiliary loss branches after each of the first two upsampling stages in the decoder to improve the resolution of the segmentation and to accelerate training. Weighing the two auxiliary losses is critical to balance the gradient flow through all the previous layers of the network. We experiment with different loss weightings and report results for the same M95 model in Table 4.12. The network achieves the highest performance for auxiliary loss weightings $\lambda_1 = 0.6$ and $\lambda_2 = 0.5$ for $\mathcal{L}_{aux1}$ and $\mathcal{L}_{aux2}$ respectively.

In order to quantify the improvement specifically along the object boundaries, we evaluate the performance of our architecture using the trimap experiment [177]. The mIoU score for the pixels that are within the trimap band of the void class labels (255) are computed by applying the morphological dilation on the void labels. Results from this experiment shown in Figure 4.18 demonstrates that our new decoder in AdapNet++ improves the performance along object boundaries compared to the decoder in AdapNet, while the M7 model with the new decoder and the skip refinement further improves the performance. Finally, the M8 model consisting of our new decoder with the skip refinement stages and our multi-resolution supervision scheme for training significantly improves the

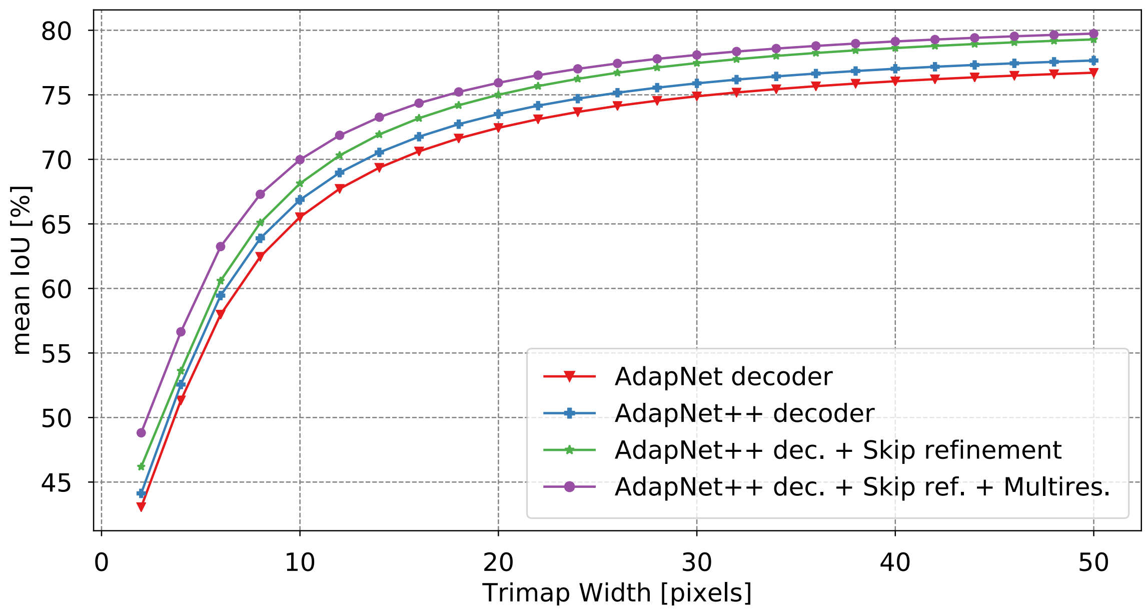


**Figure 4.18:** Influence of the proposed decoder, skip refinement and multi-resolution supervision in AdapNet++ on the segmentation along objects boundaries using the trimap experiment. The plot shows the mIoU score as a function of the trimap band width along the object boundaries.

**Table 4.13:** Effect on varying the number of filters in the skip refinement connection in the M95 model. The performance is shown for the models trained on the Cityscapes dataset.

| Encoder | ResNet | PA ResNet | ResNeXt | SEnet | Xception |
|---|---|---|---|---|---|
| mIoU (%) | 79.32 | **80.77** | 80.30 | 78.31 | 78.70 |
| Param (M) | 30.2 | 30.2 | 29.7 | 32.7 | 27.5 |
| FLOPS (B) | 135.28 | 138.47 | 145.81 | 145.34 | 137.06 |

segmentation along the boundaries which is more evident when the trimap band is narrow.

### 4.4.6.4 Encoder Topology

In recent years, several efficient network architectures have been proposed for image classification that are computationally inexpensive and have fast inference times. In order to study the trade-off between accuracy and computational requirements, we performed experiments using five widely employed architectures as the encoder backbone. Specifically, we evaluate the performance using ResNet-50 [27], full pre-activation ResNet-50 [72], ResNeXt [178], SEnet [179] and Xception [180] architectures for the encoder topology and augmented them with our proposed modules, similar to the M95 model described in Section 4.4.6.2.

Results from this experiment are shown are Table 4.13. Note that in the comparisons presented in this section, no model compression has been performed. It can be seen that the full pre-activation ResNet-50 model achieves the highest mIoU score, closely followed by the ResNeXt model. However the ResNeXt model has an additional 7.34 M parameters with a slightly lesser number of FLOPS. While, the standard ResNet-50 architecture that we employ in AdapNet has 3.19 M parameters lesser than the full pre-activation ResNet-50 model, it achieves a lower mIoU score of 79.32%. Therefore, we choose the full pre-activation ResNet-50 architecture as the encoder backbone in our AdapNet++ architecture.

#### 4.4.6.5 Image Resolution and Testing Strategies

We further performed experiments using input images with larger resolutions as well as with left-right flipped inputs and multiscale inputs while testing. In all our benchmarking experiments, we use an input image with a resolution of 768 $\times$ 384 pixels in order to enable fast inference on a consumer grade GPU. State-of-the-art semantic segmentation architectures often use the full resolution of the image in the datasets as input which could be upto 2048 $\times$ 1024 pixels. Using the full resolution image as input yields a downsampled output with a larger resolution at the end of the encoder, thereby leading to a lesser loss of information due to downsampling and more boundary delineation. Employing a larger resolution image as input also enables better segmentation of small objects that are at far away distances, especially in urban driving datasets such as Cityscapes. However, the caveat being that it requires multi-GPU training with synchronized batch normalization in order to utilize a large enough mini-batch size, which makes the training more cumbersome. Moreover, using a larger input image quickly increases the inference time of the model.

Nevertheless, we present experimental results using AdapNet++ with input images of resolution 896 $\times$ 448 pixels and 1024 $\times$ 512 pixels, in addition to the resolution of 768 $\times$ 384 pixels that we use for the benchmarking experiments. Note that our model is compact enough to train on images with these resolutions in a single consumer grade GPU with 12 GB of memory. In addition to the varying input resolutions, we also test with left-right flips and multiscale inputs. However, although this increases the mIoU score it substantially increases the computation complexity and runtime, therefore rendering it not useful to improve the performance in real-world applications. A summary of the results from this experiment are shown in Table 4.14.

It can be seen that with each higher resolution image, the model yields an increased mIoU score and simultaneously consumes a larger inference time. Similarly, left-right flips and multiscale inputs also yield an improvement in the mIoU score. For the input image resolution of 768 $\times$ 384 pixels, left-right flips yields an increase of 0.58% in the mIoU, while multiscale inputs in addition, yields a further improvement of 0.9%. The corresponding pixel accuracy and and average precision also shows an improvement. The model with an input image of resolution 1024 $\times$ 512 pixels demonstrates an improvement

**Table 4.14:** Effect on using a higher resolution input image and employing left-right flip as well as multiscale inputs during testing. The performance is shown for AdapNet++ models trained on the Cityscapes dataset.

| Image Size (pixels) | Flip | MS | mIoU (%) | Acc. (%) | AP (%) | Time (ms) |
|---|---|---|---|---|---|---|
| 768 × 384 | - | - | 80.77 | 96.04 | 90.97 | 72.77 |
| 768 × 384 | ✓ | - | 81.35 | 96.18 | 90.76 | 148.93 |
| 768 × 384 | ✓ | ✓ | 82.25 | 96.36 | 91.86 | 1775.96 |
| 896 × 448 | - | - | 81.69 | 96.06 | 89.96 | 88.89 |
| 896 × 448 | ✓ | - | 82.28 | 96.21 | 90.53 | 183.57 |
| 896 × 448 | ✓ | ✓ | 83.19 | 96.48 | 91.52 | 2342.31 |
| 1024 × 512 | - | - | 82.47 | 96.13 | 90.63 | 105.94 |
| 1024 × 512 | ✓ | - | 83.07 | 96.28 | 91.17 | 219.28 |
| 1024 × 512 | ✓ | ✓ | **84.27** | **96.66** | **92.36** | 3061.11 |

of 1.7% in the mIoU in comparison to the model with a lower resolution image that we use for benchmarking. Furthermore, using left-right flips and multiscale inputs yields an overall improvement of 3.7% in the mIoU and additional improvements in the other metrics in comparison to the benchmarking model.

### 4.4.7 Qualitative Comparison

In this section, we qualitatively evaluate the semantic segmentation performance of our AdapNet and AdapNet++ architectures in comparison to the best performing state-of-the-art model for each dataset according to the quantitative results presented in Section 4.4.1. We utilize this best performing model as a baseline for the qualitative comparisons presented in this section and show two examples for each dataset that we benchmark on. The colors for the segmented labels shown correspond to the colors and the object categories mentioned in the benchmarking tables shown in Section 4.4.1.

Figures 4.19 (a) and (b) show examples from the Cityscapes dataset in which the improvement over the baseline output (AdapNet) can be seen in the better differentiation between inconspicuous classes such as *sidewalk* and *road* as well as *pole* and *sign*. This can be primarily attributed to the eASPP which has a large receptive field and thus captures larger object context which helps to discern the differences between the inconspicuous classes. The improvement due to better boundary segmentation of thin object classes such

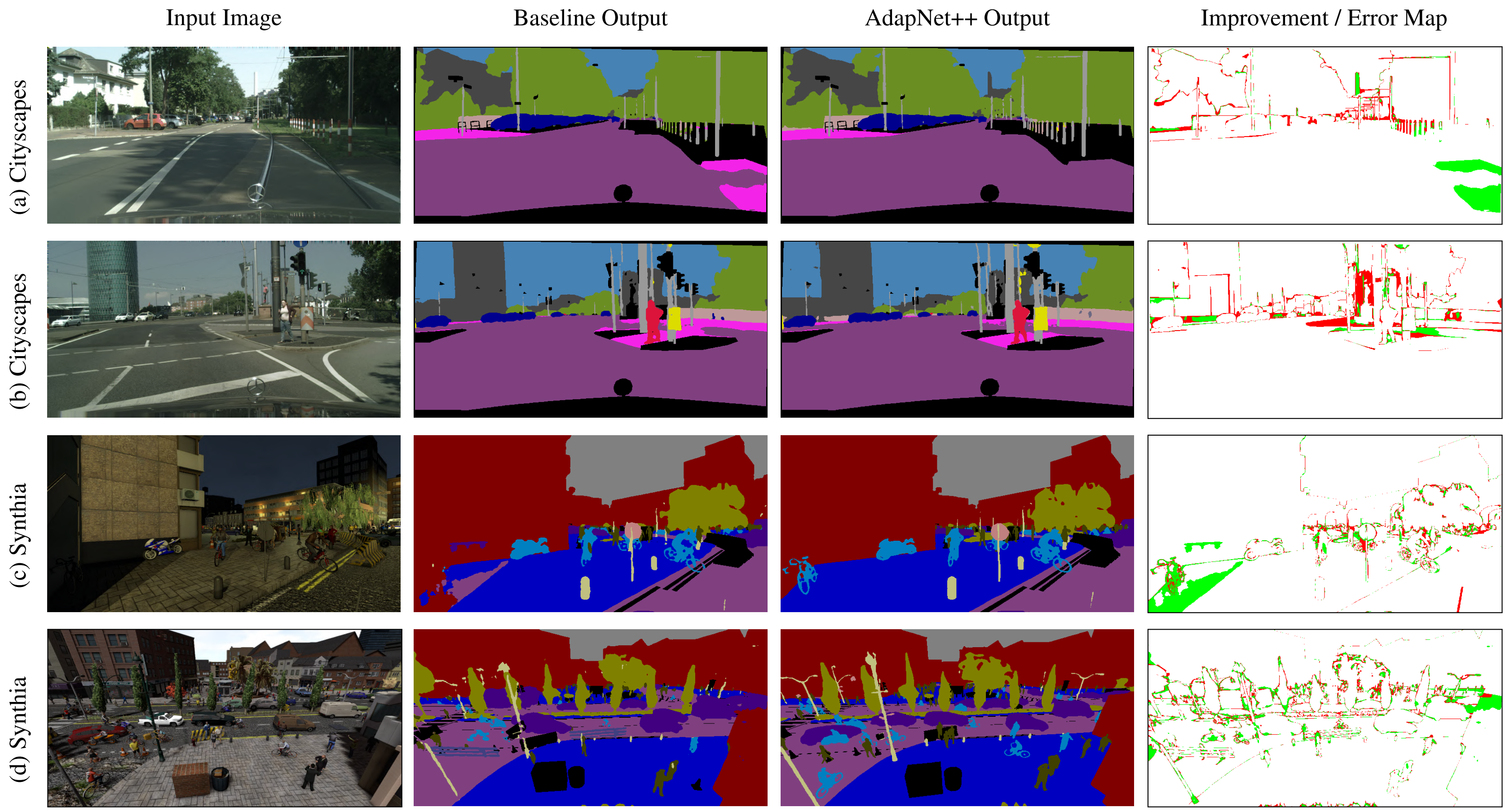


**Figure 4.19:** Qualitative segmentation results of our AdapNet++ architecture in comparison to the best performing state-of-the-art model (AdapNet) on Cityscapes and Synthia datasets. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 4.4.1.

as *poles* can be seen in the images.

Figures 4.19 (c) and (d) show examples from the Synthia dataset, where objects such as *bicycles*, *cars* and *people* are better segmented. The baseline output (AdapNet) shows several missing *cars*, *people* and *bicycles*, whereas the AdapNet++ output accurately captures these objects. Moreover, it can also be seen that the pole-like structures and trees are often discontinuous in the baseline output, while they are more well defined in the AdapNet++ output. In Figure 4.19 (d), an interesting observation is made where an entire fence is segmented in the baseline output but is absent in the scene. This is due to the fact that the intersection of the sidewalk and the road gives an appearance of a fence which is then misclassified. In the same image, it can also be observed that a small building-like structure on the right is not captured, whereas our AdapNet++ model accurately segments the structure.

Figures 4.20 (a) and (b) show examples from the indoor SUN RGB-D dataset. Examples from this dataset show significant misclassification due to inconspicuous objects. Often scenes in indoor datasets have large objects that require the network to have very large receptive fields to be able to accurately distinguish between them. Figure 4.20 (a) shows a scene in which parts of the *chair* and the *table* are incorrectly classified as a *desk* in the output of the baseline model (DeepLab v3). These two classes have very similar structure and appearance which makes distinguishing between them extremely challenging. In Figure 4.20 (b), we can see parts of the *sofa* incorrectly classified in the baseline model output, whereas the entire object is accurately predicted in the AdapNet++ output. In the baseline output, misclassification can also be seen for the *picture* on the wall which is precisely segmented in the AdapNet++ output.

In Figures 4.20 (c) and (d), we show examples from the indoor ScanNet dataset. Figure 4.20 (c) shows misclassification in the output of the baseline model (DeepLab v3) in the boundary where the wall meets the floor and for parts of the *desk* that is misclassified as other furniture. Figure 4.20 (d) shows a significant improvement in the segmentation of AdapNet++ in comparison to the baseline model. The *cabinet* and *counter* are entirely misclassified as a *desk* and *other furniture* correspondingly in the output of the baseline model, whereas they are accurately predicted by our AdapNet++ mode.

Figures 4.21 (a) and (b) show examples from the unstructured Freiburg Forest dataset where the improvement can largely be seen in discerning the object boundaries of classes such as *grass* and *vegetation*, as well as *trail* and *grass*. By observing these images, we can see that even for us humans it is difficult to estimate the boundaries between these classes. Our AdapNet++ architecture predicts the boundaries comparatively better than the baseline model (DeepLab v3). The improvement in the segmentation can also been seen in the finer segmentation of the *vegetation* and the *trail* path in the AdapNet++ output.

Furthermore, Figure 4.21 (c) shows a failure mode where the image is underexposed due to direct sunlight on the optics which causes the obstacle in the left of the image indicated in black to be not fully captured in the segmentation. This could lead to unforeseen situations

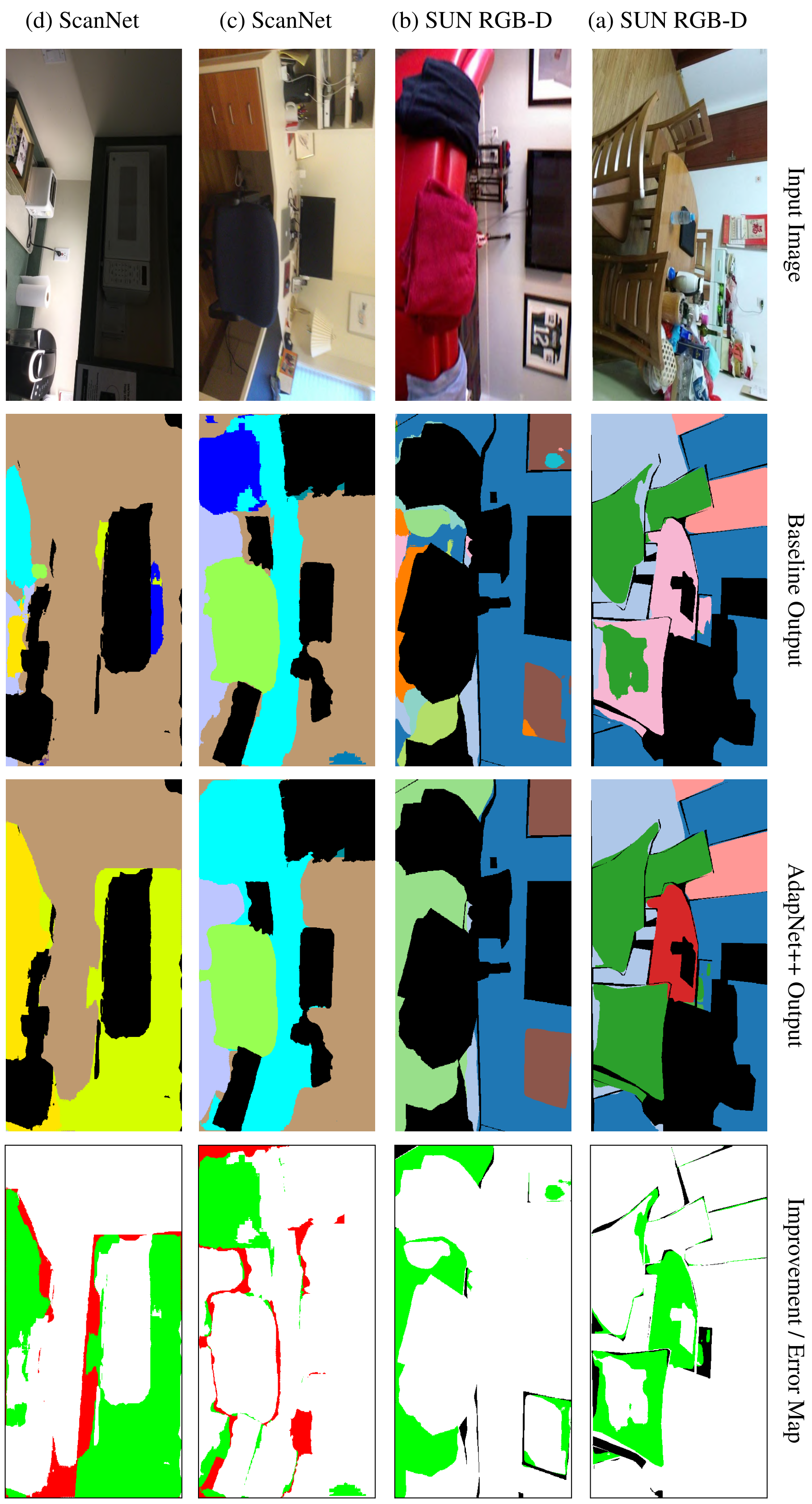


**Figure 4.20:** Qualitative segmentation results of our AdapNet++ architecture in comparison to the best performing state-of-the-art model (DeepLab v3) on SUN RGB-D and ScanNet datasets. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 4.4.1.

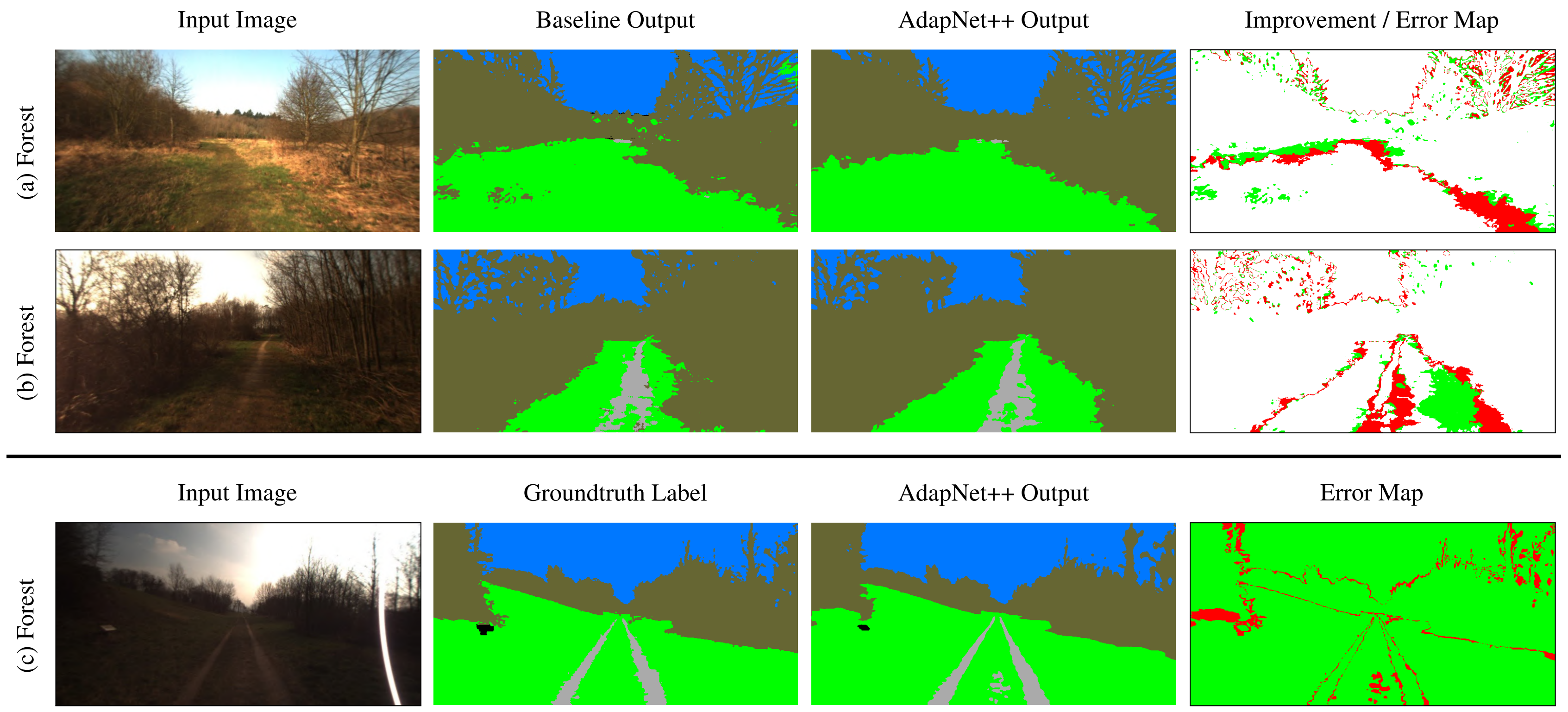


**Figure 4.21:** Qualitative segmentation results of our AdapNet++ architecture in comparison to the best performing state-of-the-art model (DeepLab v3) on the Freiburg Forest dataset. The last row shows a failure mode where the obstacle in the left of the image depicted in black is not fully captured in the segmentation output due to underexposure of the image caused by the glaring sun on the optics. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and in (a, b) the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green, while in (c) the pixels that are correctly classified by AdapNet in green. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 4.4.1.

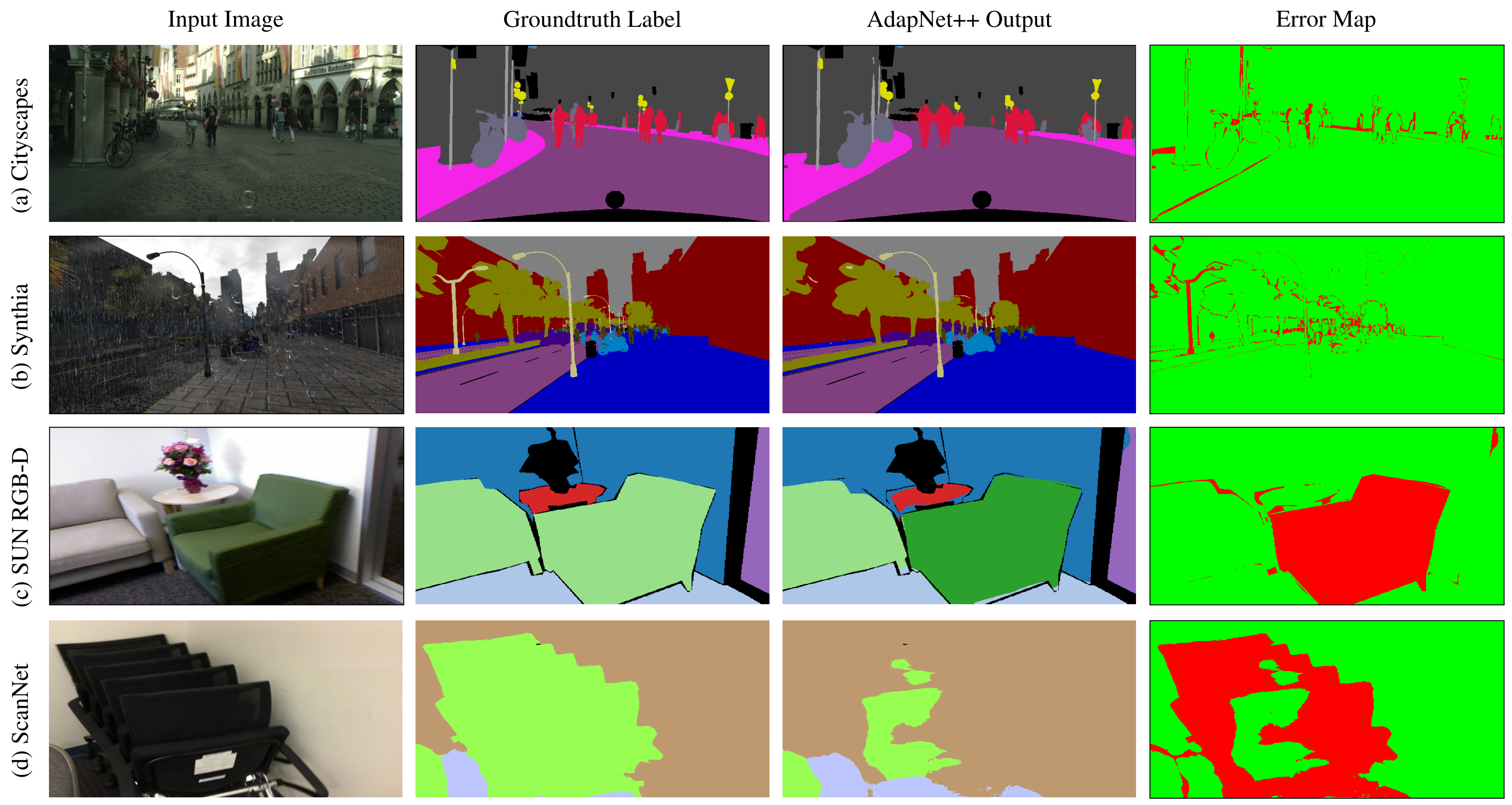


**Figure 4.22:** Example failure modes of our AdapNet++ model that are primarily caused due to: (a) obstacles in the field of view , (b) poor visibility due to weather condition such as rainfall, (c) inconsistent labels in the datasets and (d) change in appearance due to multiple objects being stacked together. In addition to the segmentation output and the groundtruth label, we also show the error map which indicates the misclassified pixels in red and correctly predicted pixels in green. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 4.4.1.

if the robot drives over the area and misses detecting the obstacle. Figure 4.22 shows more failure modes from the other datasets that we benchmark on. In Figure 4.22 (a), the *cyclist* behind the two people in the center of the image is misclassified as a *person* as most of the bicycle is covered by the two people in front of the cyclist. Figure 4.22 (b) shows a dark scene with rainfall which causes a significant amount of *poles* and *persons* that are at far away distances to be not captured in the segmentation due to bad visibility. Figure 4.22 (c) shows an example from the SUN RGB-D dataset where a *sofa* is misclassified as a *chair*. This is caused by the inconsistency in the labels in this dataset. As the SUN RGB-D dataset is a mixture of three different datasets, some instances have the single-person sofa labeled as a *chair*, while the others have it labeled as the *sofa* class which causes confusion among these object classes. In another example, Figure 4.22 (d) shows a scene where the chairs are folded and stacked with each other which gives a completely different appearance than the examples of the *chair* class in the training set. In the next chapter, we demonstrate how leveraging complementary modalities such as depth and near-infrared can enable robust semantic segmentation in most of these situations.

### 4.4.8 Generalization Analysis

In this section, we qualitatively evaluate the generalization performance of our AdapNet and AdapNet++ architectures on data collected by our autonomous car setup in Freiburg. The data was collected over three days in different driving scenarios including both in the inner-city and highways. We use our models trained on the Cityscapes dataset and only test on the images captured in Freiburg. Figure 4.23 shows four examples from this experiment in which we can observe that the structure of these scenes are substantially different than the images in the Cityscapes dataset and a significant amount of illumination changes can be seen. The colors for the segmented labels shown correspond to the colors and the object categories mentioned in the Cityscapes benchmarking results shown in Table 4.1.

The AdapNet++ model in general, demonstrates more accurate semantic segmentation than the AdapNet model in these environments. In Figure 4.23 (a), we see that the AdapNet++ model precisely captures the *fences* in the scene along the right side of the sidewalk as well as traffic *signs* that are at far away distances. While in Figure 4.23 (b), we observe that the AdapNet++ model shows a greater granularity in the segmentation of *vegetation* and it more accurately captures the dividers in the middle of the road as well as *sidewalks*. This can be attributed to the new decoder in AdapNet++ which enables accurate segmentation of thin structures and improves the granularity of the segmentation along the object boundaries. Figure 4.23 (c) shows a hard example where the middle island on the road is at the same level as the road, as opposed to being raised similar to sidewalks. Even in this scenario, the AdapNet++ model reasonable segments the middle island due to its eASPP which enables it to capture long-range context with a large receptive field resolution, whereas AdapNet produces misclassifications in this region. It can also be

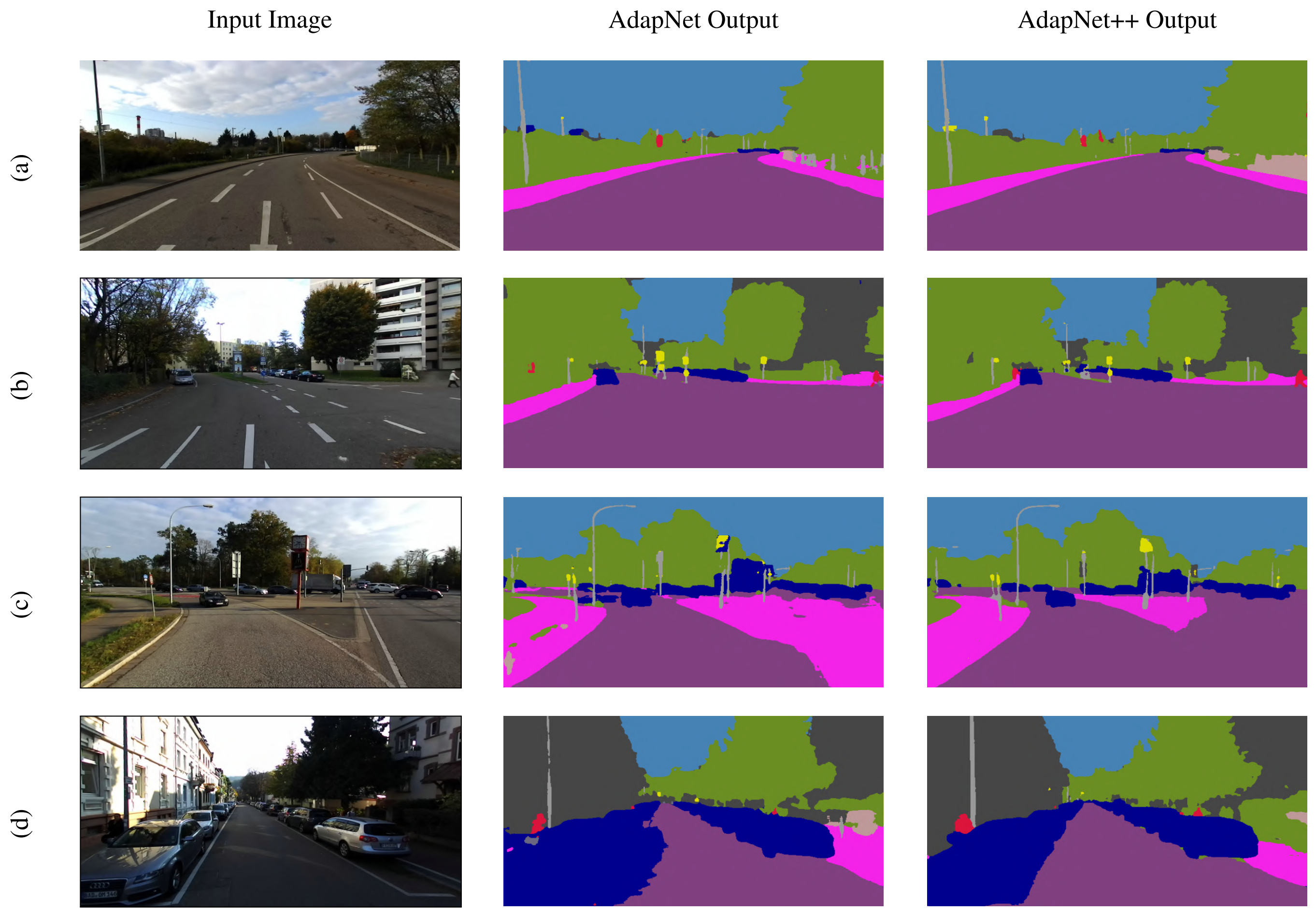


**Figure 4.23:** Qualitative semantic segmentation results of our AdapNet and AdapNet++ architectures demonstrating the generalization ability of our models to previously unseen cities. The models were trained on the Cityscapes dataset and evaluated on images captured in Freiburg using our autonomous car setup. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 4.4.1.

observed that in some instances, AdapNet segments the shadows made by cars as the *car* class and does not capture all the traffic *sings* in the scene, while the AdapNet++ output accurately captures these objects due to the multiscale context aggregation in the encoder. Finally, in Figure 4.23 (d), we can see a more precise segmentation in the AdapNet++ output compared to AdapNet for object classes such as *sidewalks, person, vegetation, fences* and *cars*. Nevertheless, these results demonstrate that our models reliably segments the scene and generalizes effectively to scenes from previously unseen cities that were not present in the training data.

## 4.5 Related Work

In this chapter, we addressed the problem of efficient semantic segmentation using convolutional neural networks. There is a significant amount of prior work in this fundamental perception problem. However, in the last decade, there has been a sharp transition from employing hand engineered features with flat classifiers such as Support Vector Machines [181], Boosting [182] or Random Forests [183, 184], to end-to-end DCNN-based approaches [24, 136]. In order to highlight our contributions, we present a through review in this section by first briefly discussing some of the classical methods before delving into the state-of-the-art techniques.

Semantic segmentation is one of the fundamental problems in computer vision. Some of the earlier approaches for semantic segmentation use small patches to classify the center pixel using flat classifiers [182, 183] followed by smoothing the predictions using Conditional Random Fields (CRFs) [182]. Rather than only relying on appearance based features, structure from motion features have also been used with randomized decision forests [182, 184]. View independent 3D features from dense depth maps have been shown to outperform appearance based features, that also enabled classification of all the pixels in an image, as opposed to only the center pixel of a patch [185]. Plath *et al.* [186] propose an approach to combine local and global features using a CRF and an image classification method. However, the performance of these approaches is largely bounded by the expressiveness of handcrafted features which is highly scenario-specific.

The remarkable performance achieved by CNNs in classification tasks led to their application for dense prediction problems such as semantic segmentation, depth estimation and optical flow prediction. Initial approaches that employed neural networks for semantic segmentation still relied on patch-wise training [187, 188, 189]. Pinheiro *et al.* [189] use a recurrent CNN to aggregate several low-resolution predictions for scene labeling. Clement *et al.* [188] transforms the input image through a Laplacian pyramid followed by feeding each scale to a CNN for hierarchical feature extraction and classification. Although these approaches demonstrated improved performance over handcrafted features, they often yield a grid-like output that does not capture the true object boundaries. One of the first end-to-end approaches that learns to directly map the low-resolution representations from a classification network to a dense prediction output was the Fully Convolutional Network (FCN) model [24]. FCN proposed an encoder-decoder architecture in which the encoder is built upon the VGG-16 [153] architecture with the inner-product layers replaced with convolutional layers. While, the decoder consists of successive deconvolution and convolution layers that upsample and refine the low-resolution feature maps by combining them with the encoder feature maps. The last decoder then yields a segmented output with the same resolution as the input image.

DeconvNet [169] propose an improved architecture containing stacked deconvolution and unpooling layers that perform non-linear upsampling and outperforms FCNs but at the cost of a more complex training procedure. The SegNet [136] architecture eliminates the need for learning to upsample by reusing pooling indices from the encoder layers to perform upsampling. U-Net [122] adds skip connections from the encoder to each corresponding decoder section for biomedical image segmentation. Similarly, RefineNet [190] proposed a network to exploit features along the down-sampling process to enable high-resolution segmentation. Oliveira *et al.* [48] propose an architecture that builds upon FCNs and introduces more refinement stages and incorporates spatial dropout to prevent over fitting. The ParseNet [168] architecture models global context directly instead of only relying on the largest receptive field of the network. Several recent approaches employ Conditional Random Fields (CRFs) in cascade with CNNs to encode long-range context and to improve object boundary segmentation [138, 191]. Some approaches jointly train CNN and CRF components [192, 193], while others employ several convolutional layers on top of the belief maps to capture context information [194]. Vemulapalli *et al.* [195] propose an approach that combines Gaussian Conditional Random Fields with CNNs and outperforms other approaches that combine CNNs with discrete CRF models. However, these techniques are not feed-forward in test time as they require MAP inference over a CRF or other aids such as region proposals.

Recently, there has been more focus on learning multiscale features, which was initially achieved by providing the network with multiple rescaled versions of the image [188] or by fusing features from multiple parallel branches that take different image resolutions [24]. The general goal of these approaches is to provide the network with both local and global context [196] by using features extracted at multiple scales and incorporating feature maps from early network layers for improving prediction along object boundaries as they retain more high frequency details. Most of these networks are difficult to train due to the number of parameters they consume [197], therefore, multi-stage training procedures are often employed. In addition, they have slow runtimes due to multiple convolutional pathways for feature extraction.

In order to alleviate this problem, Yu *et al.* [67] propose dilated convolutions that allows for exponential increase in the receptive field without decrease in resolution or increase in parameters. Since then several approaches have explored the use of dilated convolutions for semantic segmentation. Wu *et al.* [198] study the effect of atrous convolutions with different dilation rates for capturing long-range information. Wang *et al.* [199] employ hybrid atrous rates in a residual network architecture. Dai *et al.* [200] proposes deformable convolutions that generalize dilated convolutions by sampling the input features with a learned offset. DeepLab [132, 138] build upon the aforementioned idea and uses dilated convolutions of different atrous rates to aggregate multiscale global context. Subsequently, Zhao *et al.* introduce the PSPNet [135] architecture that employs spatial pooling at different grid scales to capture multiscale information. However, a major drawback in employing these approaches is the computational complexity and the substantially large inference time even using modern GPUs that hinder them from being deployed in robots that often have limited resources. Conversely, in this chapter we presented several new contributions for learning multiscale features, capturing long range context and improving the upsampling in the decoder, while simultaneously reducing the number of parameters and maintaining a fast inference time. Our proposed architectures that incorporate these techniques achieve a good trade-off between performance and computational complexity of the model for enabling efficient deployment in real-world robotic perception applications.

## 4.6 Conclusions

In this chapter, we presented the novel AdapNet and AdapNet++ architectures for efficient semantic scene segmentation. Our AdapNet architecture follows the encoder-decoder topology and incorporates our multiscale residual units with atrous convolutions that have gradually increasing dilation rates to encode multiscale information throughout the network without increasing the number of parameters. The proposed multiscale residual units are more effective at learning multiscale features and outperform the commonly employed multigrid method. Furthermore, our proposed AdapNet++ architecture builds upon AdapNet and additionally incorporates several new network modules for improving the performance, including the Efficient Atrous Spatial Pyramid Pooling (eASPP), a new strong decoder and the multi-resolution supervision strategy. Our eASPP employs both parallel and cascaded atrous convolutions with different dilation rates in a bottleneck fashion to efficiently capture long range context and probe the features with filters at multiple sampling rates and field of views. The eASPP has a larger effective receptive field and achieves 10 times reduction in the number of parameters with a simultaneous increase in performance compared to the standard ASPP. Our new decoder with skip refinement stages fuses low and mid-level features from the encoder for object boundary refinement. Finally, the proposed multi-resolution supervision scheme accelerates the training and further improves the performance along object boundaries. Additionally, we presented a holistic network-wide pruning approach that is invariant to shortcut connections, to compress our model and to enable efficient deployment.

We introduced a first-of-its-kind Freiburg Forest dataset that contains images of multiple modalities and spectra with pixel-wise ground truth annotations of unstructured forested environments. We presented an exhaustive theoretical analysis, visualizations, quantitative and qualitative results on the Cityscapes, Synthia, SUN RGB-D, ScanNet and Freiburg Forest datasets. The results demonstrate that our architectures achieve state-of-the-art performance with a significantly lesser number of parameters and a substantially faster inference time in comparison to several strong state-of-the-art models. Furthermore, we presented qualitative evaluations on data collected by our autonomous car setup in Freiburg that demonstrates the generalization ability of our model to previously unseen scenarios.

# Chapter 5

# Multimodal Semantic Segmentation

**Learning to reliably perceive and understand the scene is an integral enabler for robots to operate in the real-world. This problem is inherently challenging due to the multitude of object types as well as appearance changes caused by varying illumination and weather conditions. Leveraging complementary modalities can enable learning of semantically richer representations that are resilient to such perturbations. Despite the tremendous progress in recent years, most CNNs directly concatenate different modalities from the outset, or concatenate learned features from modality-specific streams, rendering the model incapable of focusing only on the relevant complementary information. To address this limitation, we propose two mutimodal semantic segmentation frameworks that dynamically adapt the fusion of modality-specific features based on the scene condition. Our first fusion scheme termed CMoDE learns to probabilistically fuse features from modality-specific network streams to exploit the most discriminative complementary class-specific features. While our subsequently proposed SSMA fusion scheme dynamically fuses intermediate representations from modality-specific encoder streams into a single decoder, while being sensitive to the object category, spatial location and scene context in a self-supervised manner. Comprehensive empirical evaluations on several benchmarks show that both our fusion techniques achieve state-of-the-art performance, while demonstrating substantial robustness in adverse perceptual conditions.**

## 5.1 Introduction

Robust scene understanding is a critical and essential task for autonomous navigation. This problem is heavily characterized by changing environmental conditions that take

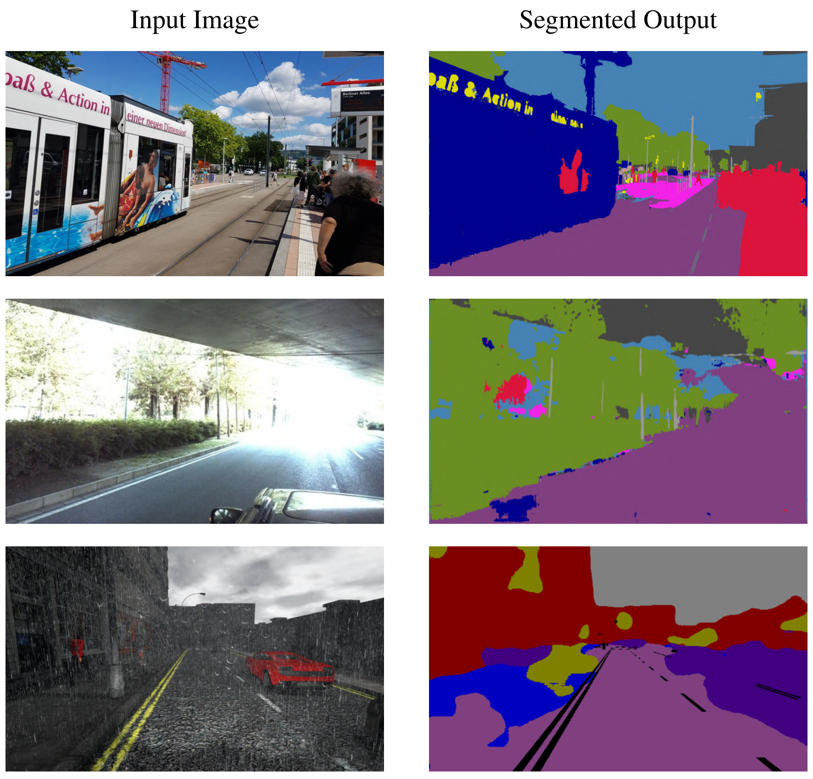


**Figure 5.1:** Example real-world scenarios where current state-of-the-art models demonstrate misclassifications. The first row shows an issue of mismatched relationship as well as inconspicuous classes where a decal on the train is falsely predicted as a person and the decal text is falsely predicted as a traffic sign. While, the second row shows misclassifications caused by overexposure of the camera due to the car exiting a tunnel and finally, the last row shows misclassifications due to bad visibility caused by rain.

place throughout the day and across seasons. Robots should be equipped with models that are impervious to these factors in order to be operable and more importantly to ensure safety in the real-world. A robot with a perception system that is incapable of handling such large visual appearance changes can quickly jeopardize its operation and cause accidents that imperil the people around. As we demonstrated in the previous chapter, deep Convolutional Neural Network (CNN) based approaches have achieved unprecedented performance in semantic segmentation tasks [53, 59, 132, 135]. We presented two efficient semantic segmentation architectures that incorporate several new techniques, including for learning multiscale information, capturing long range context, aggregating multiscale features, improving object boundary refinement and for compressing the model effectively by pruning unimportant neurons. These techniques enable our models to achieve state-of-the-art performance while being compact and having fast inference times.

Nevertheless, state-of-the-art semantic segmentation architectures still face several challenges while being employed in the real-world due to frequent visual appearance changes caused by transitioning weather, illumination and seasons, in addition to the diversity of complex scenes that cause mismatched relationship as well as inconspicuous object classes. Figure 5.1 shows example scenes from real-world scenarios in which misclassifications are produced due to the decal on the train which is falsely predicted as a person and a traffic sign (first row), overexposure of the camera caused by the vehicle exiting a tunnel (second row), and bad visibility conditions caused by rain and cloudy weather (third row). In order to accurately predict the elements of the scene in these situations, features from complementary modalities such as depth and infrared can be leveraged to correspondingly exploit object properties such as geometry and reflectance. Moreover, the network can exploit complex intra-modal dependencies more effectively by directly learning to fuse visual appearance information from RGB images with learned features from complementary modalities in an end-to-end fashion. This not only enables the network to resolve inherent ambiguities and improve the reliability but also obtain a more holistic scene segmentation.

While most existing work focuses on where to fuse modality-specific streams topologically [50, 162, 201] and what transformations can be applied on the depth modality to enable better fusion with visual RGB features [161, 163], it still remains an open question as to how to enable the network to dynamically adapt its fusion strategy based on the nature of the scene such as the types of objects, their spatial location in the world and the present scene context. This is a crucial requirement in applications such as robotics and autonomous driving where these systems run in continually changing environmental contexts. For example, an autonomous car navigating in ideal weather conditions can primarily rely on visual information but when it enters a dark tunnel or exits an underpassage, the cameras might experience under/over exposure, whereas the depth modality will be more informative. Furthermore, the strategy to be employed for fusion also varies with the types of objects in the scene, for instance, infrared might be more useful to detect categories such as people, vehicles, vegetation and boundaries of structures but it does not provide much information on object categories such as the sky. Additionally, the spatial location of objects in the scene also has an influence, for example, the depth modality provides rich information on objects that are at nearby distances but degrades very quickly for objects that are several meters away. More importantly, the approach employed should be robust to sensor failure and noise as constraining the network to always depend on both modalities and use noisy information can worsen the actual performance and lead to disastrous situations.

Due to these complex interdependencies, naively treating modalities as multi-channel input data or concatenating independently learned modality-specific features does not allow the network to adapt to the aforementioned situations dynamically. Moreover, due to the nature of this dynamicity, the fusion mechanism has to be trained in a self-supervised manner in order to make the adaptivity emergent and to generalize effectively to different real-world scenarios. As a solution to this problem, we present two novel multimodal fusion mechanisms: the Convoluted Mixture of Deep Experts (CMoDE) and the Self-Supervised Model Adaptation (SSMA) scheme. We build upon our AdapNet++ semantic segmentation architecture presented in Chapter 4, and propose two multimodal variants incorporating the CMoDE and SSMA fusion mechanisms respectively. The CMoDE acts as a multiplexer and adaptively weighs class-specific features of individual modality-specific network streams using learned probability distributions. The CMoDE module consists of multiple Adaptive Gating Networks (AGN) that each take a two-dimensional encoder feature channel corresponding to particular object class from each modality-specific stream as input and uses them to map the decoder features to a probabilistically fused representation. We employ the CMoDE to fuse the high-level semantically mature features at the end of the modality-specific encoder-decoder streams and feed the output to an additional convolution layer to further learn discriminative complementary fused kernels. Although our CMoDE fusion mechanism exploits complementary features from modality-specific network streams according to the different object classes in the scene, it does not address the fact that the spatial location of the object in the environment also influences the selection of modality-specific features. For example, modalities such as depth are accurate at nearby distances but degrades quickly for objects that are at far away distances, therefore, the alternate visual RGB features should be leveraged for segmenting distant objects in this case.

In order to address this limitation, we present the SSMA fusion mechanism that adaptively recalibrates and fuses modality-specific feature maps based on the object class, its spatial location and the scene context. The SSMA module takes intermediate encoder representations of modality-specific streams as input and fuses them probabilistically based on the activations of individual modality streams. As we model the SSMA module in a fully convolutional fashion, it yields a probability for each activation in the feature maps (as opposed to only a probability for each object class in the CMoDE approach) which represents the optimal combination to exploit complementary properties. These probabilities are then used to amplify or suppress the representations of the individual modality streams, followed by the fusion. As we base the fusion on modality-specific activations, it is intrinsically tolerant to sensor failure and noise such as missing depth values. We employ the SSMA module to fuse representations at the end of the modality-specific encoder streams as well as to fuse the mid-level encoder representations. The fused representations from the SSMA modules are input to the decoder at different stages for upsampling and refining the predictions.

We employ a combination of mid-level fusion and late-fusion as several experiments have demonstrated that fusing semantically meaningful representations yields better performance in comparison to early-fusion [50, 120, 162, 163]. Moreover, studies of the neural dynamics of the human brain has also shown evidence of late-fusion of modalities for recognition tasks [202]. However, intermediate network representations are not aligned across different modality-specific network streams. Hence, integrating fused multimodal mid-level features into high-level features requires explicit prior alignment. Therefore, we propose an attention mechanism that weighs the fused multimodal mid-level skip features with spatially aggregated statistics of the high-level decoder features for better correlation, followed by channel-wise concatenation.

Finally, we present extensive experimental evaluations of our proposed multimodal semantic segmentation architectures on all the datasets that we benchmarked our unimodal AdapNet++ architecture on, in Chapter 4. Specifically, we benchmark on the Cityscapes [143], Synthia [144], SUN RGB-D [145], ScanNet [146] and Freiburg Forest [50] datasets. The results demonstrate that both our dynamically adapting multimodal architectures achieve state-of-the-art performance while being exceptionally robust to adverse perceptual conditions such as fog, snow, rain and night-time. Thus enabling them to be effectively employed in perception critical applications such as robotics, where not only accuracy but robustness of the model is equally important. To the best of our knowledge, this is the first multimodal segmentation work to benchmark on these wide range of datasets containing several modalities and diverse environments ranging from urban city driving scenes to indoor environments and unstructured forested scenes.

In summary, the primary contributions that we make in this chapter are as follows:

- A novel multimodal semantic segmentation architecture incorporating our proposed CMoDE fusion scheme that learns robust kernels from complementary modalities and spectra according to the different object categories in the scene.
- A second novel multimodal fusion architecture incorporating our proposed SSMA fusion modules that adapts the fusion of modality-specific features dynamically according to the object category, its spatial location in the world as well as the scene context and learns in a self-supervised manner.
- An attention mechanism for effectively correlating fused multimodal mid-level and high-level features for better object boundary refinement.
- Extensive benchmarking of existing multimodal fusion approaches with quantitative and qualitative evaluations on five different benchmark datasets consisting of multiple modalities and spectra.
- Real-world results from field trials in which our robot autonomously navigated a forested environment for 4.52 km using only the proposed multimodal semantic segmentation for perception.
- Implementations of our proposed multimodal semantic segmentation architectures are made publicly available at `http://deepscene.cs.uni-freiburg.de`.

The remainder of this chapter is organized as follows. In Section 5.2.1, we describe the topology of our multimodal semantic segmentation architecture that incorporates our proposed CMoDE fusion scheme, followed by the architecture that incorporates our SSMA fusion module in Section 5.2.2 and finally, the topologies of baseline CNN fusion approaches in Section 5.2.3. In Section 5.3, we present extensive experimental evaluations, comprehensive ablation studies and detailed qualitative comparisons in different adverse perceptual conditions. Subsequently, in Section 5.3.7, we present robustness and generalization analysis using navigation experiments in a forested environment where we use only the proposed multimodal segmentation for perception. Finally, we discuss the recent related work on multimodal semantic segmentation in Section 5.4, before concluding the chapter in Section 5.5.

## 5.2 Technical Approach

In this section, we first formulate the problem of multimodal semantic segmentation and then detail the topology of our architecture that incorporates our proposed Convoluted Mixture of Deep Experts (CMoDE) fusion scheme to probabilistically fuse high-level class-specific decoder feature maps from multiple modality-specific network streams. Subsequently, we describe the topology of our improved multimodal semantic segmentation architecture that incorporates our proposed Self-Supervised Modal Adaptation (SSMA) modules to adaptively fuse mid-level and high-level encoder features into the decoder based to object types in the scene, its spatial locations and the scene context. We build upon the AdapNet++ topology described in Chapter 4 for the base architecture and reconfigure its structure for multimodal semantic segmentation.

For ease of notation, we formulate the multimodal semantic segmentation problem in the context of learning from two different modalities. However, our framework is easily adaptable to learn from arbitrary number of modalities. We represent the training set for multimodal semantic segmentation as $\mathcal{T} = \{(I_n, K_n, M_n) \mid n = 1, \ldots, N\}$, where $I_n = \{u_r \mid r = 1, \ldots, \rho\}$ denotes the input frame from modality $a$, $K_n = \{k_r \mid r = 1, \ldots, \rho\}$ denotes the corresponding input frame from modality $b$, $N$ denotes the number of training samples and the groundtruth label is given by $M_n = \{m_r \mid r = 1, \ldots, \rho\}$, where $m_r \in \{1, ..., C\}$ is the set of semantic classes. The image $I_n$ is only shown to the modality-specific encoder $E_a$ and similarly, the corresponding image $K_n$ from a complementary modality is only shown to the modality-specific encoder $E_b$. This enables each modality-specific encoder to specialize in a particular sub-space learning their own hierarchical representations individually. We assume that the input images $I_n$ and $K_n$, as well as the label $M_n$ have the same dimensions $\rho = H \times W$. Let $\theta$ be the network parameters consisting of weights and biases, and $s_j(u_r, \theta)$ as the score assigned for labeling pixel $u_r$ with label $j$. We obtain the probabilities $\mathbf{P} = (p_1, \ldots, p_C)$ for all the semantic classes using the softmax function as

$$p_j(u_r, \theta \mid I_n, K_n) = \sigma\left(s_j\left(u_r, \theta\right)\right) = \frac{exp\left(s_j\left(u_r, \theta\right)\right)}{\sum_k^C exp\left(s_k\left(u_r, \theta\right)\right)}. \tag{5.1}$$

The optimal network parameters are then estimated by minimizing the cross-entropy

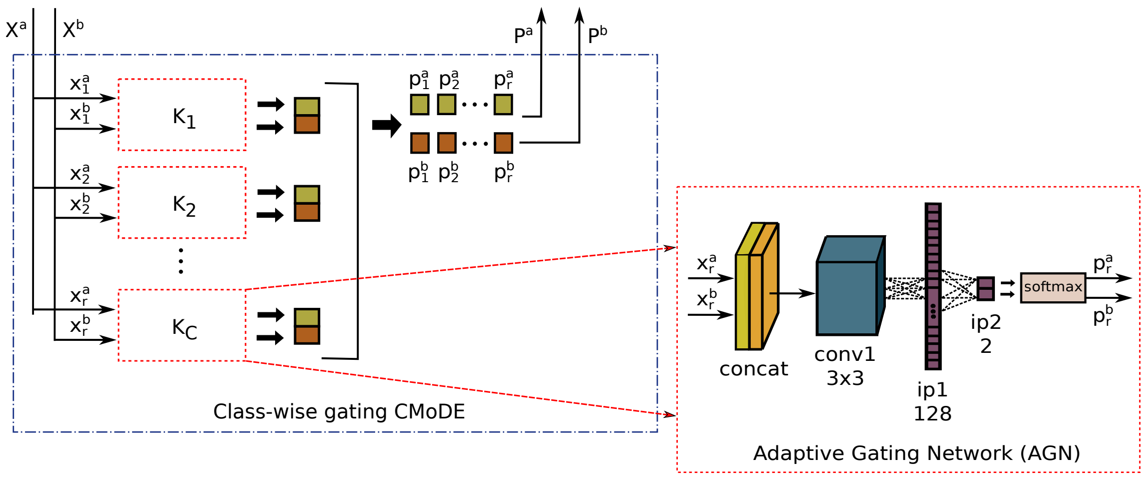


**Figure 5.2:** Topology of our proposed Convoluted Mixture of Deep Experts (CMoDE) fusion module. The CMoDE takes mid-level intermediate network representations from modality-specific network streams as input and outputs the class-wise probabilities representing the confidence in the features learned by the modality-specific network stream for a particular object class of interest.

loss function as

$$\mathcal{L}_{seg}(\mathcal{T}, \theta) = -\sum_{n=1}^{N}\sum_{r=1}^{\rho}\sum_{j=1}^{C} \delta_{m_r,j} \log p_j(u_r, \theta \mid I_n, K_n), \tag{5.2}$$

for $(I_n, K_n, M_n) \in \mathcal{T}$, where $\delta_{m_r,j}$ is the Kronecker delta.

### 5.2.1 Convoluted Mixture of Deep Experts

The CMoDE framework consists of three components: modality-specific encoder-decoder streams for which we use the AdapNet++ architecture, the CMoDE module for adaptively fusing class-specific features and a post-fusion section for learning deeper discriminative fused representations. In the following sections, we first describe the topology of the CMoDE module, followed the structure of the entire multimodal semantic segmentation architecture that incorporates the CMoDE module for fusion.

#### 5.2.1.1 CMoDE Fusion Module

In order to fuse class-specific features of modality-specific network streams we propose the CMoDE module depicted in Figure 5.2. The CMoDE module fuses the high-level features based on the mid-level representations of the individual modality-specific network streams. The mid-level representations are the most discriminative features in the network and they still retain location information of the features as described in Chapter 4. Therefore, we employ them as inputs to the CMoDE to compute the probabilities for the fusion of high-level decoder features. Let $\mathbf{X}^a = \{x_r^a \mid r = 1, \ldots, C\}$ and $\mathbf{X}^b = \{x_r^b \mid r = 1, \ldots, C\}$ denote the mid-level modality-specific encoder feature maps from modality $a$ and modality $b$ network streams respectively, where $C$ is the number of object classes in the dataset and $x_r$ has a spatial dimension of $H \times W$. The class-wise gating network takes $\mathbf{X}^a$ and $\mathbf{X}^b$ as input and outputs the respective probabilities $P^a$ and $P^b$ for the fusion.

The class-wise gating CMoDE network consists of $C$ number of adaptive gating sub-networks (AGN). Each AGN takes the class-specific features $x_r^a$ and $x_r^b$ representing the object class $r$ and from the modality-specific network streams $a$ and $b$ as inputs. The features are first concatenated to obtain $x_r^{ab}$, followed by applying a $3 \times 3$ convolution *conv1* with weights $\mathcal{W}_1 \in \mathbb{R}^{C \times H \times W}$ and a non-linearity function $\delta(\cdot)$. We use ReLUs for the non-linearity and add dropout on the convolution layer to prevent overfitting. The convolved features are then passed through an inner-product layer *ip1* having weights $\mathcal{W}_2 \in \mathbb{R}^{\gamma_1 \times H \times W}$ to learn non-linear combinations of features, where $\gamma_1$ is number of output feature channels in *ip1*. We use $\gamma_1 = 128$ as identified in the ablation study presented in Section 5.3.4.1. Finally, the resulting feature maps are passed into another inner-product layer *ip2* with weights $\mathcal{W}_3 \in \mathbb{R}^{\gamma_2 \times H \times W}$, where $\gamma_2 = 2$ in this case as we fuse features from two modality-specific network streams. The feature maps from the last inner-product layer *ip2* can then be represented as

$$s_r^{ab} = F_{cmode}(\mathbf{x}_r^{ab}; \mathcal{W}) = \mathcal{W}_3 \left( \mathcal{W}_2 \delta \left( \mathcal{W}_1 \mathbf{x}_r^{ab} \right) \right). \tag{5.3}$$

The feature maps $s_r^{ab}$ are then passed through a $\mathsf{softmax}$ layer $\sigma(\cdot)$ and using the fusion scores $f$, we obtain the probabilities $p_r$ for the fusion as

$$p_r(s_r^{ab}, \theta \mid I_n, K_n) = \sigma\left(f\left(s_r^{ab}, \theta\right)\right) = \frac{exp\left(f\left(s_r^{ab}, \theta\right)\right)}{\sum_k^E exp\left(f_k\left(s_r^{ab}, \theta\right)\right)}, \tag{5.4}$$

where $\theta$ are the parameters of the AGN and E is the number of modality-specific network streams. The resulting probabilities $p_r$ denote how much the gating network trusts the class-specific kernels learned the modality-specific streams. This procedure is then employed in $C$ number of AGNs to compute the probabilities for fusion corresponding to each class in the dataset. The resulting class-specific probabilities for the individual modality streams $a$ and $b$ are then concatenated to yield $P^a$ and $P^b$ respectively. This class-specific probability distribution learned by the gating network gives the model the ability to choose different modalitites or spectra according to the object classes present in the scene.

#### 5.2.1.2 CMoDE Fusion Architecture

We incorporate our CMoDE module in a late-fusion framework shown in Figure 5.3. The framework is adaptable to fuse arbitrary number of modalities, for simplicity we consider fusing two different modalities in our descriptions. Individual modality-specific network streams specializing in a particular subspace, first map the representation of the input

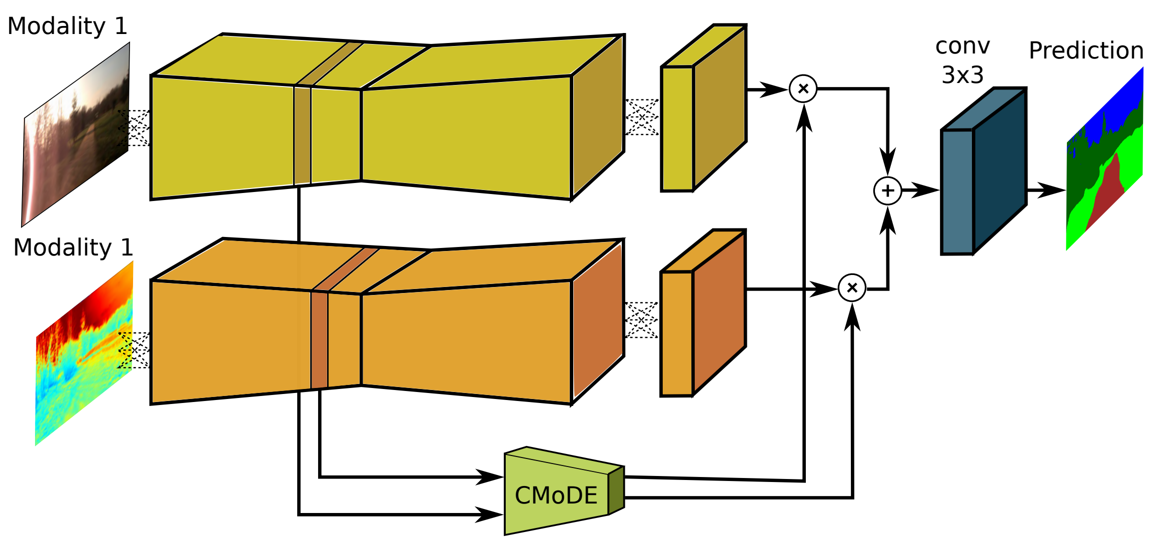


**Figure 5.3:** The late-fusion architecture incorporating the proposed CMoDE fusion module for adaptive multimodal semantic segmentation. Any segmentation network can be employed for the modality-specific network streams, we use our AdapNet++ architecture described in Chapter 4 for this work.

to a corresponding segmentation mask. The CMoDE acts as a multiplexer, which maps outputs of the modality-specific streams to a probabilistically fused representation. We employ our AdapNet++ network architecture for the individual modality-specific network streams. In Chapter 4, we demonstrated that the representations from *Res3d* are the most discriminative in the AdapNet++ architecture. Therefore, we use the feature maps from *Res3d* as the inputs to the CMoDE in this work. We train the CMoDE to learn a convex combination of modality-specific features by back-propagating into the weights, thus making them learnable parameters, similar to any other synapse weight or convolutional kernel. We further describe the training procedure that we employ in Section 5.3.1.1.

We use output class-wise confidences from the CMoDE to weight the feature maps at the end of the modality-specific encoder-decoder streams where the tensors are of dimensions $C \times 384 \times 768$ with $C$ being the number of object classes in the dataset. The weighed modality-specific feature maps are then added element-wise and passed through a $3 \times 3$ convolutional layer to further learn complementary fused kernels. This enables our multimodal semantic segmentation network to learn the most discriminative features for each modality as well as exploit the complementary relationship between the modalities with respect to the different objects in the scene.

### 5.2.2 Self-Supervised Modal Adaptation

In this section, we describe our approach to multimodal fusion using our proposed self-supervised model adaptation (SSMA) framework. Our framework consists of three compo-

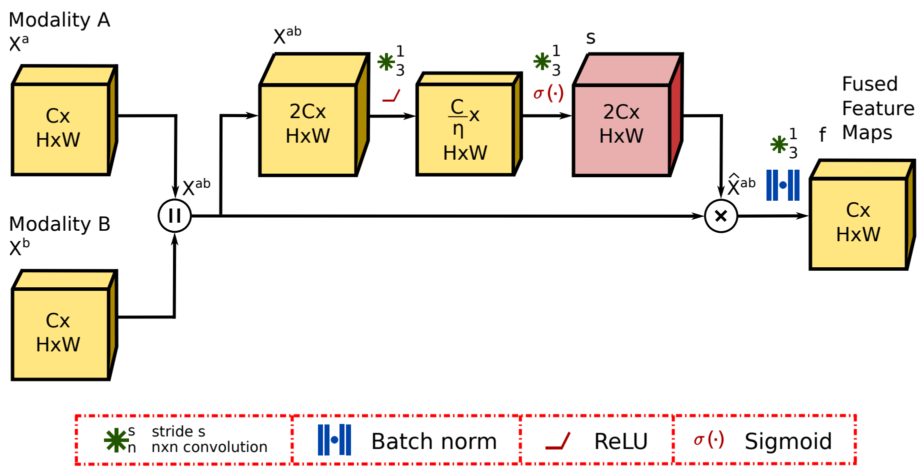


**Figure 5.4:** The topology of our proposed SSMA fusion module that adaptively recalibrates and fuses modality-specific feature maps based on the inputs in order to exploit the more informative features from the modality-specific streams. The symbol $\eta$ denotes the bottleneck compression rate, $\|$ denotes concatenation across the channel dimension and $\times$ denotes the Hadamard product.

nents: a modality-specific encoder for which we adopt the AdapNet++ encoder topology, a decoder built upon the topology our AdapNet++ decoder and our proposed SSMA module for adaptively recalibrating and fusing modality-specific feature maps. In the following section, we first describe the structure of our proposed SSMA module, followed by the entire topology of the multimodal semantic segmentation architecture that incorporates our SMMA module for fusion.

#### 5.2.2.1 SSMA Fusion Module

In order to adaptively recalibrate and fuse feature maps from modality-specific networks, we propose a novel architectural unit called the SSMA module. The goal of the SSMA module is to explicitly model the correlation between the two modality-specific feature maps before fusion so that the network can exploit the complementary features by learning to selectively emphasize more informative features from one modality, while suppressing the less informative features from the other. We construct the topology of the SSMA module in a fully-convolutional fashion which empowers the network with the ability to emphasize features from a modality-specific network for only certain spatial locations or object categories, while emphasizing features from the complementary modality for other locations or object categories. Moreover, the SSMA module dynamically recalibrates the feature maps based on the input scene context.

The structure of the SSMA module is shown in Figure 5.4. Let $\mathbf{X}^a \in \mathbb{R}^{C\times H\times W}$ and $\mathbf{X}^b \in \mathbb{R}^{C\times H\times W}$ denote the modality-specific feature maps from modality $a$ and modality $b$ respectively, where $C$ is the number of feature channels and $H \times W$ is the spatial dimension. First, we concatenate the modality-specific feature maps $\mathbf{X}^a$ and $\mathbf{X}^b$ to yield

$\mathbf{X}^{ab} \in \mathbb{R}^{2 \cdot C \times H \times W}$. We then employ a recalibration technique to adapt the concatenated feature maps before fusion. In order to achieve this, we first pass the concatenated feature map $\mathbf{X}^{ab}$ through a bottleneck consisting of two $3 \times 3$ convolutional layers for dimensionality reduction and to improve the representational capacity of the concatenated features. The first convolution has weights $\mathcal{W}_1 \in \mathbb{R}^{\frac{1}{\eta} \cdot C \times H \times W}$ with a channel reduction ratio $\eta$ and a non-linearity function $\delta(\cdot)$. We use ReLU for the non-linearity, similar to the other activation functions in the encoder and experiment with different reductions ratios in Section 5.3.4.2. Note that we omit the bias term to simplify the notation. The subsequent convolutional layer with weights $\mathcal{W}_2 \in \mathbb{R}^{2 \cdot C \times H \times W}$ increases the dimensionality of the feature channels back to concatenation dimension $2C$ and a sigmoid function $\sigma(\cdot)$ scales the dynamic range of the activations to the $[0, 1]$ interval. This can be represented as

$$\begin{aligned} \mathbf{s} = F_{ssma}(\mathbf{X}^{ab}; \mathcal{W}) &= \sigma\left(g\left(\mathbf{X}^{ab}; \mathcal{W}\right)\right) \\ &= \sigma\left(\mathcal{W}_2 \delta\left(\mathcal{W}_1 \mathbf{X}^{ab}\right)\right). \end{aligned} \tag{5.5}$$

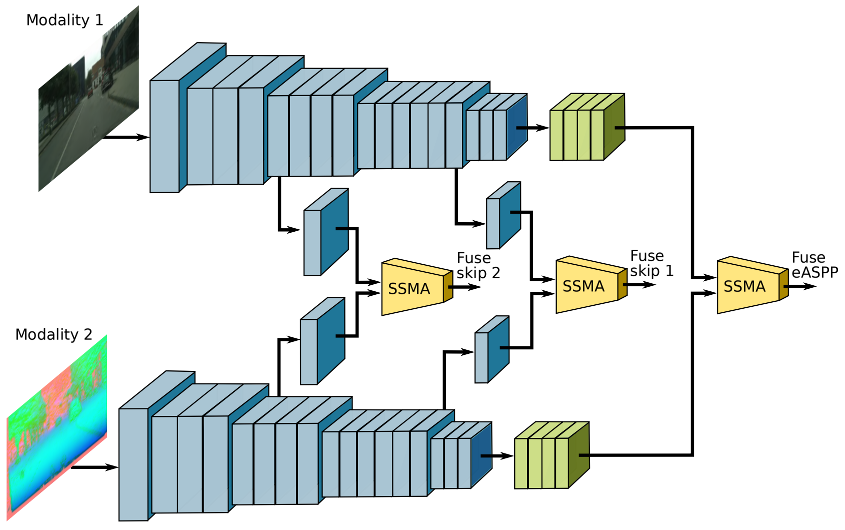
The resulting output $\mathbf{s}$ is used to recalibrate or emphasize/de-emphasize regions in $\mathbf{X}^{ab}$ as

$$\hat{\mathbf{X}}^{ab} = F_{scale}(\mathbf{X}^{ab}; \mathbf{s}) = \mathbf{s} \circ \mathbf{X}^{ab}, \tag{5.6}$$

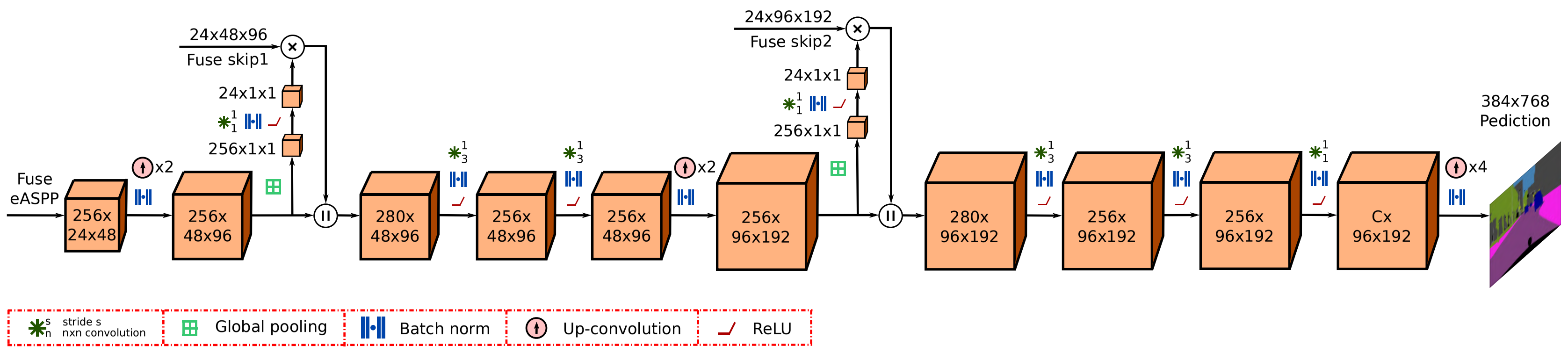
where $F_{scale}(\mathbf{X}^{ab}, \mathbf{s})$ denotes Hadamard product of the feature maps $\mathbf{X}^{ab}$ and the matrix of scalars $\mathbf{s}$ such that each element $x_{c,i,j}$ in $\mathbf{X}^{ab}$ is multiplied with a corresponding activation $s_{c,i,j}$ in $\mathbf{s}$ with $c \in \{1, 2, \ldots, 2C\}$, $i \in \{1, 2, \ldots, H\}$ and $j \in \{1, 2, \ldots, W\}$. The activations $\mathbf{s}$ adapt to the concatenated input feature map $\mathbf{X}^{ab}$, enabling the network to weigh features element-wise spatially and across the channel depth based on the multimodal inputs $I_n$ and $K_n$. With new multimodal inputs, the network dynamically weighs and reweighs the feature maps in order to optimally combine complementary features. Finally, the recalibrated feature maps $\hat{\mathbf{X}}^{ab}$ are passed through a $3 \times 3$ convolution with weights $\mathcal{W}_3 \in \mathbb{R}^{C \times H \times W}$ and a batch normalization layer to reduce the feature channel depth and yield the fused output $\mathbf{f}$ as

$$\mathbf{f} = F_{fused}(\hat{\mathbf{X}}^{ab}; \mathcal{W}) = g(\hat{\mathbf{X}}^{ab}; \mathcal{W}) = \mathcal{W}_3 \hat{\mathbf{X}}^{ab}. \tag{5.7}$$

As described in the following section, we employ our proposed SSMA module to fuse modality-specific feature maps both at intermediate stages of the network and towards the end of the encoder. Although we utilize a bottleneck structure to conserve the number of parameters consumed, further reduction in the parameters can be achieved by replacing the $3 \times 3$ convolution layers with $1 \times 1$ convolutions, which yields comparable performance. We also remark that the SSMA modules can be used for multimodal fusion in other tasks such as image classification or object detection, as well as for fusion of feature maps across tasks in multitask learning.



(a) Encoder Topology



(b) Decoder Topology

**Figure 5.5:** Topology of our modified AdapNet++ encoder and decoder used for multimodal fusion. The encoder employs a late-fusion technique to fuse feature maps from modality-specific streams using our proposed SSMA module. While, the decoder employs our proposed mechanism to better correlate the fused low and mid-level skip refinement features from the encoder that are combined with the high-level decoder features. The symbol || denotes concatenation along the channel dimension and × denotes the Hadamard product.

#### 5.2.2.2 SSMA Fusion Architecture

We propose a framework for multimodal semantic segmentation by reconfiguring of our AdapNet++ architecture and incorporating the proposed SSMA modules. For simplicity, we consider the fusion of two modalities, but the framework can be easily extended to arbitrary number of modalities. The encoder of our multimodal semantic segmentation architecture shown in Figure 5.5 (a) contains two streams, where each stream is based on the AdapNet++ encoder topology described in Chapter 4. Each encoder stream is modality-specific and specializes in a particular sub-space. In order to fuse the feature maps from both streams, we adopt a combination of mid-level and late-fusion strategy in which we fuse the latent representations of both encoders using the SSMA module and pass the fused feature map to the first decoder stage. We denote this as latent SSMA fusion as it takes the output of the eASPP from each modality-specific encoder as input. We set the reduction ratio $\eta = 16$ in the latent SSMA. As the AdapNet++ architecture contains skip connections for high-resolution refinement, we employ an SSMA module at each skip refinement stage after the $1 \times 1$ convolution. As the $1 \times 1$ convolutions reduce the feature channel depth to 24, we only use a reduction ratio $\eta = 6$ in the two skip SSMAs as identified from the ablation experiments presented in Section 5.3.4.2.

In order to upsample the fused predictions, we build upon the AdapNet++ decoder described in Chapter 4. The main stream of our decoder resembles the topology of the decoder in our AdapNet++ architecture consisting of three upsampling stages. The output of the latent SSMA module is fed to the first upsampling stage of the decoder. Following the AdapNet++ topology, the outputs of the skip SSMA modules would be concatenated into the decoder at the second and third upsampling stages (*skip1* after the first deconvolution and *skip2* after the second deconvolution). However, we find that concatenating the fused mid-level features into the decoder does not substantially improve the resolution of the segmentation, as much as in the unimodal AdapNet++ architecture. We hypothesise that directly concatenating the fused mid-level features and fused high-level features causes a feature localization mismatch as each SSMA module adaptively recalibrates at different stages of the network where the resolution of the feature maps and channel depth differ by one half. Moreover, training the fusion network end-to-end from scratch also contributes to this problem as without initializing the encoders with modality-specific pre-trained weights, concatenating the uninitialized mid-level fused encoder feature maps into the decoder does not yield any performance gains, rather it hampers the convergence.

With the goal of mitigating this problem, we propose two strategies. In order to facilitate better fusion, we adopt a multi-stage training protocol where we first initialize each encoder in the fusion architecture with pre-trained weights from the unimodal AdapNet++ model. We describe this procedure in Section 5.3.1.2. Secondly, we propose a mechanism to better correlate the mid-level fused features with the high-level semantic features. We propose to weigh the fused mid-level skip features with the spatially aggregated statistics of the high-level decoder features before the concatenation. Following the notation convention, we define $\mathbf{D} \in \mathbb{R}^{C\times H\times W}$ as the high-level decoder feature map before the skip concatenation stage. A feature statistic $\mathbf{s} \in \mathbb{R}^{C}$ is produced by projecting $\mathbf{D}$ along the spatial dimensions $H \times W$ using a global average pooling layer as

$$s_c = F_{shrink}(d_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} d_c(i,j), \tag{5.8}$$

where $s_c$ represents a statistic or a local descriptor of the $c^{th}$ element of $\mathbf{D}$. We then reduce the number of feature channels in $\mathbf{s}$ using a $1 \times 1$ convolution layer with weights $\mathcal{W}_4 \in \mathbb{R}^{C\times H\times W}$, batch normalization and an ReLU activation function $\delta$ to match the channels of the fused mid-level feature map $\mathbf{f}$, where $\mathbf{f}$ is computed as shown in Eq. (5.7). We can represent resulting output as

$$z = F_{reduce}(\mathbf{s}; \mathcal{W}) = \delta(\mathcal{W}_4 \mathbf{s}). \tag{5.9}$$

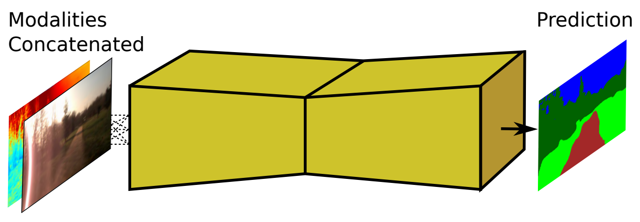
Finally, we weigh the fused mid-level feature map $\mathbf{f}$ with the reduced aggregated descriptors $\mathbf{z}$ using channel-wise multiplication as

$$\hat{\mathbf{f}} = F_{loc}(\mathbf{f}_c; z_c) = (z_1\mathbf{f}_1, z_2\mathbf{f}_2, \ldots, z_c\mathbf{f}_c)\,. \tag{5.10}$$

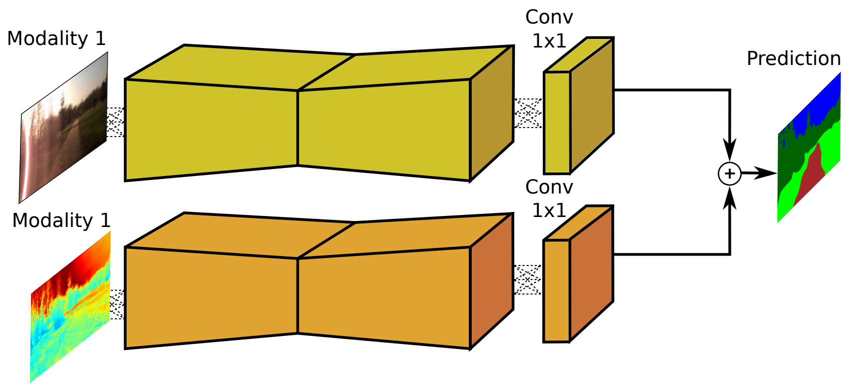
As shown in Figure 5.5 (b), we employ the aforementioned mechanism on the fused feature maps from skip1 SSMA as well as skip2 SSMA and concatenate their outputs with the decoder feature maps at the second and third upsampling stages respectively. We find that this mechanism guides the fusion of mid-level skip refinement features with the high-level decoder feature more effectively than direct concatenation and yields a notable improvement in the resolution of the segmentation output.

## 5.2.3 Baseline Fusion Architectures

In addition to comparing against existing multimodal semantic segmentation architectures, we also implement four different baseline fusion architectures. The first baseline fusion strategy that we employ is the early-fusion approach as depicted in Figure 5.6 (a), where the different modalities are first concatenated channel-wise to yield a four or a six-channel image. Subsequently, the concatenated modalities are then used as input to standard unimodal semantic segmentation architectures for end-to-end feature learning. The early-fusion approach is the most widely employed fusion method as existing semantic segmentation architectures can be leveraged for this purpose and its implementation is relatively straightforward. Early-fusion aims to extract the joint representation directly from the raw images. Fusing at an early stage amplifies the common discriminant information in both modalities but also simultaneously accumulates the noise in both modalities. Learning a good joint representation throughout the network using this configuration is difficult



(a) Early fusion



(b) Late fusion

**Figure 5.6:** Depiction of the early and late-fusion approaches for multimodal semantic segmentation. In early-fusion, the modalities are concatenated to obtain a four or a six-channel image and then fed to the network as input for end-to-end semantic segmentation. While, in the late-fusion approach, individual network streams that take different modalities as input are employed, followed by combining the features towards the end of the network using a $1 \times 1$ convolution and element-wise addition.

due to significantly different data formats and distinct distributions between modalities. Another disadvantage of this method is that pre-trained weights from networks trained on large image segmentation datasets cannot be leveraged for initialization using transfer learning.

We employ the late-fusion approach for the second baseline. The topology of the late-fusion shown in Figure 5.6 (b) consists of individual modality-specific network streams that are first trained end-to-end using their respective modalities. Followed by initializing the joint model with pre-trained weights from the previous step and adding a $1 \times 1$ convolution layer at the end of the streams before the element-wise addition of feature maps from both modality-specific streams for the fusion. While training the joint model, the weights of the individual streams before the $1 \times 1$ convolution layer are often kept fixed. This enables them to retain their modality-specific low and mid-level features while focusing only on learning the fusion of the high-level semantically mature features. The $1 \times 1$ convolution layer is added so that the networks learn to adapt their individual high-level representations before the fusion. The late-fusion approach is a popular fusion baseline as it builds upon existing semantic segmentation architectures and therefore the model can be initialized with pre-trained weights from networks trained on large semantic segmentation datasets. More importantly, this configuration enables the network to exploit the high-level semantically rich information for the fusion, therefore complementary information can be leveraged based on the features describing the semantic object classes in the dataset. The performance of late-fusion approaches is largely dependent on the correlation mechanism that is employed at the end of the individual modality-specific network streams for the fusion.

For the third baseline, we employ the averaging technique in which individual semantic segmentation architectures are first trained on a specific modality and then the prediction probabilities of each of the networks are class-wise averaged before computing the argmax. Averaging the class-wise probabilities gives us a measure of the consensus and certainty among the modality-specific networks. For instance, if both networks are highly confident in their predictions for a specific object class, then the consensus value is highest for this particular class leading to its subsequent section by the argmax. On the other hand, if one or both of the networks produce a low score for a particular object class, taking the average of their probabilities smoothens the score to reflect the joint confidence. Similarly, for the last fusion baseline, as opposed to computing the average of the class-wise prediction probabilities, we compute the maximum of the class-wise prediction probabilities across the different modality-specific networks before computing the argmax. This enables us to select the most certain predictions across the modality-specific networks for a particular object class.

## 5.3 Experimental Evaluation

In this section, we first describe the procedure that we employ for training our multimodal semantic segmentation architectures that incorporate our proposed CMoDE and SSMA fusion modules in Section 5.3.1, followed by detailed benchmarking results using the various modalities and spectra contained in the datasets in Section 5.3.2 and comprehensive ablation studies that describe the various architectural decisions that we made while designing the fusion module topologies in Section 5.3.4. In Section 5.3.5, we present extensive qualitative evaluations on all the datasets that we benchmark on and finally in Section 5.3.7, we present the results from our autonomous navigation experiments using only the multimodal semantic segmentation for perception.

For the experiments presented in this chapter, we benchmark on the five multimodal indoor and outdoor datasets described in Chapter 4. Namely, Cityscapes [143], Synthia [144], SUN RGB-D [145], ScanNet [146] and Freiburg Forest [50] datasets. We use the TensorFlow [159] deep learning library for the implementations and all the experiments were carried out on a system with an Intel Xeon E5, 2.4 GHz and an NVIDIA TITAN X GPU. We primarily use the standard Jaccard Index, also known as the intersection-over-union

(IoU) metric to quantify the performance. It can be computed for each object class as $IoU = TP/(TP + FP + FN)$, where $TP, FP$ and $FN$ correspond to true positives, false positives and false negatives respectively. We also report the mean intersection-over-union (mIoU) metric, pixel-wise accuracy (Acc) and average precision (AP) for all the models. We made our models publicly available at `http://deepscene.cs.uni-freiburg.de`.

### 5.3.1 Network Training

We train our networks with an input image resolution of $768 \times 384$ pixels, therefore we use bilinear interpolation for resizing the RGB images and the nearest-neighbour interpolation for resizing the other modalities as well as the groundtruth labels. We employ a multi-stage training protocol to effectively train our multimodal semantic segmentation architectures. We first train each modality-specific Adapnet++ model individually using the training protocol described in Section 4.4.3 of Chapter 4. Subsequently, in the second stage, we leverage transfer learning to train the joint fusion model in either the CMoDE or the SSMA framework. We describe the procedure that we employ in the second stage and the parameter settings for training the multimodal fusion in the following sections.

#### 5.3.1.1 CMoDE Training

In the second stage, we initialize the modality-specific encoders and decoders using weights from the previous stage, while we initialize the class-specific CMoDE and the fused convolution layer using the He initialization [93] scheme. We keep the weights of the modality-specific encoders and decoders fixed, which forces the CMoDE to use the representations learned by the individual streams from the first stage, while exploiting discriminative complementary features from different modalities at the high-level in the second stage. We use the Adam solver for optimization with $\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-10}$. We train the second stage of the CMoDE model for a maximum of 50,000 iterations using an initial learning rate of $\lambda_0 = 10^{-5}$ with a mini-batch size of 8 and a dropout probability of 0.5.

#### 5.3.1.2 SSMA Training

In order to train the SSMA fusion model, we initialize only the modality-specific encoders using weights from the previous stage and similar to the CMoDE, we use He initialization [93] scheme for the other layers. We then set the learning rate of the encoder layers to $\lambda_0 = 10^{-4}$ and the decoder layers to $\lambda_0 = 10^{-3}$, and train the fusion model with a mini-batch of 7 for a maximum of 100,000 iterations using the Adam solver with $\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-10}$. This enables the SSMA modules to learn the optimal combination of multimodal feature maps from the well trained encoders, while slowly adapting the encoder weights to improve the fusion. In the final stage, we fix the weights of the encoder layers while only training the decoder and the SSMA modules with a learning rate of $\lambda_0 = 10^{-5}$ and a mini-batch size of 12 for 50,000 iterations. This enables us to train the network with a larger batch size, while focusing more on the upsampling stages to yield the high-resolution segmentation output.

## 5.3.2 Comparison with the State-of-the-Art

In this section, we present comprehensive results on the performance of our multimodal fusion architectures that incorporate our proposed fusion modules. We compare the performance with state-of-the-art multimodal fusion methods, namely, LFC [50] and FuseNet [162], in addition to the four baseline fusion architectures that we described in Section 5.2.3. Note that the FuseNet architecture was designed for RGB-D fusion and uses one-channel depth images in the modality-specific encoder. Therefore, employing this approach for RGB-HHA or RGB-EVI fusion is infeasible as the HHA and EVI modalities are three-channel images. Furthermore, we compare the performance of multimodal semantic segmentation against the unimodal AdapNet++ models trained individually on the different modalities and spectra contained in the datasets. We also report the performance of our multimodal SSMA model evaluated with left-right flips as well as multiscale testing and denote the resulting model as SSMA_msf in our experiments.

In Table 5.1, we show the results on the Cityscapes validation set considering visual RGB images, depth and the HHA encoding of the depth as modalities for the fusion. As hypothesised, the visual RGB images perform the best among the other modalities achieving a mIoU of 80.80%. This is especially observed in outdoor scene understanding datasets containing stereo depth images that quickly degrade the information contained, with increasing distances from the camera. Among the baseline fusion approaches, Stacking achieves the highest performance for both RGB-D and RGB-HHA fusion. However, the performance of Stacking is still lower than the unimodal visual RGB segmentation model. This can be attributed to the fact that the baseline approaches are not able to exploit the complementary features from the modalities due to the naive multimodal fusion. Our proposed CMoDE fusion with RGB-HHA outperforms all the state-of-the-art approaches as well as the baselines, further surpassing the performance of the unimodal segmentation models. While, our proposed SSMA fusion model for RGB-HHA fusion achieves a mIoU of 82.64% outperforming all the other approaches and setting the new state-of-the-art on this benchmark. The SSMA_msf model using RGB-HHA fusion further improves upon the performance of the SMMA model by 1.3%. As the Cityscapes dataset does not contain harsh environments, the improvement that can be achieved using fusion is limited to scenes that contain inconspicuous object classes or mismatched relationship. However, the additional robustness that it demonstrates due to multimodal fusion is still notable as shown in the qualitative results in Section 5.3.5. We refer the reader to Appendix A.1 for the detailed comparisons of the individual class IoU scores for the Cityscapes dataset.

**Table 5.1:** Comparison of multimodal fusion approaches on the Cityscapes dataset.

| Network | Approach | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|
| RGB | Unimodal | 80.80 | 96.04 | 90.97 |
| Depth | Unimodal | 66.36 | 91.21 | 80.23 |
| HHA | Unimodal | 67.66 | 91.66 | 81.81 |
| RGB-D | Average | 78.84 | 95.58 | 90.49 |
| | Maximum | 78.81 | 95.58 | 90.37 |
| | Stacking | 80.21 | 95.96 | 90.05 |
| | Late Fusion | 78.75 | 95.57 | 90.48 |
| | LFC [50] | 81.04 | 96.11 | 91.10 |
| | FuseNet [162] | 78.20 | 95.31 | 90.18 |
| | CMoDE (Ours) | 81.33 | 96.12 | 90.29 |
| | SSMA (Ours) | 82.29 | 96.36 | 90.77 |
| | SSMA_msf (Ours) | **83.44** | **96.59** | **92.21** |
| RGB-HHA | Average | 79.44 | 95.56 | 90.27 |
| | Maximum | 79.40 | 95.55 | 90.09 |
| | Stacking | 80.62 | 96.01 | 90.09 |
| | Late Fusion | 79.01 | 95.49 | 90.25 |
| | LFC [50] | 81.13 | 96.14 | 91.32 |
| | CMoDE (Ours) | 81.42 | 96.12 | 90.29 |
| | SSMA (Ours) | 82.64 | 96.41 | 90.65 |
| | SSMA_msf (Ours) | **83.94** | **96.68** | **91.99** |

Additionally, the benchmarking results on the Cityscapes test set is shown in Table 4.3. The results demonstrate that our SSMA fusion architecture with the AdapNet++ network backbone achieves a comparable performance as the top performing DPC [172], and DRN [173] architectures, while outperforming the other networks on the leaderboard.

We benchmark on the Synthia dataset to demonstrate the utility of fusion when both modalities contain rich information. It consists of scenes with adverse perceptual conditions including rain, snow, fog and night, therefore the benefit of multimodal fusion for outdoor environments is most evident on this dataset. As the Synthia dataset does not provide camera calibration parameters, it is infeasible to compute the HHA encoding, therefore we benchmark using visual RGB and depth images. Results from benchmarking on this dataset are shown in Table 5.2. Due to the high-resolution depth information, the unimodal depth model achieves a mIoU of 87.87%, outperforming segmentation using visual RGB images by 1.17%. This demonstrates that accurate segmentation can be obtained using only depth images as input, provided that the depth sensor gives accurate long range information. Our proposed CMoDE fusion approach using RGB-D images outperforms the baseline

**Table 5.2:** Comparison of multimodal fusion approaches on the Synthia dataset.

| Network | Approach | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|
| RGB | Unimodal | 86.70 | 97.18 | 93.17 |
| Depth | Unimodal | 87.87 | 97.78 | 94.23 |
| RGB-D | Average | 89.22 | 98.03 | 95.04 |
| | Maximum | 89.13 | 98.01 | 94.97 |
| | Stacking | 88.95 | 98.03 | 94.41 |
| | Late Fusion | 89.13 | 98.01 | 94.66 |
| | LFC [50] | 89.48 | 98.09 | 94.96 |
| | FuseNet [162] | 86.10 | 97.09 | 93.08 |
| | CMoDE (Ours) | 89.57 | 98.13 | 94.58 |
| | SSMA (Ours) | 91.25 | 98.48 | 95.68 |
| | SSMA_msf (Ours) | **92.10** | **98.64** | **96.37** |

fusion approaches and the state-of-the-art techniques achieving a mIoU of 89.57%, while exceeding the performance of the unimodal depth model by 1.7%. On the other hand, our proposed SSMA fusion architecture demonstrates the state-of-the-art performance of 91.25% and further improves the mIoU to 92.10% using the SSMA_msf model. This amounts to a large improvement of 5.4% over the best performing unimodal segmentation model. It can also be observed that the other metrics such as the pixel accuracy and average precision show similar improvement. We refer the reader to Appendix A.2 for the detailed comparisons of the individual class IoU scores for the Synthia dataset.

One of the main motivations to benchmark on this dataset is to evaluate our fusion model in diverse scenes with adverse perpetual conditions. For this experiment, we trained our state-of-the-art SSMA fusion model on the Synthia-Rand-Cityscapes training set and evaluated the performance on each of the conditions contained in the Synthia-Sequences dataset. The Synthia-Sequences dataset contains individual video sequences in different conditions such as *Summer, Fall, Winter, Spring, Dawn, Sunset, Night, Rain, Soft Rain, Fog, Night Rain* and *Winter Night*. Results from this experiment are shown in Figure 5.7. The unimodal visual RGB model achieves an overall mIoU score of 49.27% $\pm$ 4.22% across the 12 sequences. While, the model trained on the depth maps achieves a mIoU score of 67.07% $\pm$ 1.16%, thereby substantially outperforming the model trained using visual RGB images.

As Synthia is a synthetic dataset captured in a hyperrealistic simulation environment, the depth maps provided are accurate and dense even for structures that are several hundreds of meters away from the camera. Therefore, this enables the unimodal depth model to learn representations that accurately encode the structure of the scene and these structural representations are proven to be invariant to the changes in perceptual conditions. It

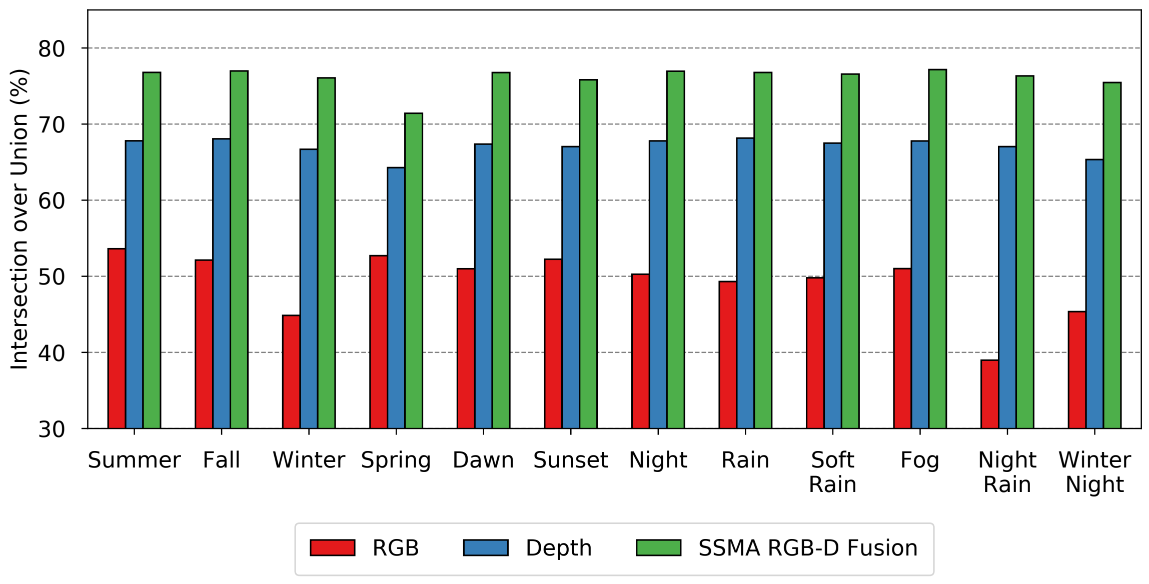


**Figure 5.7:** Evaluation of our proposed SMMA fusion technique on the Synthia-Sequences dataset containing a variety of seasons and weather conditions. We use the models trained on the Synthia-Rand-Cityscapes dataset and only test on the individual conditions in the Synthia-Sequences dataset to quantify its robustness. Our model that performs RGB-D fusion consistently outperforms the unimodal models which can be more prominently seen qualitatively in Figure 5.14.

can also be observed that the unimodal depth model performs consistently well in all the conditions with a variance of 1.36%, demonstrating its generalization to different weather and seasonal changes. However, the visual RGB model with a variance of 17.79% performs inconsistently across different conditions. Nevertheless, we observe that our RGB-D SSMA fusion model outperforms the unimodal visual RGB model by achieving a mIoU score of 76.51% $\pm$ 0.53% across the 12 conditions, accounting to a significant improvement of 27.24%. Moreover, the SSMA fusion model has a variance of 0.28%, demonstrating better generalization abilities across varying adverse perceptual conditions.

We benchmark on the indoor SUN RGB-D dataset which demonstrates a different set of challenges than the outdoor datasets. The improvement due to the multimodal fusion is more evident in indoor scenes as the images are often captured in smaller confined spaces with several cluttered objects and the depth modality provides valuable structural information that can be exploited. Results from multimodal RGB-D and RGB-HHA fusion is shown in Table 5.3. Among the unimodal models, semantic segmentation using visual RGB images yields the highest mIoU of 38.40%. The model trained on depth images performs 4.13% lower than the visual RGB model. This can be attributed to the fact that the depth images are extremely noisy with numerous missing depth values in the SUN RGB-D dataset. Nevertheless, our CMoDE approach outperforms the state-of-the-art fusion models as well the unimodal segmentation models. The CMoDE model using RGB-HHA fusion achieves a mIoU of 42.55% which is an improvement of 4.15% over the unimodal visual

**Table 5.3:** Comparison of multimodal fusion approaches on the SUN RGB-D dataset.

| Network | Approach | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|
| RGB | Unimodal | 38.40 | 76.90 | 62.78 |
| Depth | Unimodal | 34.27 | 73.83 | 74.39 |
| HHA | Unimodal | 34.59 | 74.39 | 57.18 |
| RGB-D | Average | 40.70 | 78.58 | 64.54 |
| | Maximum | 40.58 | 78.50 | 64.04 |
| | Stacking | 36.48 | 76.68 | 57.92 |
| | Late Fusion | 41.68 | 79.27 | 66.63 |
| | LFC [50] | 41.82 | 79.36 | 66.75 |
| | FuseNet [162] | 37.41 | 76.37 | 61.58 |
| | CMoDE (Ours) | 41.87 | 79.84 | 66.81 |
| | SSMA (Ours) | 43.90 | 80.16 | 66.11 |
| | SSMA_msf (Ours) | **44.52** | **80.67** | **67.92** |
| RGB-HHA | Average | 41.01 | 78.54 | 64.93 |
| | Maximum | 40.91 | 78.49 | 64.78 |
| | Stacking | 37.49 | 76.42 | 57.88 |
| | Late Fusion | 41.91 | 79.49 | 67.31 |
| | LFC [50] | 42.42 | 79.55 | 67.41 |
| | CMoDE (Ours) | 42.55 | 79.94 | 65.38 |
| | SSMA (Ours) | 44.43 | 80.21 | 64.94 |
| | SSMA_msf (Ours) | **45.73** | **80.97** | **67.82** |

RGB model. Furthermore, our proposed SSMA approach using multimodal RGB-HHA fusion achieves the state-of-the-art performance with a mIoU of 44.43%, constituting to a substantial improvement of 6.03% over the unimodal visual RGB model. Moreover, our SSMA_msf model further improves upon the mIoU by 1.3%. Similar to the performance observed in other datasets, the fusion of RGB-HHA yields a higher mIoU than the RGB-D fusion, corroborating the fact that CNNs learn more effectively from the HHA encoding but with a small additional preprocessing time. Additionally, for the detailed comparisons of the individual class IoU scores for the SUN RGB-D dataset, we refer the reader to Appendices A.3 and A.4 for the RGB-D and RGB-HHA fusion correspondingly.

We present results on the ScanNet validation set in Table 5.4. ScanNet is the largest indoor RGB-D dataset to date with over 1513 different scenes and 2.5 M views. Unlike the SUN RGB-D dataset, ScanNet contains depth maps of better quality and with lesser number of missing depth values. The unimodal visual RGB model achieves a mIoU of 52.68% with an accuracy of 78.64%, while the unimodal depth model achieves an mIoU of 54.00% with an accuracy of 79.67%. For multimodal fusion, our proposed CMoDE

**Table 5.4:** Comparison of multimodal fusion approaches on the ScanNet dataset.

| Network | Approach | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|
| RGB | Unimodal | 52.68 | 78.64 | 76.18 |
| Depth | Unimodal | 54.00 | 79.67 | 66.22 |
| HHA | Unimodal | 53.07 | 79.44 | 77.38 |
| RGB-D | Average | 57.15 | 81.59 | 79.93 |
| | Maximum | 56.90 | 81.47 | 79.56 |
| | Stacking | 52.77 | 78.95 | 74.66 |
| | Late Fusion | 58.18 | 81.93 | 81.38 |
| | LFC [50] | 60.50 | 83.08 | 78.84 |
| | FuseNet [162] | 49.08 | 76.98 | 75.17 |
| | CMoDE (Ours) | 61.74 | 83.83 | 79.91 |
| | SSMA (Ours) | 64.19 | 85.68 | 79.71 |
| | SSMA_msf (Ours) | **65.67** | **85.91** | **80.23** |
| RGB-HHA | Average | 56.35 | 81.01 | 80.63 |
| | Maximum | 56.21 | 80.97 | 80.28 |
| | Stacking | 53.51 | 79.20 | 75.36 |
| | Late Fusion | 58.25 | 81.91 | 80.93 |
| | LFC [50] | 60.11 | 82.88 | 79.03 |
| | CMoDE (Ours) | 62.28 | 83.44 | 79.65 |
| | SSMA (Ours) | 64.54 | 86.10 | 81.95 |
| | SSMA_msf (Ours) | **65.90** | **86.48** | **82.28** |

model using RGB-HHA outperforms all the state-of-the-art architectures as well as the fusion baselines, achieving a mIoU of 62.28%. While our proposed SSMA model using RGB-HHA fusion achieves a mIoU of 64.54% and sets the new state-of-the-art on this benchmark. This accounts to a significant improvement of 11.86% over the unimodal visual RGB model. Moreover the SSMA_msf model additionally improves the performance to 65.90%, which is an improvement of 13.22% over the visual RGB model. To the best of our knowledge, this is the largest improvement due to multimodal fusion that has been reported thus far. We refer the reader to Appendix A.5 for the detailed comparisons of the individual class IoU scores for the ScanNet dataset. An interesting observation that can be made from the results on the SUN RGB-D and ScanNet datasets is that the lowest multimodal fusion performance is obtained using the Stacking approach, reaffirming our hypothesis that fusing semantically more mature features enables the model to exploit complementary relationship from different modalities more effectively. We also benchmark on the ScanNet test set and report the results in Table 4.7. Our proposed SSMA fusion architecture with the AdapNet++ network backbone sets the new state-of-the-art on the

**Table 5.5:** Comparison of multimodal fusion approaches on the Freiburg Forest dataset.

| Network | Approach | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|
| RGB | Unimodal | 83.09 | 95.15 | 89.83 |
| Depth | Unimodal | 73.93 | 91.42 | 85.36 |
| EVI | Unimodal | 80.96 | 94.20 | 88.88 |
| RGB-D | Average | 79.51 | 92.87 | 90.99 |
| | Maximum | 81.62 | 93.93 | 90.87 |
| | Stacking | 83.13 | 95.19 | 89.95 |
| | Late Fusion | 82.11 | 93.95 | 90.85 |
| | LFC [50] | 82.53 | 94.99 | 90.96 |
| | FuseNet [162] | 81.84 | 93.97 | 90.93 |
| | CMoDE (Ours) | 83.21 | 95.19 | 90.19 |
| | SSMA (Ours) | 83.81 | 95.62 | 92.78 |
| | SSMA_msf (Ours) | **83.99** | **95.70** | **93.08** |
| RGB-EVI | Average | 83.00 | 95.10 | 90.19 |
| | Maximum | 83.00 | 95.10 | 90.17 |
| | Stacking | 83.18 | 95.21 | 90.11 |
| | Late Fusion | 82.80 | 95.01 | 90.07 |
| | LFC [50] | 83.00 | 95.13 | 90.28 |
| | CMoDE (Ours) | 83.31 | 95.22 | 90.19 |
| | SSMA (Ours) | 83.90 | 95.56 | 92.28 |
| | SSMA_msf (Ours) | **84.18** | **95.64** | **92.60** |

ScanNet benchmark.

Finally, we present benchmarking results on the Freiburg Forest dataset that contains three inherently different modalities including visual RGB images, Depth data and EVI. EVI or Enhanced Vegetation Index was designed to enhance the vegetation signal in high biomass regions and it is computed from the information contained in three bands, namely, Near-InfraRed, Red and Blue channels [203]. As this dataset contains scenes in unstructured forested environments, EVI provides valuable information to discern between inconspicuous classes such as *vegetation* and *grass*. Table 5.5 shows the results on this dataset for multimodal fusion of RGB-D and RGB-EVI. For unimodal segmentation, the RGB model yields the highest performance, closely followed by the model trained on EVI images. For multimodal semantic segmentation, our proposed CMoDE fusion approach outperforms the other state-of-the-art fusion techniques achieving a mIoU of 83.31% for RGB-EVI fusion. While, our SSMA fusion model trained on RGB-EVI yields the highest mIoU of 83.90% setting the new state-of-the-art on this benchmark. Moreover, our SMMA_msf model further improves upon the performance and achieves a mIoU of

84.18%. Additionally, we refer the reader to Appendix A.6 for the detailed comparisons of the individual class IoU scores for the Freiburg Forest dataset..

### 5.3.3 Multimodal Fusion Discussion

To summarize, the models trained on visual RGB images perform the best in comparison to unimodal segmentation with other modalities and spectra. However, when the depth data is less noisy and the environment is confined to an indoor space, the model trained on depth or HHA-encoded depth outperforms visual RGB models. Among the multimodal fusion baselines, Late Fusion and Stacking, each perform well in different environments. Stacking performs better in outdoor environments in the absence of challenging perceptual conditions, while late-fusion performs better in indoor environments and outdoors when the noise in the modalities is uncorrelated. This can be attributed to the fact that the late-fusion method fuses semantically mature representations and effectively exploits complementary features. Therefore, in indoor environments, modalities such as depth maps from stereo cameras are less noisy than in outdoors and as the environment is confined, all the objects in the scene are well represented with dense depth values. This enables the late-fusion architecture to leverage semantically rich complementary representations for fusion. However in outdoor environments, depth values are very noisy and no information is present for objects at far away distances. Therefore, the semantic representations from the depth stream are considerably less informative for certain parts of the scene which does not allow the late-fusion network to fully exploit the complementary features and hence it does not provide significant gains. On the other hand, in indoor environments or in synthetic scenes outdoors where the depth modality is dense and rich with information, late-fusion significantly outperforms the stacking approach.

Our proposed CMoDE fusion approach outperforms existing state-of-the-art multimodal semantic segmentation networks in each of the diverse environments. To recapitulate, CMoDE employs a class-wise probabilistic late-fusion technique that adaptively weighs the modality-specific decoder features based on the scene condition. Moreover, our proposed SSMA fusion technique further outperforms CMoDE in all the datasets and sets the new state-of-the-art in multimodal semantic segmentation. This demonstrates that fusion of modalities is an inherently complex problem that depends on several factors such as the object class, the spatial location of the object and the environmental scene context. Our proposed SSMA fusion approach dynamically adapts the fusion of both mid-level and high-level semantically mature representations based on the aforementioned factors, thereby enabling our model to effectively exploit the complementary relationship between the modalities. Moreover, as the dynamicity is learned in a self-supervised fashion, it efficiently generalizes to different diverse environments, perceptual conditions and types of modalities employed for fusion.

**Table 5.6:** Effect on varying the number of output units $\gamma_1$ in the inner-product layer *ip1* of the Adaptive Gating Network (AGN). The performance is shown on the Cityscapes dataset for multimodal RGB-HHA fusion.

| No. of units $\gamma_1$ | 64 | 128 | 256 | 384 | 512 |
|---|---|---|---|---|---|
| mIoU (%) | 80.94 | **81.42** | 81.41 | 81.17 | 81.04 |

### 5.3.4 Ablation Study

In this section, we study the influence of the various contributions that we make in our proposed CMoDE and SSMA techniques, for more effective multimodal fusion. Specifically, we study the influence of the number of output units in the inner-product layer of the AGN in CMoDE module, followed by evaluations of the SSMA fusion configuration by comparing the performance of fusion at different intermediate network stages. We then evaluate the utility of our proposed channel attention scheme for better correlation of mid-level encoder and high-level decoder features in the SSMA architecture. Subsequently, we experiment with different SSMA bottleneck downsampling rates and qualitatively analyze the convolution activation maps of our SSMA fusion model at various intermediate network stages to study the effect of multimodal fusion on the learned network representations.

#### 5.3.4.1 Influence of Inner-product Output Units in AGN of CMoDE

Our proposed CMoDE fusion module contains an Adaptive Gating Network (AGN) for each object class present in the dataset. Each AGN has two inner-product layers, *ip1* for learning non-linear combinations of features from the convolution layer *conv1* and *ip2* to reduce the dimensionality to the number modality-specific streams. The number of output units $\gamma_1$ in *ip1* plays an important role in performance of the CMoDE fusion as well as the scalability of the CMoDE to the number of object categories being classified. If the number of output units in *ip1* increases by $u$, then the number of parameters in the CMoDE increases by $C \times u$, which can quickly explode while training on datasets with a large number of object classes. Therefore, we experiment with varying the number of output units in *ip1* and present the results for RGB-HHA fusion on the Cityscapes dataset in Table 5.6. The network achieves the highest performance with $\gamma_1$ = 128. Although we also observe that setting $\gamma_1$ = 256 yields a comparable performance, it substantially increases the number of parameters in the CMoDE module. Therefore, for all the experiments, we set the number of output units $\gamma_1$ = 128 in the inner-product layer *ip1* of the AGN.

**Table 5.7:** Effect of the various contributions that we proposed for multimodal fusion in the SMMA architecture. The performance is shown for RGB-HHA fusion on the Cityscapes dataset.

| Model | SSMA Fusion | | | mIoU | Acc. | AP |
|---|---|---|---|---|---|---|
| | ASPP | Skip | Ch. Agg. | (%) | (%) | (%) |
| F0 | - | - | - | 80.77 | 96.04 | 90.97 |
| F1 | ✓ | - | - | 81.55 | 96.19 | 91.15 |
| F2 | ✓ | ✓ | - | 81.75 | 96.25 | 91.09 |
| F3 | ✓ | ✓ | ✓ | **82.64** | **96.41** | **90.65** |

**Table 5.8:** Effect of varying the SSMA bottleneck downsampling rate $\eta$ on the RGB-HHA fusion performance for the Cityscapes dataset.

| Encoder | Skip | mIoU | Acc. | AP |
|---|---|---|---|---|
| SSMA | SSMA | (%) | (%) | (%) |
| $\eta_{enc} = 2$ | $\eta_{skip} = 2$ | 82.15 | 96.32 | 91.17 |
| $\eta_{enc} = 4$ | $\eta_{skip} = 4$ | 82.11 | 96.27 | 91.34 |
| $\eta_{enc} = 8$ | $\eta_{skip} = 4$ | 82.21 | 96.32 | 91.61 |
| $\eta_{enc} = 16$ | $\eta_{skip} = 4$ | 82.25 | 96.31 | 91.12 |
| $\eta_{enc} = 16$ | $\eta_{skip} = 6$ | **82.64** | **96.41** | **90.65** |

#### 5.3.4.2 Detailed Study on the SSMA Fusion Architecture

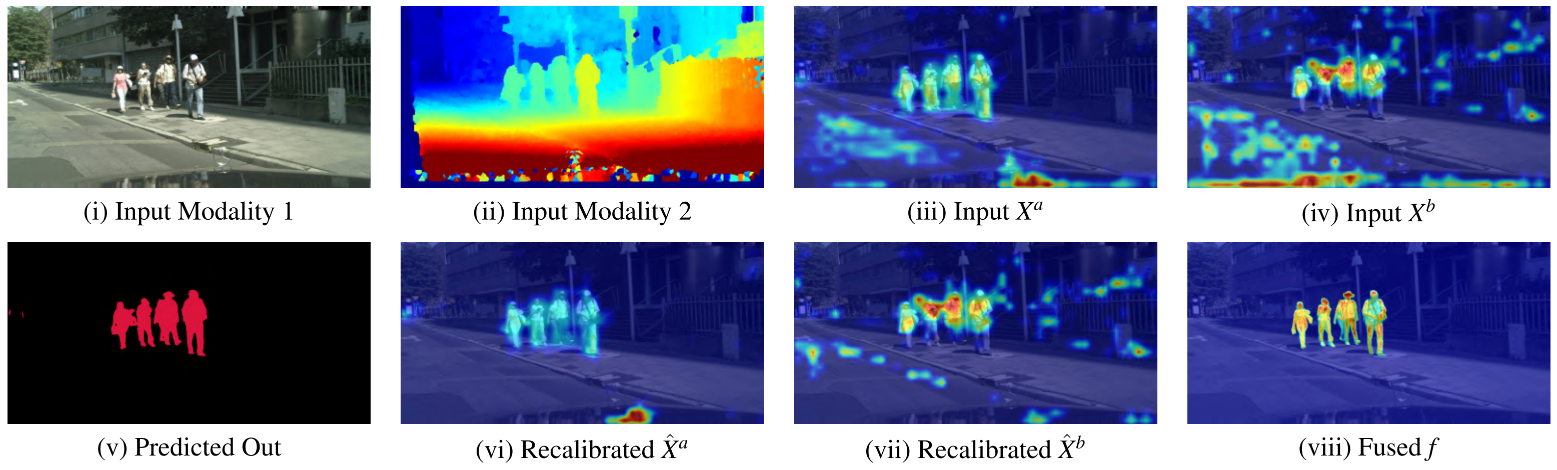
In our proposed multimodal SSMA fusion architecture, we employ a combination of both mid-level fusion and late-fusion. In order to evaluate the influence of fusion at each of these network stages, we present experimental comparisons in Table 5.7. We denote the unimodal model without the SSMA fusion as F0. First, we employ the main SSMA fusion module at the end of the two modality-specific encoders, after the eASPPs and we denote this model as F1. The F1 model achieves a mIoU of 81.55%, which constitutes to an improvement of 0.78% over the unimodal F0 model. We then employ an SSMA module at each skip refinement stage to fuse the mid-level skip features from each modality-specific encoder stream. The fused skip features are then integrated into the decoder for refinement of high-level decoder features. The F2 model that performs multimodal fusion at both stages, yields a mIoU of 81.75%, which is not a significant improvement while compared to the improvement that we achieve in fusion of the mid-level features into the decoder in our unimodal AdapNet++ architecture. As described in Section 5.2.2.2, we hypothesize that this occurs due to the fact that the mid-level representations learned by the network do not align across different modality-specific encoder streams. Therefore, we employ our proposed channel attention mechanism to better correlate these features using the spatially aggregated statistics of the high-level decoder features. The model that incorporates this proposed attention mechanism achieves a mIoU of 82.64%, which is an improvement of 1.09%, while only an improvement of 0.2% was achieved without the channel attention mechanism. Note that the increase in quantitative performance due to multimodal fusion is more apparent in the indoor or synthetic datasets where the depth modality is more informative as shown in Section 5.3.2.

The proposed SSMA fusion module has a bottleneck structure in which the middle convolution layer downsamples the number of feature channels according to a rate $\eta$ as described in Section 5.2.2.1. As we perform multimodal fusion both at the mid-level and at the end of the encoder section, we have to estimate the downsampling rates individually for each of the SSMA modules. We start by using values from a geometric sequence for the main encoder SSMA downsampling rate $\eta_{enc}$ and correspondingly vary the values for the skip SSMA downsampling rates $\eta_{skip}$. Results from this experiment shown in Table 5.8 demonstrates that the best performance is obtained for $\eta_{enc} = 16$ and $\eta_{skip} = 6$ which also increases the parameter efficiency compared to lower downsampling rates.

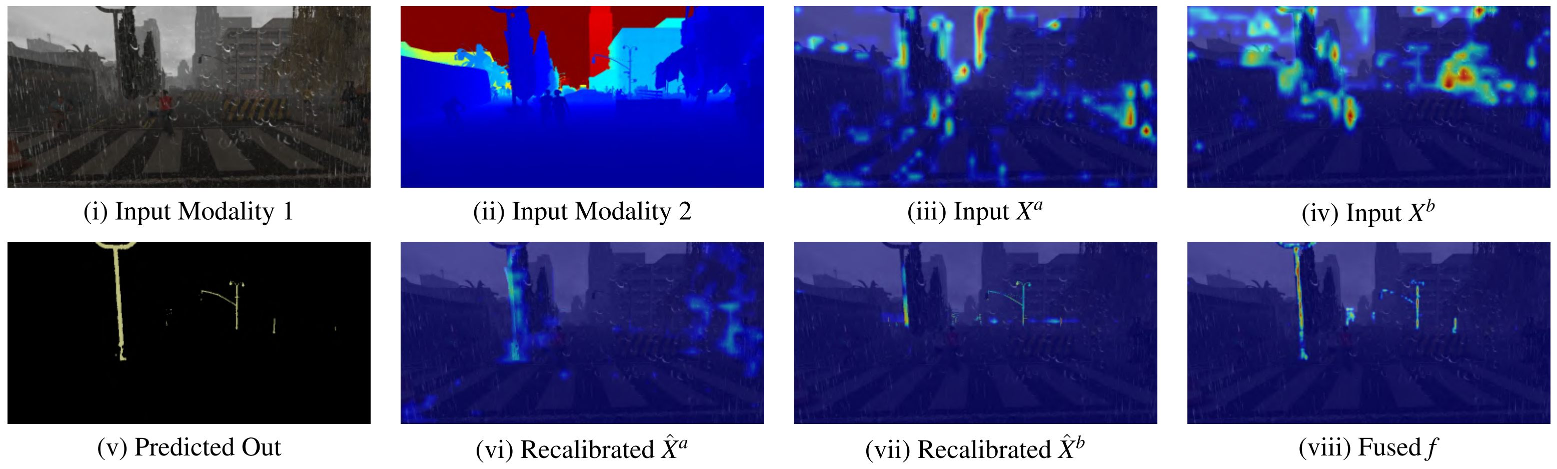
#### 5.3.4.3 Influence of Multimodal SSMA Fusion on Activation Maps

In an effort to present visual explanations for the improvement in performance due to multimodal fusion, we study the activation maps at various intermediate network stages before and after the multimodal fusion using the GradCam++ technique [204]. The approach introduces pixel-wise weighting of the gradients of the output with respect to a particular spatial location in the convolutional feature map to generate a score. The score provides a measure of the importance of each location in feature map towards the overall prediction of the network. We apply a colormap over the obtained scores to generate a heat map as shown in Figure 5.8. We visualize the activation maps at five different stages of the network. Firstly, at the output of each modality-specific encoder $X^a$ and $X^b$ which is the input to the SSMA fusion module. Secondly, after recalibrating the individual modality-specific feature maps inside the SSMA module $\hat{X}^a$ and $\hat{X}^b$, and finally after the fusion with the $3 \times 3$ convolution inside the SSMA module $f$. Figures 5.8, 5.9 and 5.10 illustrates one example for each dataset that we benchmark on, with the activation maps, the input modalities and the corresponding segmentation output for the particular object category.

For the Cityscapes dataset, we show the activation maps for the *person* category in Figure 5.8 (a). It can be seen that the activation map $X^a$ from the visual RGB stream is well defined for the *person* class but it does not show high activations centered on the objects, whereas the activation map from the depth stream $X^b$ is more noisy but high activations are



(a) Cityscapes - *Person* class



(b) Synthia - *Pole* class

**Figure 5.8:** Visualization of activation maps with respect to a particular class at various stages of the network before and after multimodal fusion on examples from the Cityscapes and Synthia datasets. $X^a$ and $X^b$ are at the outputs of the modality-specific encoder which is input to the SSMA, $\hat{X}^a$ and $\hat{X}^b$ are the feature maps after recalibration inside the SSMA module, and $f$ is after the fusion of both modalities.

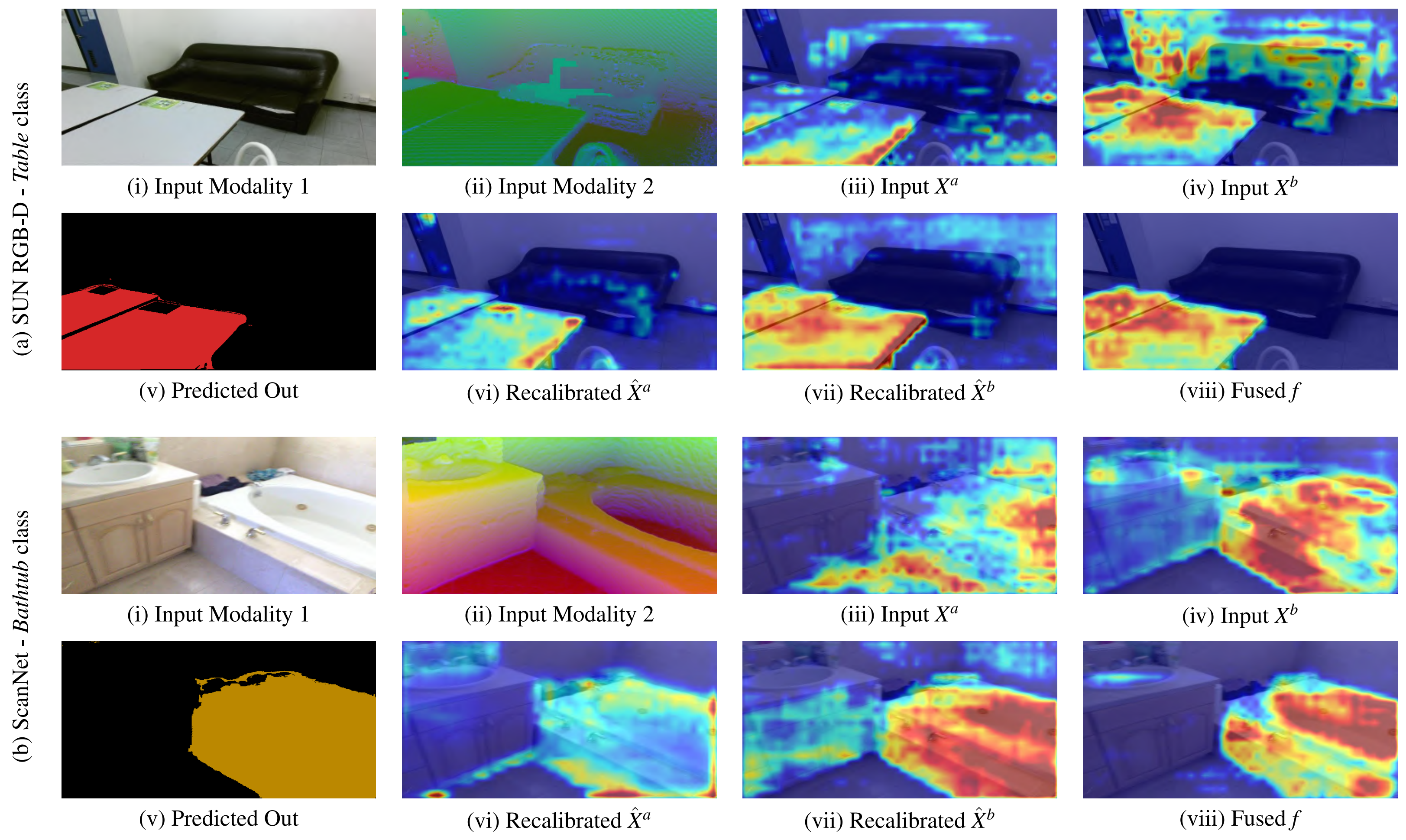


**Figure 5.9:** Visualization of activation maps with respect to a particular class at various stages of the network before and after multimodal fusion on examples from the SUN RGB-D and ScanNet datasets. $X^a$ and $X^b$ are at the outputs of the modality-specific encoder which is input to the SSMA, $\hat{X}^a$ and $\hat{X}^b$ are the feature maps after recalibration inside the SSMA module, and $f$ is after the fusion of both modalities.

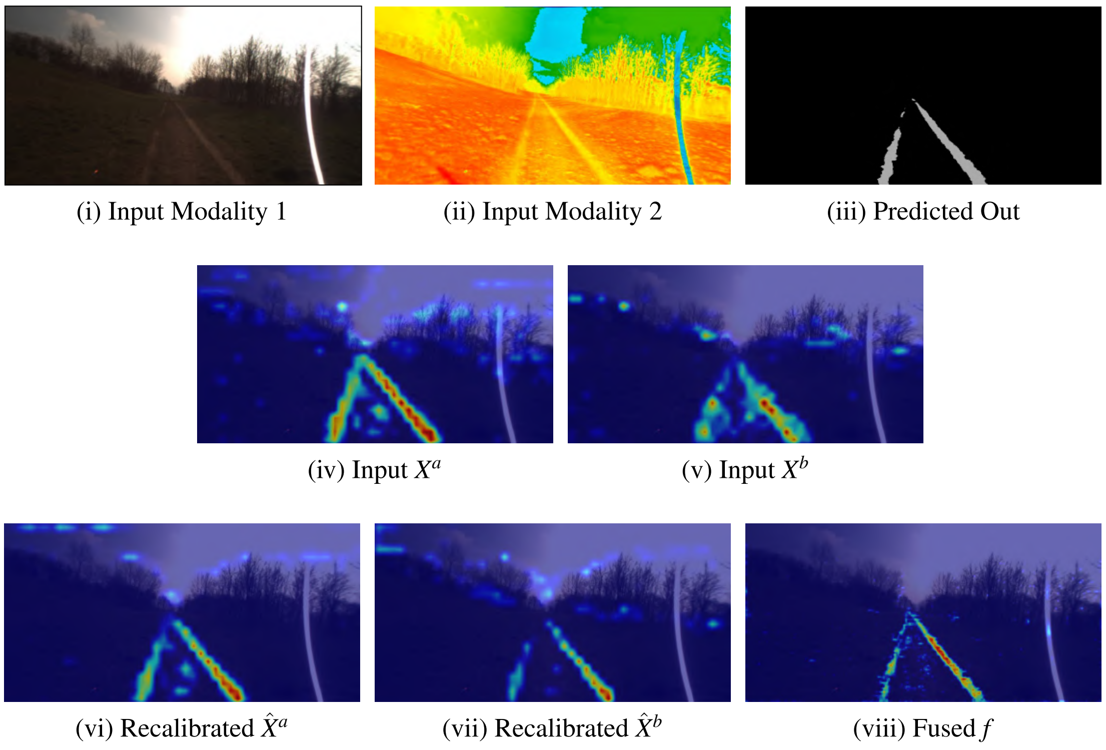


**Figure 5.10:** Visualization of activation maps with respect to the *Trail* class at various stages of the network before and after multimodal fusion on an example from the Freiburg Forest dataset. $X^a$ and $X^b$ are at the outputs of the modality-specific encoder which is input to the SSMA, $\hat{X}^a$ and $\hat{X}^b$ are the feature maps after recalibration inside the SSMA module, and $f$ is after the fusion of both modalities.

shown on the objects. For the locations in the input depth map that show noisy depth data, the activation map correspondingly shows prominent activations in these regions. After the recalibration of the feature maps, both $\hat{X}^a$ and $\hat{X}^b$ are less noisy while maintaining the structures with high activations. Furthermore, the activation map of the fused convolution $f$ shows very well defined high activations that almost correspond to the segmentation output.

Figure 5.8 (b) shows the activations for the *pole* class in the Synthia dataset. As the scene was captured during rainfall, the objects in the visual RGB image are indistinguishable. However, the depth map still maintains some structure of the scene. Studying both the modality-specific activation maps at the input to the SSMA module shows substantial amount of noisy activations spread over the scene. Therefore, the unimodal visual RGB model only achieves an IoU of 74.94% for the *pole* class. Whereas, after the recalibration of the feature maps, the activation maps show significantly reduced noise. It can be seen the recalibrated activation map $\hat{X}^b$ of the depth stream shows more defined high activations on the *pole*, whereas $\hat{X}^a$ of the visual RGB stream shows less amount of activations indicating that the network suppresses the noisy RGB activations in order to better leverage the well defined features from the depth stream. Activations of the final fused convolution layer show higher activations on the *pole* than either of the recalibrated activation maps demonstrating the utility of multimodal fusion. This enables the fusion model to achieve an improvement of 8.4% for the *pole* class.

While, for the indoor SUN RGB-D dataset, Figure 5.9 (a) shows the activation maps for the *table* class. Interestingly, both the modality-specific activation maps at the input show high activations at different locations indicating the complementary nature of the features in this particular scene. However, the activation map $X^b$ from the HHA stream also shows high activations on the *couch* in the background which would cause misclassifications. After the recalibration of the HHA feature maps, the activation map $\hat{X}^b$ no longer has high activations on the *couch* but it retains the high activations on the *table*. While, the recalibrated activation map $\hat{X}^a$ of the visual RGB stream shows significantly lesser noisy activations. The activation map of the fused convolution $f$ shows well defined high activations on the *table*, more than the modality-specific input activation maps. This enables the SSMA fusion model to achieve an improvement of 4.32% in the IoU for the *table* category.

For the ScanNet dataset, we show the activation maps for the *bathtub* category in Figure 5.9 (b). It can be seen that the modality specific activation maps at the input of the SSMA module shows high activations at complementary locations, corroborating the utility of exploiting features from both modalities. Moreover, the activation map $X^b$ from the HHA stream shows significantly higher activations on the object of interest than the RGB stream. This also aligns with the quantitative results, where the unimodal HHA model outperforms the model trained on visual RGB images. After the recalibration of the feature maps inside the SSMA module, the activation maps show considerably lesser noise while maintaining the high activations at complementary locations. The activation map of the fused convolution $f$ shows only high activations on the *bathtub* and resembles the actual structure of the segmented output.

Finally, Figure 5.10 (a) shows the activation maps for the *trail* category in the Freiburg Forest dataset. Here we show the fusion with visual RGB and EVI modalities. The EVI modality does not provide substantial complementary information for the *trail* class in comparison to the RGB images. This is also evident in the visualization of the activations at the input of the SSMA module. The activation maps of the EVI modality after the recalibration show significantly lesser noise but also smaller number of regions with high activation than the recalibrated activation maps of the visual RGB stream. Nevertheless, the activation map after the fusion $f$ shows more defined structure of the *trail* than either of the modality-specific activation maps at the input to the SSMA module prior to the recalibration.

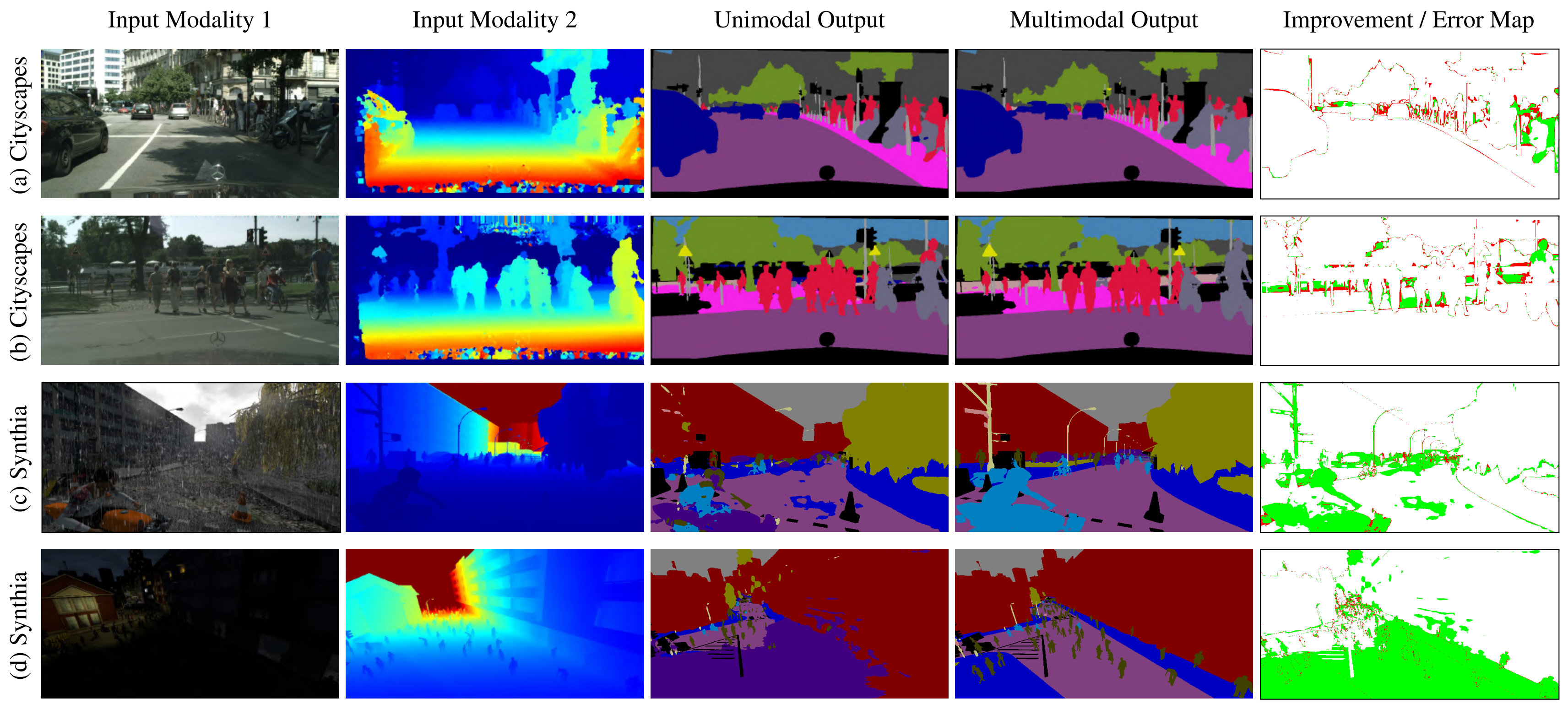


**Figure 5.11:** Qualitative multimodal fusion results in comparison to the output of the unimodal visual RGB model on the Cityscapes and Synthia datasets. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the unimodal AdapNet++ model but correctly predicted by multimodal SSMA model in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Tables A.1 and A.2. The regions denoted in black are the ignore labels in the datasets.

### 5.3.5 Qualitative Comparison

In this section, we present qualitative comparisons of multimodal semantic segmentation using our SSMA fusion approach on each of the five benchmark datasets. We compare with the output of unimodal AdapNet++ architecture and show the improvement / error map which indicates the improvement over the unimodal AdapNet++ output in green and the misclassifications by the multimodal model in red. Figures 5.11 (a) and (b) show interesting examples from the Cityscapes dataset. In both these examples, we can see a significant improvement in the segmentation of *cyclists*. As *cyclists* constitute to a person riding a bike, often models assign a part of the pixels on the person riding a bike as the *person* class, instead of the *cyclist* class.

Another common scenario is when there is a person standing a few meters behind a parked bike, the model misclassifies the *person* as a *cyclist* but since he is not on the bike, the right classification would be the *person* category. In these examples, we can see that by leveraging the features from the depth modality our network makes accurate predictions in these situations. In Figure 5.11 (a), we can also see that parts of the *car* that is several meters away is not completely segmented in the unimodal segmentation output but it is accurately captured in the multimodal segmentation output. Furthermore, in the unimodal output of Figure 5.11 (b), we see parts of the *sidewalk* behind the people is misclassified as *road* and parts of the *fence* that is several meters away is misclassified as a *sidewalk*. As the distinction between these object categories can clearly be seen in the depth images, our fusion model accurately identifies these boundaries.

Figures 5.11 (c) and (d) illustrate examples from the Synthia dataset. Here we show the first scene during rainfall and the second scene during night-time. In the unimodal output of first scene, we can see significant misclassifications in all the object classes, except *building* and *vegetation* that are substantially large in size. Whereas, the multimodal SSMA fusion model leverages the depth features to reliably identify the objects in the scene. In Figure 5.11 (d), even for us humans it is impossible to detect the *people* on the road due to the darkness in the scene. As predicted, the unimodal visual RGB model misclassifies the entire road with people as a *car*, which could lead to disastrous situations if it occurred in the real-world. Whereas, the multimodal SSMA fusion model accurately predicts the scene even in such poor illumination conditions.

In Figures 5.12 (a) and (b), we show examples on the indoor SUN RGB-D dataset. Due to the large number of object categories in this dataset, several inconspicuous classes often exist in the scene. Leveraging structural properties of objects from the HHA-encoded depth can enable better discrimination between them. Figure 5.12 (a) shows a scene where the unimodal model misclassifies parts of the wooden *bed* as a *chair* and parts of the *pillow* as the *bed*. We can see that the multimodal SSMA output significantly improves upon the unimodal counterpart. Figure 5.12 (b) shows a complex indoor scene with substantial clutter. The unimodal AdapNet++ model misclassifies the *table* as a *desk* and a hatch in the

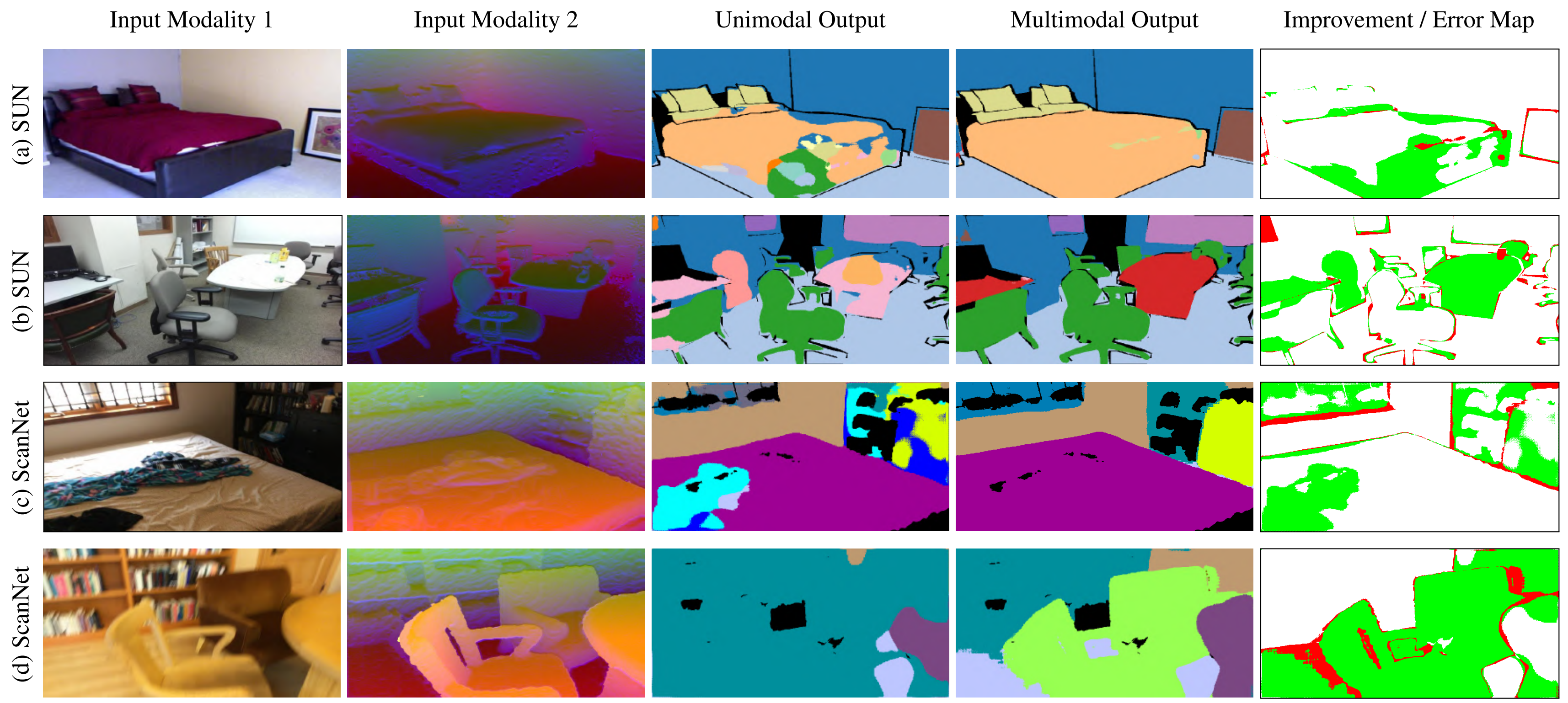


**Figure 5.12:** Qualitative multimodal fusion results in comparison to the output of the unimodal visual RGB model on the SUN RGB-D and ScanNet datasets. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the unimodal AdapNet++ model but correctly predicted by multimodal SSMA model in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Tables A.3 and A.5. The regions denoted in black are the ignore labels in the datasets.

wall is misclassified as a *door*. Moreover, partly occluded *chairs* are not entirely segmented in the unimodal output. Whereas, the HHA-encoded depth image shows the well defined structure of these objects, which enables the SSMA fusion approach to precisely segment the scene. Note that the *window* in the top left corner of Figure 5.12 (b) is mislabeled as a *desk* in the groundtruth mask.

Figures 5.12 (c) and (d) show examples of indoor scenes from the ScanNet dataset. In the unimodal output of Figure 5.12 (c), overexposure of the image near the windows causes misclassification of parts of the *window* as a *picture*. While, the crumpled bedsheets as well as the *bookshelf* are misclassified as a *desk*. On the other hand, the multimodal SSMA segmentation output does not demonstrate these errors. Figure 5.12 (d) shows an image with motion-blur due to camera motion. The motion-blur causes a significant percentage of the image to be misclassified as the largest object in the scene, which is a *bookshelf* in this scene. Analyzing the HHA-encoded depth map, we can see that it does not contain overexposed sections or motion-blur, rather it strongly emphasizes the structure of the objects in the scene. By leveraging features from the HHA-encoded depth stream, our multimodal SSMA model robustly predicts the different object classes even in the presence of these perceptual disturbances.

In Figures 5.13 (a) and (b), we show results on the unstructured Freiburg Forest dataset. Figure 5.13 (a) shows an oversaturated image due to sunlight which causes the boulders on the *grass* to be not captured in the unimodal segmentation output. Oversaturation causes boulders to appear with a similar texture as the *trail* or the *vegetation* class. However, the multimodal RGB-EVI model leverages the complementary EVI features to reliably segment these structures. Figure 5.13 (b) shows an example scene with glare on the camera optics and snow on the ground. In the unimodal semantic segmentation output, the presence of these disturbances often causes localized misclassifications in the areas where they are present. Whereas, the multimodal semantic segmentation model employing our SSMA fusion compensates for these disturbances exploiting the complementary relationship between the two modalities.

The last two rows of Figure 5.13 show interesting failure modes where the multimodal SSMA fusion model demonstrates incorrect predictions. In Figure 5.13 (c), we show an example from the Cityscapes dataset which contains an extremely thin *fence* connected by wires along the median of the road. The thin wires are barely captured by the depth modality and it is visually infeasible to detect the fence from the RGB image. Moreover, due its thin structure, the vehicles on the opposite lane are clearly visible. This causes both the unimodal and multimodal models to partially segment the vehicles behind the fence, thereby causing incorrect predictions according to the groundtruth mask. However, we can see that the multimodal SSMA fusion model still captures more of the *fence* structure than the unimodal AdapNet++ model by leveraging the complementary depth information.

In Figure 5.13 (d), we show an example from the SUN RGB-D dataset in which misclassifications are produced due to inconspicuous classes. The scene contains two

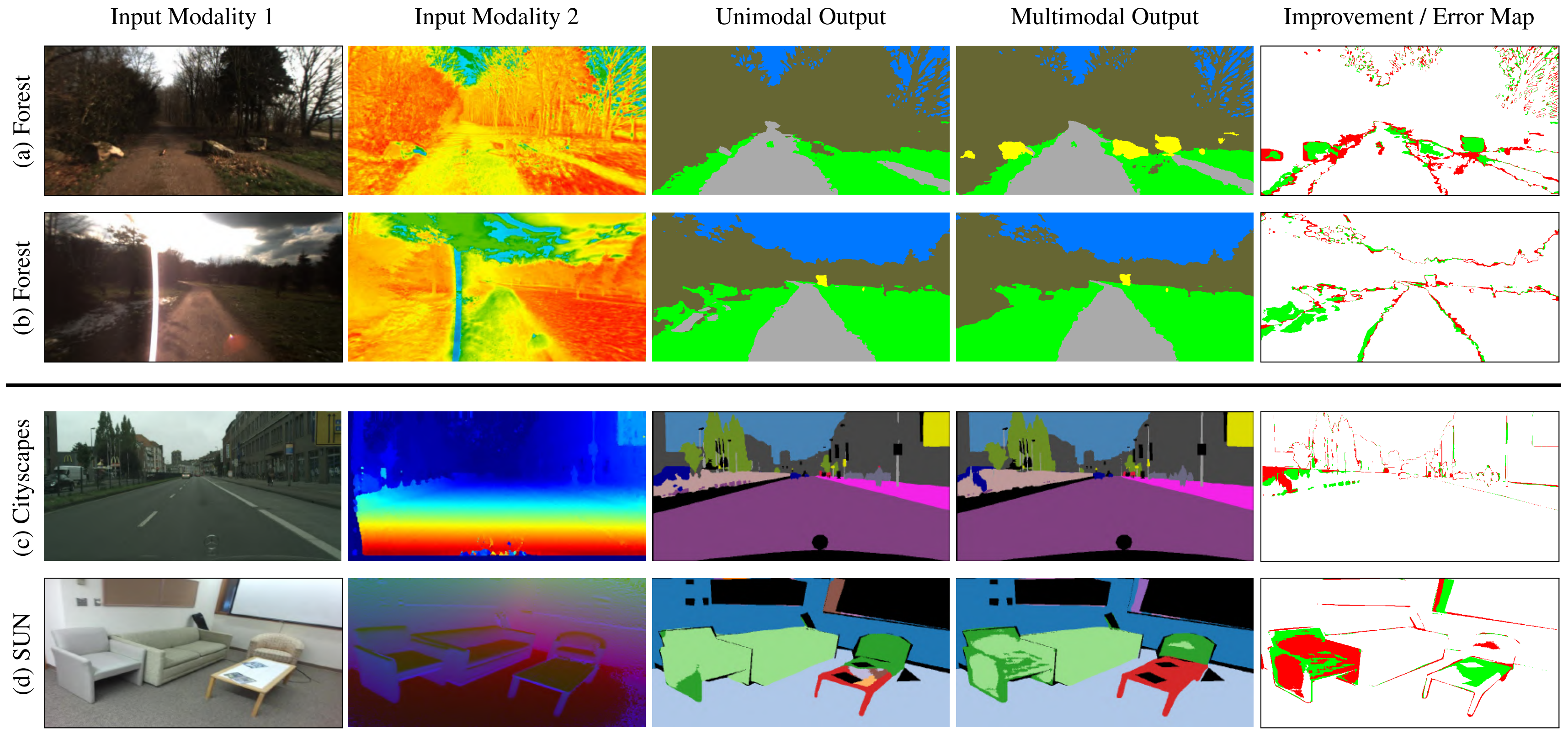


**Figure 5.13:** Qualitative multimodal fusion results in comparison to the output of the unimodal visual RGB model on the Freiburg Forest dataset. The last two rows show failure modes on the Cityscapes and SUN RGB-D datasets. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the unimodal AdapNet++ model but correctly predicted by multimodal SSMA model in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Appendix A. The regions denoted in black are the ignore labels in the datasets.

object categories that have very similar appearance in some scenes, namely, *chair* and *sofa*. The *chair* class is denoted in dark green, while the *sofa* class is denoted in light green. As we see in this scene, a single-person *sofa* is considered to be a *chair* and only the longer *sofa* in the middle of the image is considered to be in the *sofa* class according to the groundtruth mask. In this scene, the single person *sofa* is adjacent to the longer *sofa* which causes the both the unimodal AdapNet++ model and the multimodal SSMA fusion model to predict the pixels on both these objects as the *sofa* class. However, we still observe more pixels that are correctly classified as the *chair* class in the multimodal SSMA output than in the unimodal AdapNet++ output.

### 5.3.6 Visualizations Across Seasons and Weather Conditions

In this section, we present qualitative results on the Synthia-Sequences dataset that contains video sequences of 12 different seasons and weather conditions. We visualize the segmentation output of the multimodal RGB-D SSMA fusion model and unimodal AdapNet++ model for which the quantitative results are shown in Figure 5.7. For this experiment, the models were trained on the Synthia-Rand-Cityscapes dataset and only evaluated on the Synthia-Sequences dataset. The Synthia-Sequences dataset contains a diverse set of conditions such as *summer, fall, winter, spring, dawn, sunset, night, rain, soft-rain, fog, night-rain* and *winter-night*. We show qualitative evaluations on each of these conditions by comparing the multimodal segmentation performance using the SSMA fusion with the output obtained from the unimodal visual RGB model. This aim of this experiment is twofold: to study the robustness of the model to adverse perceptual conditions such as rain, snow, fog and nightime; Secondly, to evaluate the generalization of the model to unseen scenarios.

From the examples shown in Figures 5.14, 5.15 and 5.16, we can see the diverse nature of these scenes containing environments such as highway driving, inner-city with skyscrapers and small-sized cities. The RGB images show the visual appearance variations caused by changing seasons and weather conditions. These appearance changes can be observed in the color of the vegetation in Figure 5.14 (b), snow on the ground and leaf-less trees in Figure 5.14 (c), glaring light due to sunrise in Figure 5.15 (a), orange hue due to sunset in Figure 5.15 (b), dark scene with isolated lights in Figure 5.15 (c), adverse visibility due to rain in Figure 5.15 (d) and blurred visibility due to fog in Figure 5.16 (b). Even for us humans it is extremely hard to identify the different objects in some of these environments. The third column shows the output obtained from the unimodal AdapNet++ model trained on visual RGB images. The output shows significant misclassifications in scenes that contain rain, fog, snow or nighttime. Whereas, the multimodal SSMA model using RGB-D fusion precisely segments the scene by leveraging the more stable depth features.

From the improvement / error map shown in the last column, we can see a substantial improvement (green pixels) over unimodal semantic segmentation and minimal error (red

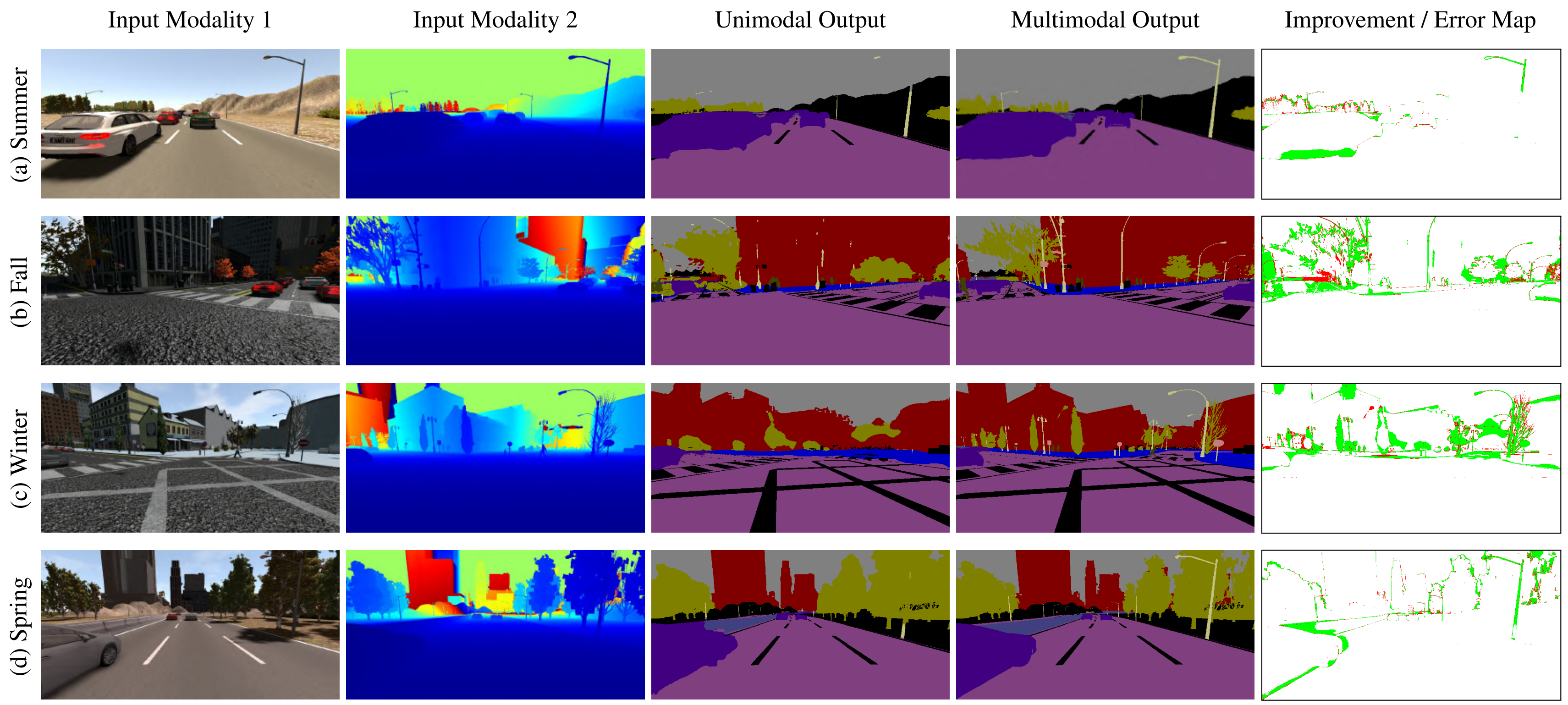


**Figure 5.14:** Qualitative multimodal semantic segmentation results in comparison to the output of the unimodal visual RGB model for scenes from the Synthia-Seasons dataset depicting (a) summer, (b) fall, (c) winter, and (d) spring. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Table A.2. The regions denoted in black are the ignore labels in the datasets.

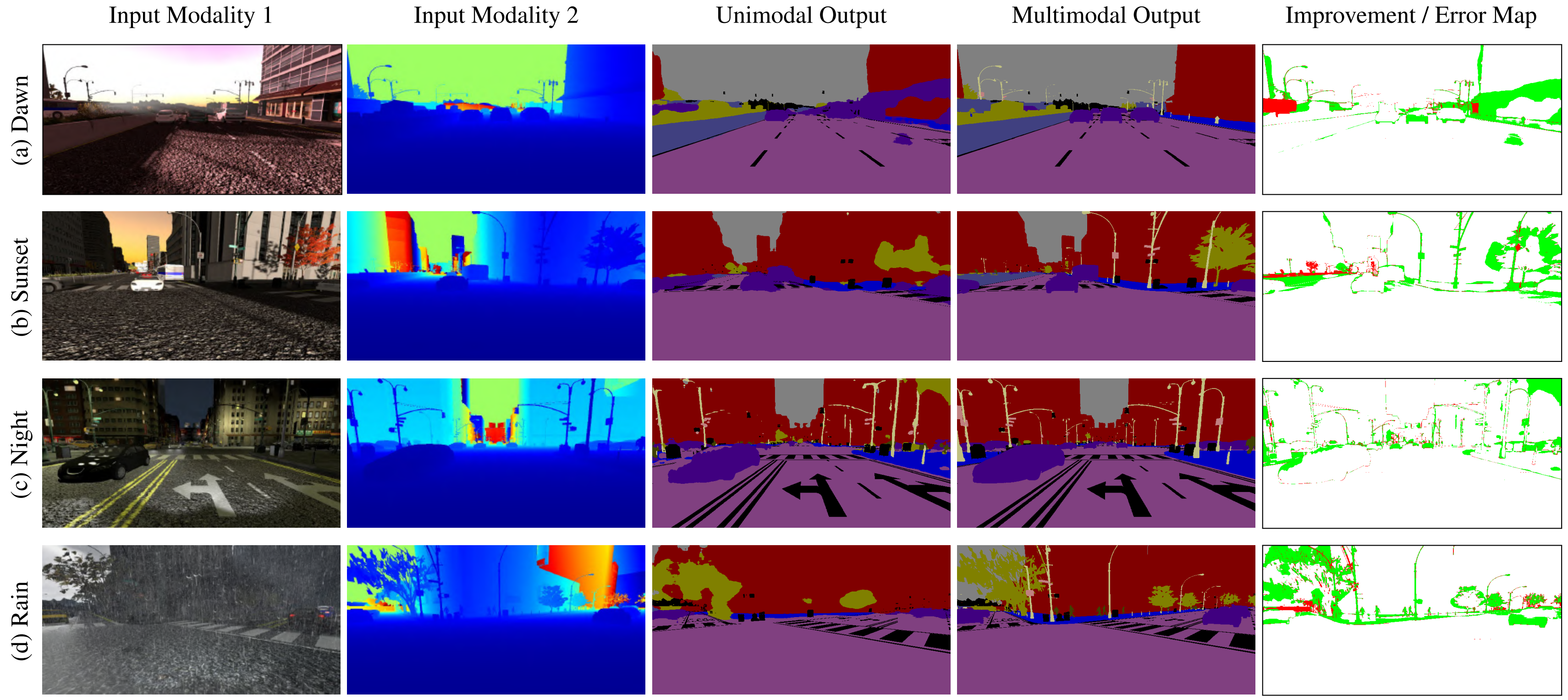


**Figure 5.15:** Qualitative multimodal semantic segmentation results in comparison to the output of the unimodal visual RGB model for scenes from the Synthia-Seasons dataset depicting (a) dawn, (b) sunset, (c) night, and (d) rain. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Table A.2. The regions denoted in black are the ignore labels in the datasets.

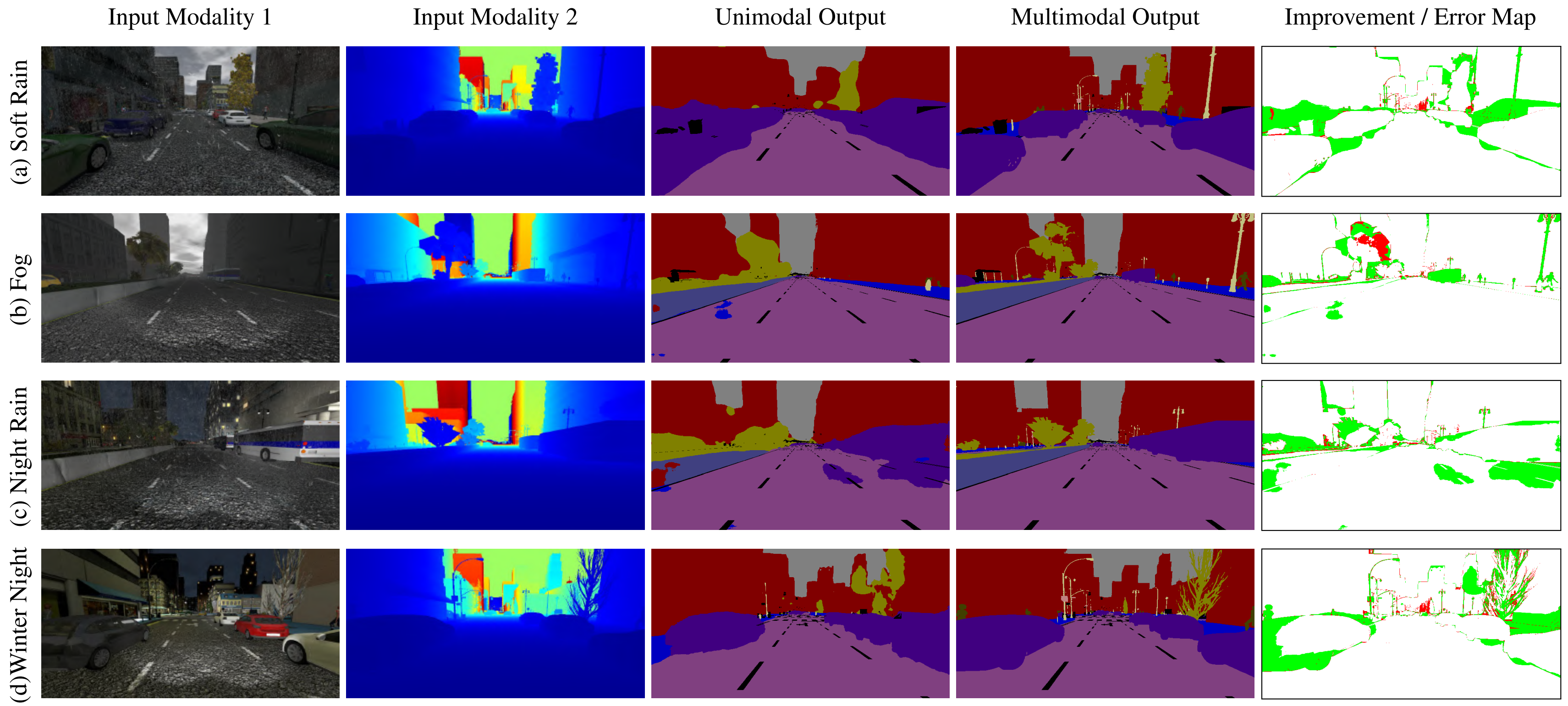


**Figure 5.16:** Qualitative multimodal semantic segmentation results in comparison to the output of the unimodal visual RGB model for scenes from the Synthia-Seasons dataset depicting (a) soft-rain, (b) fog, (c) night-rain, and (d) winter-night. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the best performing state-of-the-art model but correctly predicted by AdapNet++ in green. The color legend for the segmentation labels correspond to those shown in the detailed benchmarking results in Table A.2. The regions denoted in black are the ignore labels in the datasets.

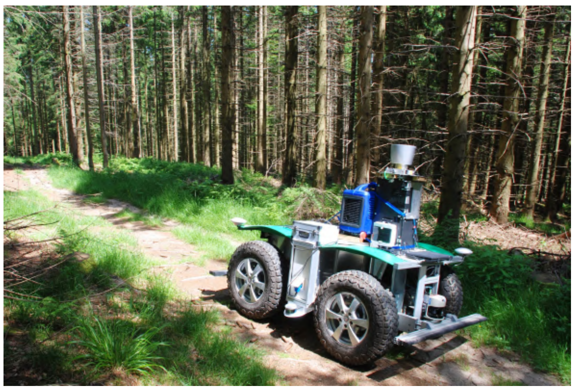


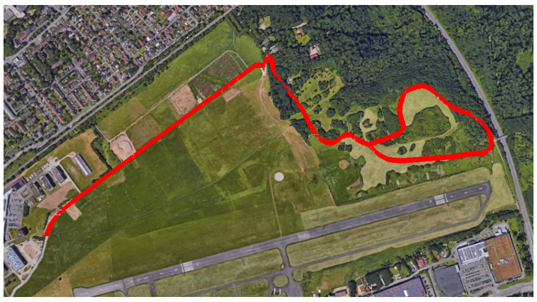
**Figure 5.17:** Our Viona robot autonomously navigating in a forested environment near Freiburg using our multimodal semantic segmentation network for perception. The robot was equipped with four Bumblebee2 stereo cameras mounted in perpendicular directions and an NVIDIA Jetson TX1 for the computation.

pixels) while employing the multimodal SSMA fusion. The error is noticeable only along the boundaries of objects that are at far away distances, which can be remedied using a higher resolution input image. Figure 5.15 (a) and Figure 5.16 (b) show partial failure cases. In the first example in Figure 5.15 (a), the occluded *bus* on the left is misclassified as a *fence* due to being parked on the sidewalk, where often fences appear in the same configuration. While, in Figure 5.16 (b), a part of the *vegetation* several meters away is misclassified as a *building* due to the haziness caused by the fog. Nevertheless, overall the SSMA fusion approach effectively generalizes to unseen environments and visibility conditions demonstrating the efficacy of our mutimodal semantic segmentation network.

### 5.3.7 Real-World Navigation Experiment

We performed real-world navigation experiments using our Viona robot that was equipped with our multimodal semantic segmentation network for perception. The robot shown in Figure 5.17 has four Bumblebee2 stereo vision cameras mounted at perpendicular directions to obtain a 360 degree field of view of the environment. We used the NVIDIA Jetson TX1 which has 256 CUDA cores for deploying our model. We implemented a segmentation pipeline using ROS and the Caffe deep learning library for this experiment. The robot first captured images of the scene using the stereo cameras and segments the images with our multimodal fusion model trained on the Freiburg Forest dataset. Waypoints for the robot to follow were then computed from the segmentation output with the aim of keeping the robot on the navigable trail. The computed waypoints were subsequently forwarded to the planner that executed the trajectory.

The robot autonomously navigated a total of 4.52 km with an average speed of $0.9\,\mathrm{m\,s^{-1}}$ in a challenging forested environment as shown in Figure 5.18. A forward-pass on


(a) Trajectory that was driven by our robot during the autonomous driving experiments.

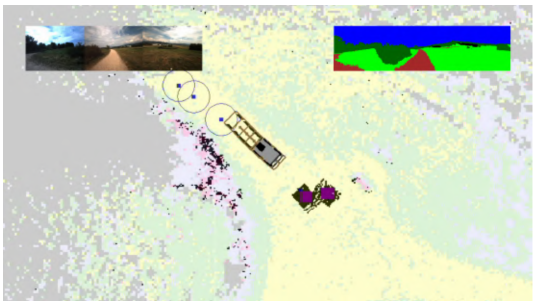

(b) Planning interface showing waypoints, camera images and the corresponding segmentation output.

**Figure 5.18:** Our robot autonomously traversed 4.52 km using only the semantic segmentation from our multimodal network for perception. Waypoints were derived from the segmentation and sent to the autonomy system on the robot for trajectory following.

the NVIDIA TX1 embedded GPU consumed 623 ms. During the entire experiment the robot did not have any prior map of the environment, therefore it employed the semantic segmentation on the fly to traverse the trail. The robot encountered several challenging situations including low-lighting due to forest canopy, occasional glare from the sun, shadows from trees and motion-blur. The perception system equipped with our multimodal semantic segmentation model was robust to these disturbances and performed inference online that enabled the successful autonomous navigation.

## 5.4 Related Work

The work presented in this chapter addresses the problem of multimodal semantic segmentation using convolutional neural networks that incorporate adaptive fusion techniques to exploit complementary features from multiple modalities based on the scene condition. These techniques enable a robot to robustly perceive and understand scenes in the complex real-world even in challenging environmental conditions such as snow, fog and rain. In this section, we present a thorough review of the most related works in this context in order to highlight our contributions.

The advent of low-cost perceptual sensors has enabled several novel approaches to exploit features from alternate modalities in an effort to improve robustness as well as the granularity of the segmentation. Silberman *et al.* [164] propose an approach based on SIFT features and MRFs for indoor scene segmentation using RGB-D images. Subsequently, Ren *et al.* [205] propose improvements to the feature set by using kernel descriptors and by combining MRF with segmentation trees. Bo *et al.* [206] propose an approach that learns dictionaries using K-SVDs from RGB, depth, gray-scale intensities and surface normals. Subsequently, a Hierarchical Marching Pursuit (HMP) method [207] was introduced to generate high-level representations from learned spare codes of local patches. Munoz *et al.* [208] employ modality-specific classifier cascades that hierarchically propagate information and do not require one-to-one correspondence between data across modalities. In addition to incorporating features based on depth images, Hermans *et al.* [209] propose an approach that performs joint 3D mapping and semantic segmentation using Randomized Decision Forests. In most of these approaches, hand engineered or learned features are extracted from individual modalities and combined together in a joint feature set which is then used for classification. These features do not readily extend to different modalities and datasets as they are tuned for specific scenarios. Moreover, these features only capture a subset of cues that are useful for multimodal perception.

More recently, there has been a series of CNN-based techniques [163, 210, 211] that have been proposed for end-to-end learning of fused representations from multiple modalities. As opposed to visual RGB images which are three-channel images, alternate modalities such as depth and infrared are one-channel images. Although, approaches have been proposed that directly learn from one-channel depth images [197], encoding the depth map into the HHA representation [24, 161] or converting it into a three-channel image by applying a jet colormap [163] has been demonstrated to be more effective for feature learning using CNNs. Techniques have also been proposed to augment the standard convolution with a depth similarity term to force pixels with similar depths at the center of the filter to contribute more to the output [212]. CNN-based multimodal learning approaches can be categorized into early, hierarchical and late-fusion methods. An intuitive early-fusion technique is to stack data from multiple modalities channel-wise and feed it to the network as a four or six-channel input. Couprie *et al.* [213] propose a multiscale CNN for scene labeling that takes a four-channel RGB-D image as input. However, experiments have shown that this often does not enable the network to learn complementary features and cross-modal interdependencies as they do not exploit the different characteristics of the modalities [50, 162].

Hierarchical fusion approaches combine feature maps from multiple modality-specific encoders at various levels (often at each downsampling stage) and upsample the fused features using a single decoder [162, 210]. Jiang *et al.* [214] employ an hierarchical fusion approach with individual RGB and depth streams that fuse features at different encoder stages while additionally having skip connections to the decoder at the same stages. Alternatively, Schneider *et al.* [201] propose a mid-level fusion approach in which NiN layers [69] with depth as input are used to fuse feature maps into the RGB encoder in the middle of the network. Eigen *et al.* [197] employ a global-to-local strategy that extracts features using a CNN and combines different levels of predictions. Li *et al.* [211] propose a Long-Short Term Memory (LSTM) context fusion model that captures and fuses contextual information from multiple modalities accounting for the complex interdependencies between them. Lin *et al.* [215] employ an FCN-based approach that uses multiple branches for different discrete depth values in order to provide a better control on the contextual information of learned features. Qi *et al.* [216] propose an interesting approach that employs 3D graph neural networks for RGB-D semantic segmentation that accounts for both 2D appearance and 3D geometric relations, while capturing long range dependencies within images. A major shortcoming of these approaches is that the complementary nature of different modalities are not fully exploited due to the direct element-wise addition or concatenation that is performed to fuse the features.

In the late-fusion approach, identical network streams are first trained individually on a specific modality and the feature maps are fused towards the end of network using concatenation [163] or element-wise summation [50], followed by learning deeper fused representations. Lin *et al.* [217] propose an approach in which individual CNNs are employed to extract features from RGB images and HHA-encoded maps individually, followed by concatenating the resulting features and employing them as input to an SVM for classification. Socher *et al.* [218] employ a single layer CNN to learn low-level features which are then input to a Recurrent Neural Network (RNN) for high-level feature learning. Features from RGB and depth modalities are learned separately and then concatenated before the softmax layer. Although these techniques have demonstrated improved performance over models that solely employ visual RGB images, they do not enable the network to adapt the fusion to changing scene context, which is one of the main focus of the approaches presented in this chapter.

Another approach to fusing multimodal features from multiple specialized networks is related to the Mixture of Experts (MoE) technique. Hinton *et al.* [219] presented the classical MoE model consisting of experts and a supporting gating network, where each expert maps their respective inputs to a corresponding output and the gating network produces a probability distribution over the experts. Eigen *et al.* [220] extend the concept of MoEs by employing CNNs as experts to classify MNIST [221] and monophone speech data. This paper highlights the fact that using a mixture with deep networks increases the number of trainable parameters without significantly increasing the computational burden. Cheng *et al.* [222] introduce a locality-sensitive CNN that performs gated fusion by building a feature affinity matrix that enables weighted average pooling and unpooling. Subsequently, Mees *et al.* [223] propose a MoE technique that uses inner-product layers to learn modality-specific weightings which are then used to adaptively compute a weighted average over the outputs of CNN streams that take different modalities as input. The aforementioned techniques provide a framework for adaptively weighing the features of different CNN streams. However, it is insufficient to solely adapt the features according to different modalities as several other factors such the type of objects and its location in the scene also influence the selection of modality-specific features.

In order to address this problem, in this chapter, we presented the CMoDE fusion approach for learning the most discriminative features from each modality and probabilistically fusing the semantically high-level features according to the different object categories in the scene which enables more flexibility in learning the optimal multimodal feature combination. Nevertheless, there are several real-world scenarios in which class-wise fusion is insufficient, especially in outdoor scenes where different modalities perform well in different conditions. Moreover, the CMoDE module employs multiple softmax loss layers for each class to compute the probabilities for fusion which does not scale efficiently when the number of object categories to be classified significantly increases. Motivated by this observation, we also proposed a multimodal semantic segmentation architecture incorporating our SSMA fusion module that dynamically adapts the fusion of intermediate network representations from multiple modality-specific network streams according to the object class, its spatial location and the scene context while learning the fusion in a self-supervised fashion.

## 5.5 Conclusions

In this chapter, we presented two multimodal semantic segmentation architectures that adaptively fuse features from multiple modalities and spectra, in order to exploit complementary information based on the scene condition. The first architecture incorporates our proposed CMoDE fusion mechanism that probabilistically fuses high-level semantic features from multiple modality-specific network streams to exploit complementary features according to the different types of objects in the scene and further learns discriminative fused representations. Our second architecture incorporates our proposed SSMA modules dynamically adapt the fusion of features from modality-specific streams at various intermediate network stages in order to optimally exploit complementary features. Our SSMA fusion mechanism is simultaneously sensitive to critical factors that influence multimodal fusion including the object category, its spatial location and the environmental scene context, in order to fuse only the relevant complementary information. We also introduced a channel attention mechanism for better correlating the fused mid-level modality-specific encoder features with the high-level decoder features for object boundary refinement. Moreover, as the fusion mechanism is self-supervised, we demonstrated that it effectively generalizes to the fusion of different modalities, beyond the commonly employed RGB-D data and across different environments ranging from urban driving scenarios to indoor scenes and unstructured forested environments. Unlike most existing multimodal fusion techniques, our proposed fusion mechanisms are independent of the base semantic segmentation architecture and they can be easily employed in other segmentation frameworks, as well for other multimodal perception tasks.

We presented comprehensive benchmarking results on five standard scene understanding datasets including Cityscapes, Synthia, SUN RGB-D, ScanNet and Freiburg Forest. Both our proposed multimodal fusion approaches outperform existing techniques and set the new state-of-the-art on all the aforementioned benchmarks. We also presented thorough ablation studies that give insight on our architectural decisions and provided distinctly interpretable visual explanations that demonstrate the internal functioning of our multimodal fusion scheme. Furthermore, we presented extensive qualitative evaluations on each of the benchmarked datasets and more importantly, results in adverse perceptual conditions that demonstrate the exceptional robustness of our model to a variety of seasonal changes and weather conditions.

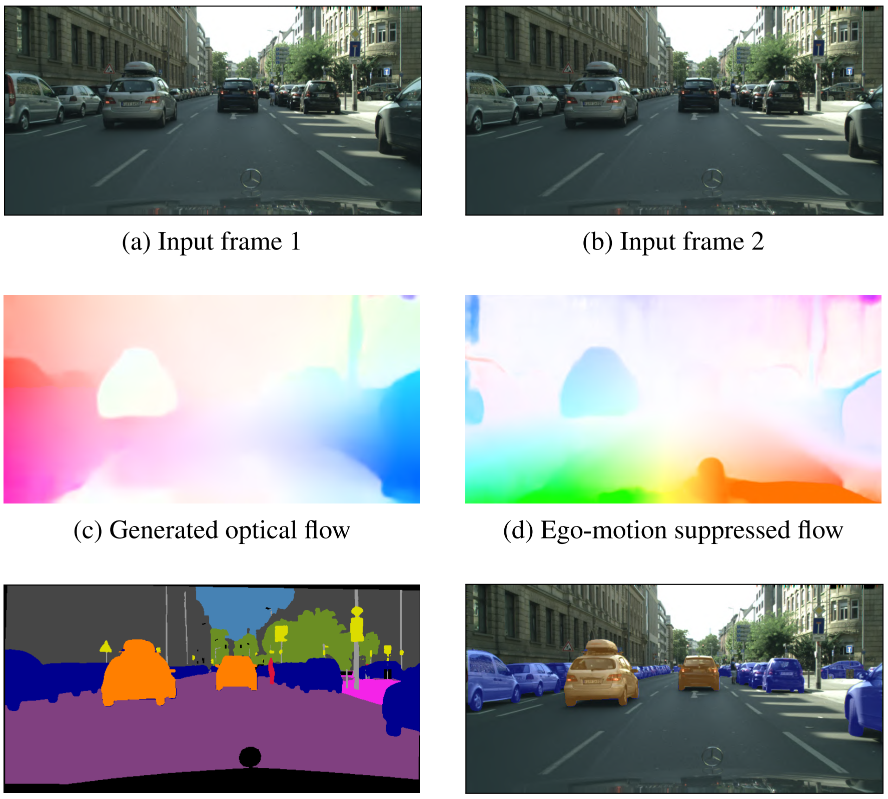
# Chapter 6

# Joint Semantic Motion Segmentation

**Understanding the dynamic characteristics of objects in the scene is an essential prerequisite for autonomous robots that enable them to reason and operate in complex real-world environments. Typically, motion segmentation of objects is achieved using dense optical flow methods that make homogenous assumptions about the spatial structure of the flow. However, different semantic objects in the scene such as pedestrians and cars inherently move differently. Moreover, the problem becomes even more challenging in the presence of fast and unstable ego-motion induced by the moving robot. In this chapter, we propose two novel convolutional neural network architectures that learn to predict both the semantic object label and motion status of each pixel in an image. Our proposed SMSnet architecture takes a pair of consecutive images as input and first learns a coarse representation of the optical flow field features, which are then augmented with learned semantic features and subsequently refined to yield pixel-wise semantic motion labels at the original high-resolution. We also propose the SMSnet++ architecture, in which we strengthen the joint modeling of both semantic and motion cues, while concurrently improving the temporal consistency of semantic segmentation using an edge-enhanced flow transformation to warp intermediate network representations across time. Extensive experiments demonstrate that our proposed architectures achieve state-of-the-art performance on the challenging Cityscapes, KITTI and ApolloScape benchmark datasets.**

## 6.1 Introduction

Recent advancement in robotic technology and machine learning has led to the successful deployment of robots to accomplish tasks in various structured and semi-structured en-



(a) Input frame 1 (b) Input frame 2

(c) Generated optical flow (d) Ego-motion suppressed flow

(e) Semantic motion segmentation (f) Overlay of static and moving cars

**Figure 6.1:** Illustration of semantic motion segmentation using our proposed SMSnet++ architecture trained on the Cityscapes dataset. Our network jointly predicts both the semantic object category and the motion status of each pixel in the image. Static cars are shown in blue, while moving cars are shown in orange.

vironments such as factory floors, domestic homes, warehouses and offices. This recent success has now paved the way to tackle more complex tasks in challenging urban environments that contain many dynamic objects such as cars, bicycles and pedestrians. Thus inferring the state of motion of these objects plays a crucial role in ensuring the viability and safe operation in such scenarios. Predicting motion patterns is a fundamental scene understanding problem that has been addressed with various techniques that exploit motion cues including optical-flow computation [224, 225], scene flow computation [226, 227], motion estimation [228, 229] and video segmentation [230, 231]. Despite the progress over the years, state-of-the-art methods are still challenged by factors such as fast motion of objects, varying pixel displacements due to object motion with different velocities, low-textured regions and occlusion boundaries around objects. Moreover, as the robot equipped with the camera is itself moving, it additionally introduces challenges such as ego-motion induced flow, lighting changes between consecutive frames and motion-blur.

Inspired by the progress in semantic segmentation of static scenes [59, 132, 135], recent work [232, 233] has explored the benefit of learning semantic cues and motion cues jointly as both provide complementary information about the scene where features learned for semantic labeling can help infer motion labels and vice versa. On one hand, the transition in motion is most likely to occur at the dynamic object class boundaries, while on the other hand, the semantic class of the object itself provides important cues for predicting motion patterns. For example, in dynamic scenes, the probability of a car or person moving is greater than the probability of a building or pole. Moreover, as different objects move differently, it provides cues on the expectation of motion that varies between regions with different object class labels. Finally, semantic cues also encourage temporal consistency of the motion segmentation as the identities of the objects remain constant over time. From the task perspective, the joint knowledge about the semantics and motion of objects gives us a more holistic understanding of the scene that allows robotic systems to increase their awareness, reason about behaviors and plan autonomous actions more effectively.

In Chapter 4, we introduced architectures for efficient semantic segmentation and subsequently in Chapter 5, we presented methods that substantially improve the robustness by exploiting complementary multimodal features. These networks enable a robot to accurately segment the scene into different semantic object categories, even in challenging perceptual conditions that robots encounter in the complex real-world. While in this chapter, we build upon the techniques that we introduced thus far and propose a joint learning framework to generate richer semantic descriptions by prefixing motion labels to potentially dynamic semantic object classes such as a *static car* and a *moving car* to enable a more comprehensive understanding of the scene. Deep Convolutional Neural Network (CNN) based approaches have significantly improved the state-of-art in both semantic segmentation [24] and motion estimation [225]. There are numerous approaches that model the motion of objects separately [228, 230, 234, 235, 236] and the semantics of the scene separately [53, 67, 135, 136, 138]. However, currently there are only a handful of methods [232, 233, 237] that aim to capture both semantic and motion cues jointly considering that they are both interrelated.

Reddy *et al.* [232] propose an approach that integrates semantic, geometric and optical flow based constraints into a dense fully connected CRF model. They report improved results in motion segmentation using the joint formulation compared to geometry-based motion segmentation, and joint optimization with superpixel-based clique and motion estimate. Fan *et al.* [233] propose an approach which combines semantic segmentation obtained from a fully convolutional neural network (FCN) with stereo-vision based motion segmentation using dense CRFs. Their semantic motion segmentation model outperforms the approach of Reddy *et al.* [232] by incorporating better semantic feature learning. Subsequently, Haque *et al.* [237] propose a three stage pipeline that integrates optical flow as a constraint with semantic features into a dilated CNN for semantic motion segmentation. Their approach achieves state-of-the-art results outperforming the other methods on the KITTI dataset. However, the aforementioned techniques have long multistage pipelines that have slow run-times deeming them impractical for real-world applications. Moreover, these approaches only exploit semantic features to improve motion segmentation, whereas they do not consider the fact that the learned motion features can also be leveraged to enable more accurate semantic segmentation. Another major hindrance is the lack of a large enough dataset with groundtruth semantic motion annotations that enable training of CNNs without overfitting and allow for credible quantitative evaluations as these approaches have only been benchmarked on 200 images.

In order to address these problems, in this chapter, we propose two convolutional neural network architectures that learn to predict both the semantic category and motion status of each pixel from a pair of consecutive images. Our first architecture termed SMSnet is composed of three components: a stream that learns coarse representations of the optical flow field features, a parallel stream that generates discriminative semantic features, and a fusion section that combines both motion and semantic features while refining the predictions to generate a high-resolution pixel-wise semantic motion segmentation output. We build the different streams of the SMSnet architecture based on the AdapNet topology that we introduced in Chapter 4 and we embed the FlowNet2 [225] architecture to generate the optical flow maps. Often, the movement of the robot in the scene causes discontinuities in the optical flow magnitude. Nearby static objects may appear to have a larger optical flow magnitudes although they are not moving. In order to alleviate this problem, we employ a ego-flow suppression technique in which we compute the optical flow purely caused by the ego-motion using IMU and odometry readings, and subtract it from the flow magnitudes predicted by the embedded flow generation network.

Our proposed SMSnet model jointly learns to accurately predict both the semantic label and motion status of each pixel in the image. Although it uses the semantic cues to improve the motion segmentation performance, it does not exploit the learned motion cues to improve the semantics learning. In order to address this factor as well as to further improve the performance of both semantic and motion segmentation, we propose the SMSnet++ architecture. The topology of the architecture follows the similar three stream principle as SMSnet. However, we build each of the streams based on our improved AdapNet++ architecture that includes our efficient atrous spatial pyramid pooling for multi-scale context aggregation and a strong a decoder that significantly improves the performance along the object boundaries. For generating optical flow maps in SMSnet++, we embed the new FlowNet3 architecture [227] that introduces a technique to estimate occlusion areas jointly along with the optical flow. We then employ our ego-flow suppression technique to remove any flow magnitudes caused by the movement of the robot equipped with the camera. Subsequently, we fuse the optical flow field features with the semantic features using our proposed SSMA fusion scheme that we introduced in Chapter 5, rather than the simple concatenation that we employ in SMSnet. The SSMA module learns to adaptively fuse the motion and semantic features by enhancing the optical flow field features at areas containing moving objects, while suppressing the activations due to noise at the static regions. Furthermore, in order to simultaneously improve the semantic segmentation performance, we first transform the ego-flow suppressed optical flow to an edge-enhanced representation using the NetWarp module [238] and subsequently utilize it to warp intermediate semantic representations of the previous frame into the corresponding representations of the current frame. We then fuse the warped feature maps and the feature maps of the current frame using our SSMA module that we introduced in Chapter 5. This representation warping method enables our network to achieve temporal consistency across frames resulting in improved prediction accuracy.

Training our proposed networks requires datasets with consecutive image pairs and pixel-level groundtruth labels with both semantic and motion annotations for the current frame. Thus far, the only publicly available semantic motion segmentation dataset contains 200 labeled images from the KITTI [239] benchmark, which is highly insufficient for training deep networks. Therefore, in order to facilitate this work, we manually labeled the 3375 training and validation images of the Cityscapes semantic segmentation dataset [143], with motion annotations. As existing techniques are benchmarked on the KITTI dataset, we labeled 255 images with semantic and motion annotations that we employ for training, while using the existing 200 labeled images from Reddy *et al.* [239] for testing. Additionally, we also labeled 40,960 training images and 8327 testing images from the recently introduced ApolloScape semantic segmentation dataset [240], with motion annotations. We benchmark our proposed architectures as well as existing semantic motion segmentation techniques on our publicly released Cityscapes-Motion, KITTI-Motion and ApolloScape-Motion datasets. Extensive empirical evaluations demonstrate that both our architectures set the new state-of-the-art on these benchmarks, while being several times faster than existing techniques. To the best of our knowledge our networks are the first end-to-end learning techniques to address the problem of semantic motion segmentation, in addition to being the first to simultaneously improve the performance of both these tasks in a joint formulation.

Concretely, we make the following contributions in this chapter:

- A novel end-to-end convolutional neural network architecture that takes consecutive images as input and jointly learns to predict both the semantic object category and motion status of each pixel in an image.
- An refined architecture that adaptively fuses optical flow field features with semantic features, while simultaneously exploiting the optical flow for learning temporally consistent semantics.
- A representational warping scheme that improves the temporal consistency of semantic segmentation using our ego-flow suppressed optical flow with the NetWarp module [238] and subsequently fusing warped temporal features dynamically using our adaptive SSMA fusion technique.
- We demonstrate how incorporating semantic cues into the motion segmentation network and motion cues into the semantic segmentation network is mutually beneficial

to both tasks.

- We extend the Cityscapes, KITTI and ApolloScape semantic segmentation datasets with pixel-level motion annotations and make them publicly available. This amounts to a total of 44,090 training images and 9027 testing images with both pixel-level semantic and motion annotations.
- Extensive benchmarking of existing semantic motion segmentation techniques on all the three aforementioned datasets with both quantitative and qualitative comparisons.
- Implementations of our proposed semantic motion segmentation architectures are made publicly available at `http://deepmotion.cs.uni-freiburg.de`.

The remainder of this chapter is organized as follows. In Section 6.2.1, we detail the topology of our SMSnet architecture for joint semantic motion segmentation, followed by the architecture of our SMSnet++ model that aims to improve both tasks simultaneously in Section 6.2.2. Subsequently, we describe the ego-flow suppression technique in Section 6.2.3 and we describe the datasets that we introduce in Section 6.3. In Section 6.4, we present comprehensive experimental evaluations, detailed ablation studies and qualitative comparisons on each of the datasets, followed by detailed generalization analysis of our models in Chapter 6.4.6. Finally, we discuss the extensive related work on motion segmentation and joint semantic motion segmentation in Section 6.5, before concluding the chapter in Section 6.6.

## 6.2 Technical Approach

In this section, we first formulate the problem of semantic motion segmentation. We then describe our proposed SMSnet architecture that jointly predicts both the semantic object class and motion status of each pixel in the image. Subsequently, we detail our improved SMSnet++ architecture that uses semantic cues to generate a more precise motion segmentation and correspondingly uses motion cues to generate a more precise semantic segmentation. Finally, we detail our ego-flow suppression technique that compensates for the flow magnitudes induced due to the ego-motion of the robot.

We represent the training set for semantic motion segmentation as $\mathcal{T} = \{(I_{n-1}, I_n, M_n) \mid n = 1, \ldots, N\}$, where $I_n = \{u_r \mid r = 1, \ldots, \rho\}$ denotes the input frame, $I_{n-1}$ denotes the preceding frame and the corresponding groundtruth label is given by $M_n = \{m_r \mid r = 1, \ldots, \rho\}$, where $m_r \in \mathcal{C} \times \mathcal{M}$, where $\mathcal{C} = \{1, ..., C\}$ is the set of $C$ semantic object classes and each class can also take the label of static or moving $\mathcal{M} = \{m_1, m_2\}$ with $m_1$ denoting a static pixel and $m_2$ denoting a moving pixel. We assume that the input images $(I_{n-1}, I_n)$ and the labels $M_n$ have the same dimensions $\rho = H \times W$. Let $\theta$ be the network parameters consisting of weights and biases, and $s_j(u_r, \theta)$ as the score assigned for labeling pixel $u_r$ with label $j$. We obtain probabilities $\mathbf{P} = (p_1, \ldots, p_{2C})$ for all the semantic classes using the softmax function as

$$p_j(u_r, \theta \mid I_{n-1}, I_n) = \sigma\left(s_j(u_r, \theta)\right) = \frac{exp\left(s_j(u_r, \theta)\right)}{\sum_k^{2C} exp\left(s_k(u_r, \theta)\right)}. \tag{6.1}$$

The optimal network parameters are then estimated by minimizing the cross-entropy loss function as

$$\mathcal{L}_{seg}(\mathcal{T}, \theta) = -\sum_{n=1}^{N}\sum_{r=1}^{\rho}\sum_{j=1}^{2C} \delta_{m_r,j} \log p_j(u_r, \theta \mid I_{n-1}, I_n), \tag{6.2}$$

for $(I_{n-1}, I_n, M_n) \in \mathcal{T}$, where $\delta_{m_r,j}$ is the Kronecker delta.

### 6.2.1 SMSnet Architecture

Our proposed SMSnet architecture shown in Figure 6.2 consists of three streams: *Motion Feature Learning*, *Semantic Feature Learning*, and *Semantic-Motion Feature Fusion*. The Motion Feature Learning stream shown as green blocks learns optical flow field features, while the Semantic Feature Learning stream shown as gray blocks learns to segment the scene into distinct semantic object categories at the pixel-level. The feature maps from both streams are then concatenated in the Semantic-Motion Feature fusion stream indicated as orange blocks and further discriminative motion representations are learned to generate the pixel-wise semantic motion segmentation output. In the rest of this section, we describe the topologies of each of these streams in detail.

**Motion Feature Learning:** This stream learns optical flow field features that represent motion-specific information. Two consecutive input images are first passed through a section of this stream that learns to generate high quality optical flow maps. We embed the recently proposed FlowNet2 [225] architecture for end-to-end learning of optical flow, however, any network that performs optical flow estimation can be employed in place. FlowNet2 consists of multiple stacked subnetworks, where the subsequent networks take the first image, the second image warped with the intermediate optical flow and the intermediate optical flow itself as input, along with the brightness error. The brightness error is the difference between the first and second image warped with the intermediate flow. Moreover, it also includes a subnetwork that focuses on small subpixel motions. The final optical flow is obtained using a small fusion network that combines both the small and large displacement flows. We use the optical flow generated by this network in the $x$ and $y$ direction, and in addition we also compute the magnitude of the flow. This yields a three-channel image with the same dimensions as the input RGB images. Figure 6.1 (c) shows an example of the generated optical flow image, while the consecutive input frames are shown in Figures 6.1 (a) and (b).

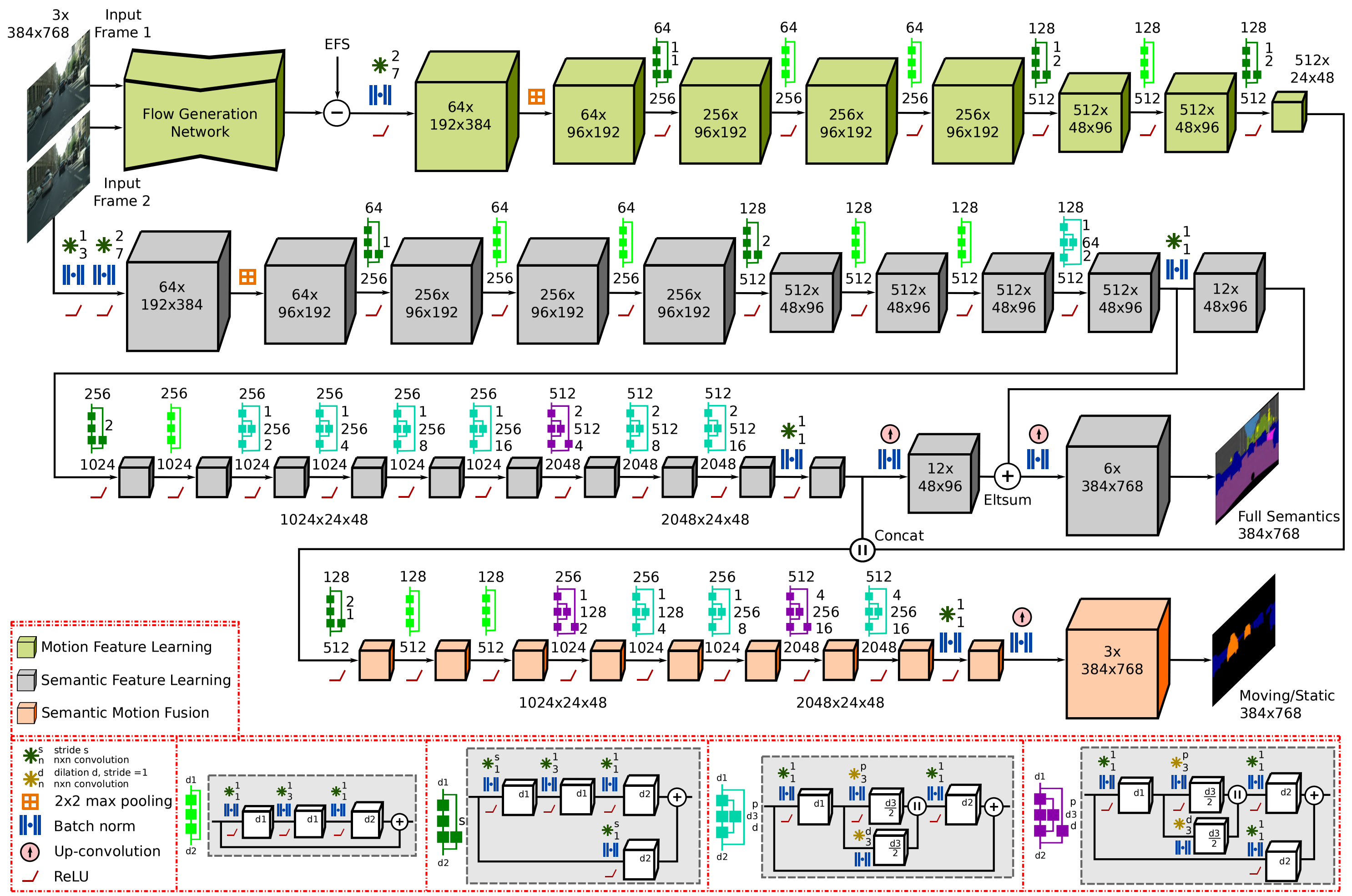


**Figure 6.2:** Topology of our proposed SMSnet architecture for semantic motion segmentation. The stream shown in green learns deep optical flow field features and the stream in gray learns semantic features, which are then both concatenated and fused representations are further learned in the stream depicted in orange. The legend enclosed in red lines show the various residual bottleneck units that we use in our architecture.

Moving objects appear as motion patterns that differ in scale, geometry and magnitude. In order to enable the network to reason about object classes and its borders, we further convolve and pool the optical flow features through multiple network blocks. We utilize the residual blocks [27] from the ResNet-50 architecture for this purpose. The concatenated optical flow and its magnitude is first passed through *Res1* and *Res2* blocks which contain one and four residual units respectively. This downsamples the feature maps to 1/4-times the input image resolution. We then subsequently feed these feature maps through two units of *Res3* and one unit of *Res4* blocks to yield feature map tensors that are 1/16-times the input image resolution. The resulting flow field features rich in motion specific information are then passed as input to the Semantic-Motion Feature Fusion stream.

**Semantic Feature Learning:** In order to learn semantic representations, we embed our AdapNet architecture that we introduced in Chapter 4. The architecture shown as gray blocks in Figure 6.2, takes the current image $I_t$ as input and learns to segment the image into semantic object categories at the pixel-level. The architecture follows the general encoder-decoder design principle, where the encoder learns discriminative deep semantic representations, while decreasing the spatial resolution of the feature maps. Subsequently, the decoder upsamples these feature maps back to the input image resolution using deconvolution layers and skip refinement stages that fuse low-resolution encoder features into the decoder for object boundary refinement. As the AdapNet architecture incorporates multiscale residual units, it enables the network to learn scale invariant deep features and captures multiscale context efficiently, without increasing the number of parameters. With the aim of improving the performance of the motion segmentation using the semantic cues learned by the network, we leverage the semantically rich low-resolution features from the last convolution layer of the encoder and fuse them along with the learned optical flow field features in the Semantic-Motion Feature Fusion stream that follows. Simultaneously, the decoder upsamples the low-resolution encoder features to yield the full semantic segmentation of the scene at the same resolution as the input image.

**Semantic-Motion Feature Fusion:** The final stream in the SMSnet architecture depicted as orange blocks in Figure 6.2, first concatenates the learned optical flow field features with the semantic features which are generated in the aforementioned network streams. The concatenated feature maps are then passed through two residual blocks that resemble the *Res4* and *Res5* blocks of the ResNet-50 architecture. This enables the network to further learn discriminative motion features accounting for the diverse motion patterns in the flow field using the incorporated semantic cues. We employ our multiscale residual units in these blocks with varying dilation rates to increase the size of the receptive field and aggregate information over different fields of view. The large receptive fields capture both the moving object and the background so that the network has sufficient context to discern which parts of the region belongs to the moving object. Note that we do not downsample the features further at this stage, therefore the feature maps are 1/16-times the input image resolution. Finally, towards the end of this stream, we use a $1 \times 1$ convolution layer to reduce the number of feature channels to two in order to represent the static and moving object classes. Subsequently, we upsample the coarse representations back to the input image resolution using a deconvolution layer. The pixels in this upsampled output denote the motion status of the semantic object class predicted by the complementary semantic stream. Figure 6.1 (e) shows the final output of the SMSnet which contains both the predicted semantic labels and the corresponding motion labels.

### 6.2.2 SMSnet++ Architecture

In this section, we describe our SMSnet++ architecture that aims to improve motion estimation using learned semantic cues and simultaneously improve semantic segmentation using learned motion cues. The architecture follows the similar three stream principle as SMSnet consisting of *Motion Feature Learning*, *Semantic Feature Learning*, and *Semantic-Motion Feature Fusion*. Unlike SMSnet, SMSnet++ exploits the temporal coherence across consecutive input images to improve the semantic segmentation performance. This is achieved by warping intermediate network representations of the previous frame into the corresponding representation of the current frame using an edge-enhanced optical flow representation computed using our ego-flow suppressed optical flow with the NetWarp module [238] and dynamically fusing them with the representations of the current frame using our adaptive SSMA fusion technique. Simultaneously, we improve the performance of the motion segmentation using our SSMA module to combine the motion and semantic features, in addition to improving the granularity of segmentation using a multi-stage refinement strategy. In the rest of this section, we detail each of these network components.

**Motion Feature Learning:** In order to learn optical flow field features from consecutive images, we employ the FlowNet3 architecture as shown in Figure 6.3. The FlowNet3 architecture improves optical flow prediction by building upon FlowNet2 and incorporating occlusion estimation as it often negatively influences correspondence estimation tasks. FlowNet3 consists of multiple stacked encoder-decoder networks, where the first network takes consecutive monocular images as input and the subsequent networks take the first image, the second image warped with the intermediate optical flow, the intermediate optical flow and the occlusions as input. Unlike in FlowNet2, the brightness error is omitted from the inputs. In SMSnet, we rescaled the output optical flow to the 0-255 range and rounded it to the nearest integer before extracting motion features. However, in SMSnet++, we maintain the pixel displacements as floating point numbers and rescale them to the 0-255 range in order to improve the quality of the subsequently learned motion features. We present experiments that demonstrate the improvement due to this transformation in the ablation study presented in Section 6.5. We obtain the optical flow maps from the

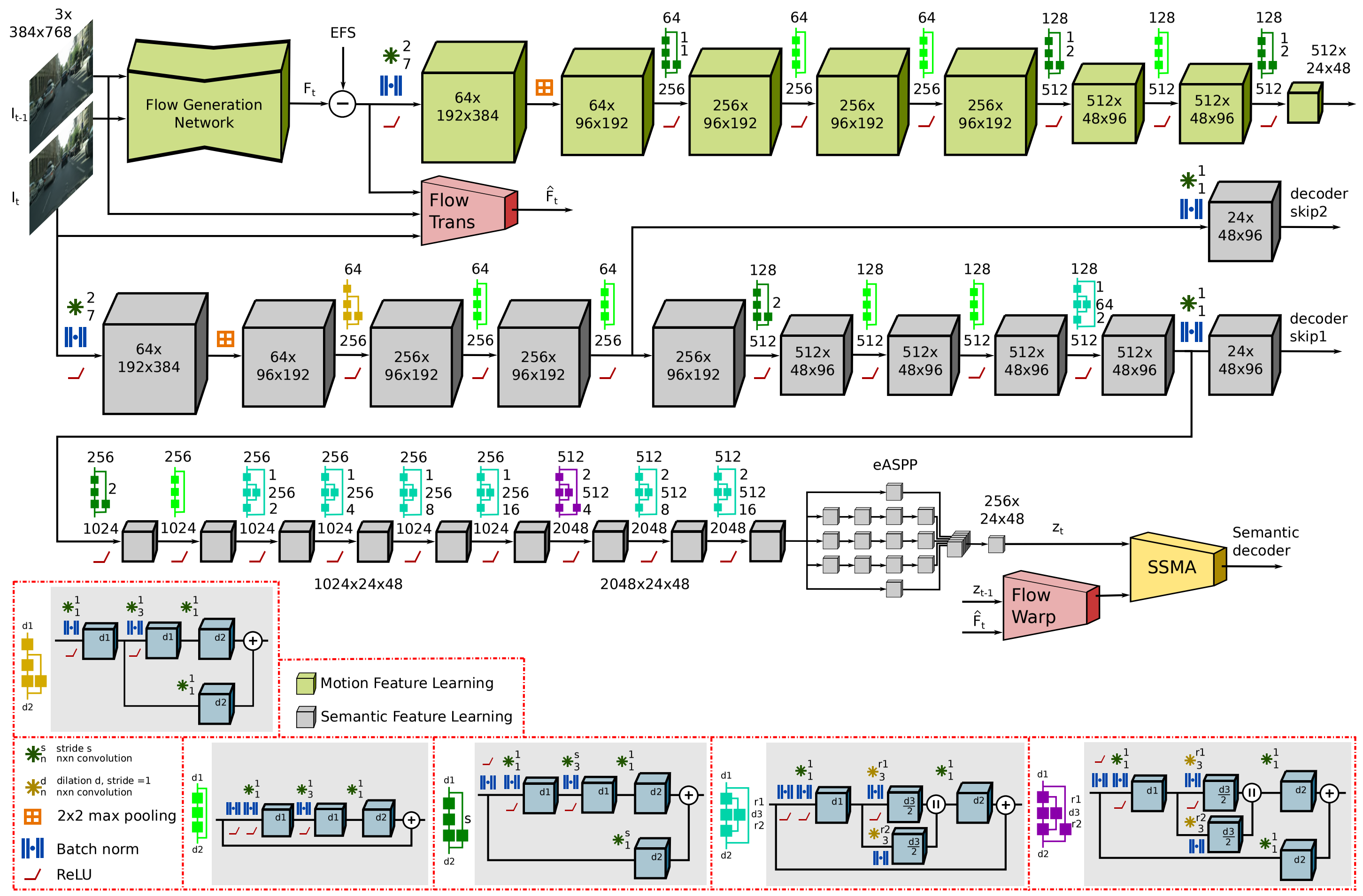


**Figure 6.3:** Topology of the motion and semantic feature learning streams of our proposed SMSnet++ architecture. The stream shown in green learns optical flow field features and the stream in gray learns semantic features, while transforming the optical flow to an edge-enhanced representation and employing it for warping features $z_{t_1}$ of the previous frame into the current frame, followed by subsequently fusing them with the features of the current frame $z_t$. The legend enclosed in red lines show the various full pre-activation residual bottleneck units that we use in our architecture.

embedded FlowNet3 network and pass the features through a series of convolution and pooling layers, similar to the SMSnet architecture, but we employ the residual units from the full pre-activation ResNet-50 architecture [72] as opposed to the standard ResNet-50 [27]. The full pre-activation residual units are shown in the legend enclosed in a red box in Figure 6.3, while the original residual units are shown in the legend in Figure 6.2. The feature maps at the output of this Motion Feature Learning stream which are 1/16-times downsampled with respect to the input image resolution are then used for fusion in the Semantic-Motion Feature Fusion stream.

**Semantic Feature Learning:** We employ our recently proposed AdapNet++ architecture that we presented in Chapter 4 for learning the semantics in the SMSnet++ architecture. The architecture builds upon AdapNet and incorporates our efficient Atrous Spatial Pyramid Pooling (eASPP) module that captures long-range context with a large effective receptive field by cascading multiple atrous convolutions. The size of the receptive field is a critical factor for motion segmentation tasks. If a filter with a small receptive field falls entirely within an object with non-zero flow values, it is impossible to identify if the flow is due to the moving object or due to the motion of the robot equipped with the camera. Whereas, if the receptive field includes a portion of the background then the network can easily identify the moving regions from the change in flow across the dynamic object boundaries. As objects in driving scenarios have multiple scales, our eASPP effectively aggregates multiscale features using atrous convolutions with different dilation rates in parallel. We also reconfigure our multiscale residual units that we incorporate in the encoder according to the full pre-activation configuration. The two multiscale residual units are shown in the bottom right of the legend enclosed in a red box in Figure 6.3.

Additionally, to further improve the semantic segmentation performance, we propose a technique to first enhance the optical flow generated by the complementary Motion Feature Learning stream using the NetWarp module [238] to a more discriminative representation with sharper object boundaries. We then utilize this edge-enhanced flow to warp intermediate semantic network representations from the previous frame into the current frame, followed by fusing them with the representations of the current frame. We show examples of the edge-enhanced flow and evaluate its utility for the temporal warping in comparison to directly using the optical flow in Section 6.4.4.3. This principle of combining temporally close representations learned by the network to improve the semantic consistency across frames has previously been explored in different ways [238, 241, 242]. However, in this work, we propose to exploit the pixel correspondences learned by our Motion Feature Learning stream to dynamically fuse intermediate network representations to improve the temporal consistency.

The NetWarp module [238] that computes the edge-enhanced flow shown in Figure 6.4 takes the consecutive images $I_{t-1}, I_t$ and the predicted optical flow $F_t$ as input, and outputs an edge-enhanced flow $\hat{F}_t$. Let $z_{t-1}^k$ and $z_t^k$ be the representations of the $k^{th}$ layer of the

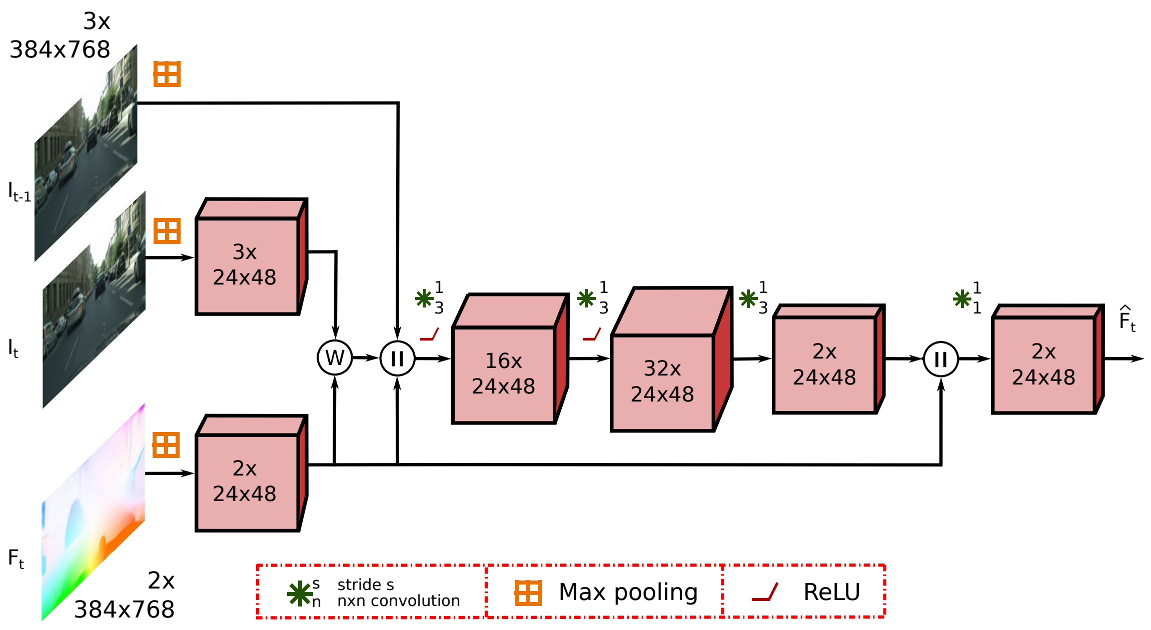


**Figure 6.4:** Topology of the NetWarp module [238] that computes the edge-enhanced flow to emphasize the object boundaries. The resulting edge-enhanced flow is then utilized to warp intermediate network representations from the previous frame into the current frame. The symbol W denotes warping operation and || denotes concatenation along the channel dimension.

network for the input frames $I_{t-1}$ and $I_t$ respectively. The network first downsamples each of the inputs using max pooling layers to match the spatial dimensions of the tensor at layer $k$ that is to be warped. As identified in the ablation study that we present in Section 6.4.4.3, we warp the intermediate representations at the end of the encoder after the eASPP, as they are semantically more meaningful than the low-level representations from earlier layers. Therefore, the pooling layers downsample the inputs to 1/16-times the input image resolution. The downsampled current image is then warped with the downsampled edge-enhanced flow, followed by concatenating the resulting tensor with the predicted flow and the previous image. The concatenated feature maps are then passed through three $3 \times 3$ convolution layers with $16, 32$ and $2$ feature channels respectively. The resulting convolved tensor is then concatenated with the original predicted flow and passed through a $1 \times 1$ convolution layer to reduce the number of feature channels to two and yield the edge-enhanced flow $\hat{F}_t$. We employ the NetWarp module [238] after the ego-flow subtraction from the embedded flow subnetwork and in parallel to the Motion Feature Learning stream as shown in Figure 6.3. We then warp the representations of the previous frame $z^k_{t-1}$ to align with the representations of the current frame $z^k_t$ using the edge-enhanced flow $\hat{F}_t$ to obtain the warped representation $\hat{z}^k_{t-1}$ as demonstrated in the work of Gadde *et al.* [238] as

$$\hat{z}^k_{t-1} = Warp\left(z^k_{t-1}, \hat{F}\right). \tag{6.3}$$

More formally, to compute the warped representation $\hat{z}^k_{t-1}$ at the pixel location $(x, y)$ of the current frame $I_t$ mapped to the spatial locations $(x', y')$ of the previous frame $I_{t-1}$, we implement the warping function as a bilinear interpolation of $z^k_{t-1}$ at the desired coordinates $(x', y')$. Let $(x_1, y_1), (x_1, y_2), (x_2, y_1)$ and $(x_2, y_2)$ be the corner coordinates of the previous frame's grid cell where $(x', y')$ lies. The warping of $z^k_{t-1}$ to obtain $\hat{z}^k_{t-1}(x, y)$ is computed as defined in the work of Gadde *et al.* [238] as

$$\hat{z}^k_{t-1}(x, y) = z^k_{t-1}(x', y') = \frac{1}{\eta} \begin{bmatrix} x_2 - x' \\ x' - x_1 \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathbf{z}^k_{t-1}(x_1, y_1) & \mathbf{z}^k_{t-1}(x_1, y_2) \\ \mathbf{z}^k_{t-1}(x_2, y_1) & \mathbf{z}^k_{t-1}(x_2, y_2) \end{bmatrix} \begin{bmatrix} y_2 - y' \\ y' - y_1 \end{bmatrix}, \tag{6.4}$$

where $\eta = 1/(x_2 - x_1)(y_2 - y_1)$. If $(x', y')$ lies outside the spatial area of $z^k_{t-1}$, we then backproject it to the nearest border of $z^k_{t-1}$. In some cases when the flow values are integers, the warping function is non-differentiable as the corner coordinates used in the interpolation suddenly change when $(x', y')$ moves across one grid cell to another. In order to circumvent this problem, we add a small $\epsilon = 0.0001$ to the flow transform to make the warping function always differentiable. In order to fuse the warped representation $\hat{z}^k_{t-1}$ with the representation $z^k_t$ of the current frame, we employ our SSMA fusion module that we presented in Chapter 5. We reconfigure our SSMA module to adaptively recalibrate the feature maps by weighing $\hat{z}^k_{t-1}$ and $z^k_t$ channel-wise according to the scene context.

**Semantic-Motion Feature Fusion:** The topologies of the streams that we presented thus far enable our network to learn highly discriminative semantic features and motion features. In order to upsample these feature maps to a high resolution semantic motion segmentation output, we employ two individual decoders as shown in Figure 6.5. We incorporate the decoder of our AdapNet++ architecture that we presented in Section 4 to upsample the semantic feature maps back to the input image resolution. To obtain the pixel-level motion status, we first employ our SSMA fusion module that we presented in Chapter 5 to adaptively fuse the semantic features from the encoder with learned motion features. We utilize our SSMA module for the fusion, rather than simple concatenation that we used in the SMSnet architecture in order to adaptively only fuse the relevant semantic features that aid in better motion estimation. We demonstrate utility of employing our SSMA fusion module in the ablation study presented in Section 6.4.4.1. We then convolve the fused semantic-motion features through the *Res4* and *Res5* blocks of the full preactivated ResNet-50 architecture [72] incorporating our multiscale residual units with different dilation rates to further capture multiscale context. The multiscale residual units are shown in the bottom right two blocks of the legend closed in a red box, while the corresponding standard residual units are shown in bottom left two blocks in Figure 6.5. The representations at the end of this stage where they are 16-times downsampled with respect to the input image, are upsampled using the AdapNet++ decoder, similar to the semantic decoder stream. The output of this is stream consists of pixel-level motion labels corresponding to the semantic object classes predicted by the parallel semantic decoder.

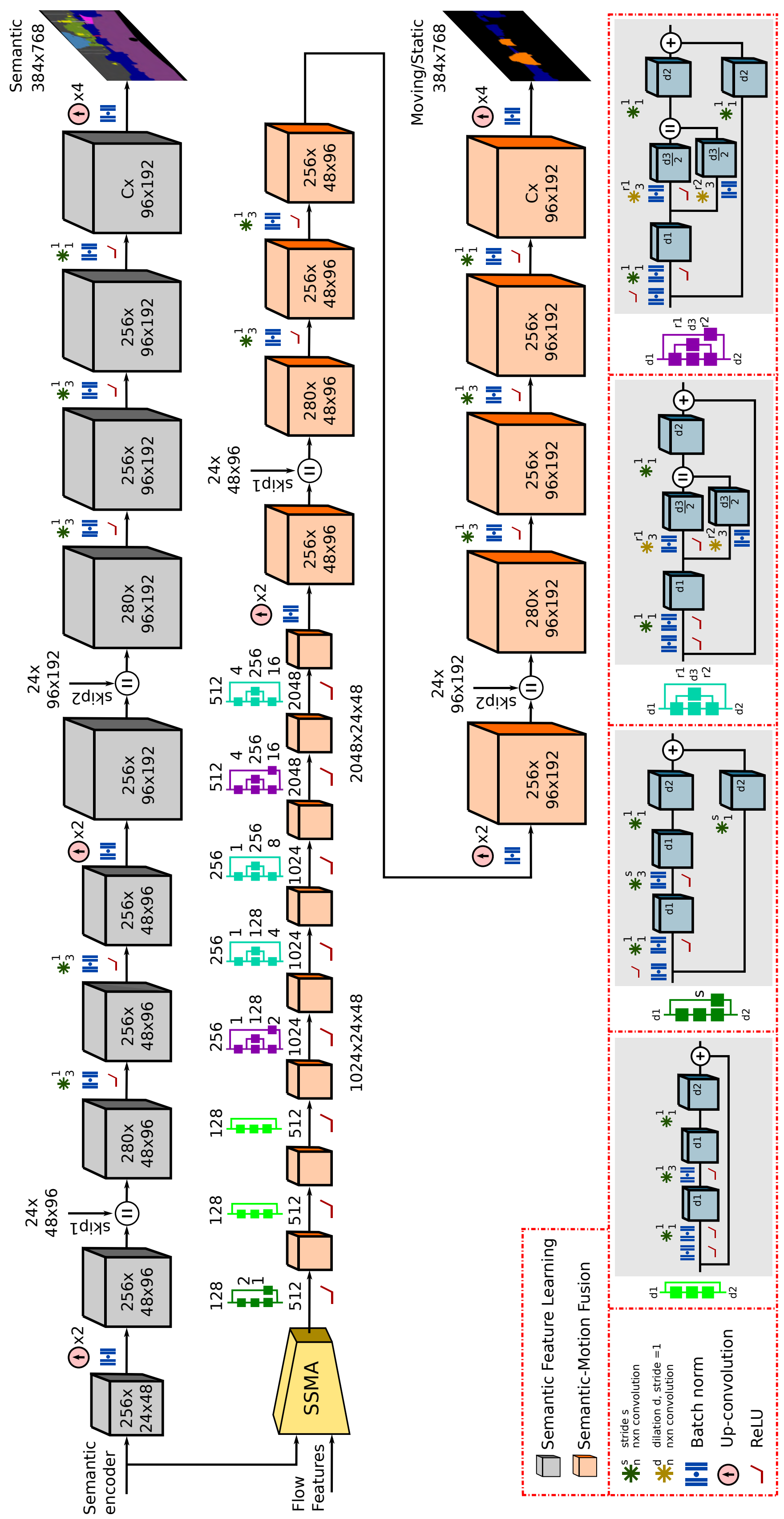


**Figure 6.5:** Topology of the motion and semantic decoder streams of our proposed SMSnet++ architecture that upsample the feature maps back to the input image resolution. We fuse the semantic features with the learned motion features using our adaptive SSMA fusion module presented in Chapter 5. The fused features are then further convolved through a series of multiscale residual units and upsampled to a high-resolution output using deconvolution layers and skip refinement stages. The legend enclosed in red lines show the various full pre-activation residual bottleneck units that we use in our architecture. The symbol ‖ denotes concatenation along the channel dimension and + denotes the element-wise addition.

### 6.2.3 Ego-Flow Suppression

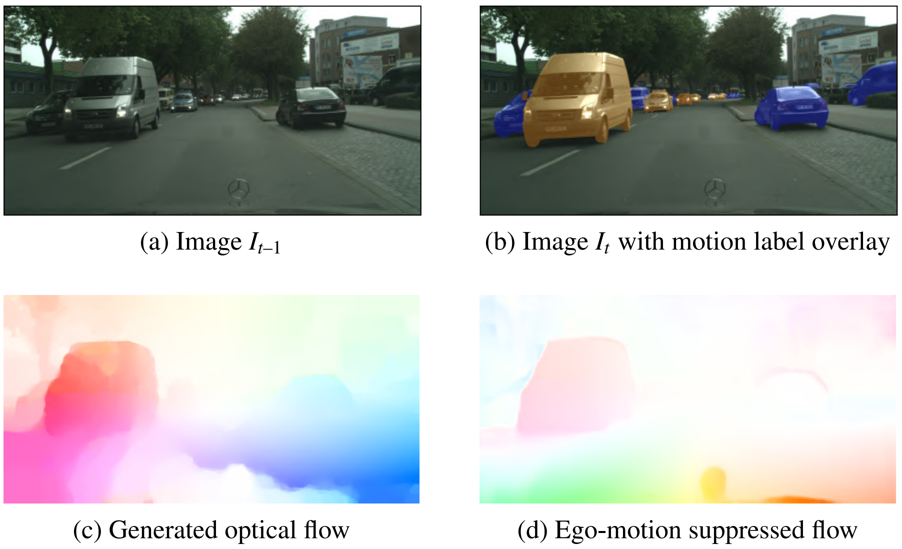
As the robot equipped with the camera is itself in motion during navigation, the resulting ego-motion introduces additional optical flow magnitudes that are not induced by moving objects in the scene. This hampers the network from distinguishing the real motion of objects from the camera motion. The flow induced from ego-motion can cause ambiguities since objects in the scene can appear with high optical flow magnitudes although they are not moving in reality. In order to circumvent this problem, we predict the optical flow map $\hat{X}'$ purely caused by the ego-motion and subtract it from the predicted optical flow. We first estimate the backward camera translation $T$ and the rotation matrix $R$ from the position of the current frame $I_t$ to the previous frame $I_{t-1}$. Using the IMU and odometry data, we can then estimate $\hat{X}'$ as

$$\hat{X}' = KRK^{-1}X + K\frac{T}{z}, \tag{6.5}$$

where $K$ is the intrinsic camera matrix, $X = (u, v, 1)^T$ is the homogenous coordinate of the pixel in the image coordinates and $z$ is the depth of the corresponding pixel in meters. By computing the flow vector for every pixel coordinate using Eq. (6.5), we obtain the 2-dimensional optical flow that purely represents the ego-motion, which we denote as the ego-flow. As Eq. (6.5) requires the depth map of the corresponding input images, we estimate the depth $z$ using the recently proposed DispNet [243] architecture that has a fast inference time. We then subtract the ego-flow $\hat{X}'$ from the predicted optical flow $\hat{X}$ right after the flow generation network in our proposed SMSnet and SMSnet++ architectures as shown in Figure 6.2. We denote this step as Ego-Flow Suppression (EFS) in this work. This enables the network to account for the induced ego-flow while maintaining the flow magnitudes that are caused by the other moving objects. We demonstrate the utility of introducing EFS in the ablation study presented in Section 6.4.4.2. An example of the optical flow with ego-flow suppression EFS is shown in Figure 6.1 (d). We present extensive ablation studies with and without the EFS in Section 6.4.4.

## 6.3 Dataset and Augmentation

Training our proposed networks require a large amount of labeled semantic and motion annotations. Although there are a handful of approaches that have tackled the problem of semantic motion segmentation, all of them have only utilized the 200 labeled images from the KITTI dataset that were hand-annotated by Reddy *et al.* [239]. Employing data augmentation strategies can alleviate this problem to a certain extent. However, for credible quantitative evaluations, several hundreds of training images with groundtruth labels are required, apart from the separate set of images and groundtruth labels for testing. Standard motion segmentation datasets such as DAVIS [244], Sintel [245] and FBMS [246] do not have semantic groundtruth annotations, while standard semantic scene segmentation


(a) Image $I_{t-1}$

(b) Image $I_t$ with motion label overlay

(c) Generated optical flow

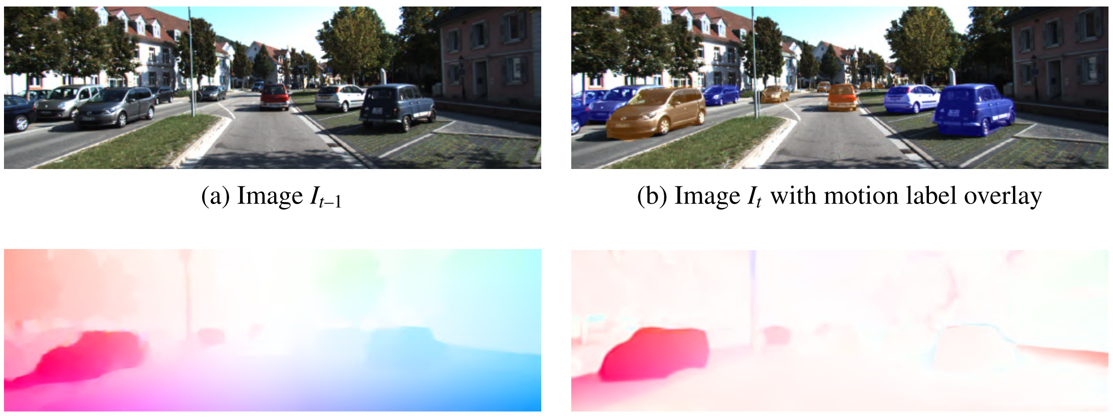
(d) Ego-motion suppressed flow

**Figure 6.6:** An example scene from the Cityscapes dataset showing a pair of consecutive images with the correspondingly generated optical flow field and the ego-motion compensated flow field. The image from the current timestep $I_t$ also shows the groundtruth static (blue) and moving (orange) object labels as an overlay.

datasets such as Cityscapes[143], SUN RGB-D [145] and ScanNet [146] do not have motion annotations.

Training our network for joint semantic motion segmentation requires semantic and motion annotations for the same scenes. Obtaining pixel-level groundtruth of object motion is particularly hard as visible pixel displacement quickly decreases with increasing distance from the camera due to the motion parallax effect. In addition, ego-motion of the vehicle makes labeling an arduous task. In order to facilitate this work and to allow for credible quantitative evaluation, we extend three benchmark scene understanding datasets with manually annotated motion labels. We made these annotations publicly available at `http://deepmotion.cs.uni-freiburg.de` to enable further progress in learning this joint task. In the rest of this section, we briefly describe these datasets with example scenes contained in them and the annotation procedure that we employ.

**Cityscapes-Motion:** The Cityscapes dataset [143] is one of the standard benchmarks for scene understanding, that we also employed for semantic segmentation and multimodal fusion in Chapters 4 and 5 correspondingly. As we already presented an extensive description of this dataset in the previous chapters, we only give a brief overview of the motion annotations that we generated. We manually labeled the dynamic objects with motion status by observing consecutive images from the video sequences and by assigning the semantic instance annotations of the objects provided in the dataset with *static* or *moving* motion tags. We labeled a total of 2975 training images and 500 validation images. The



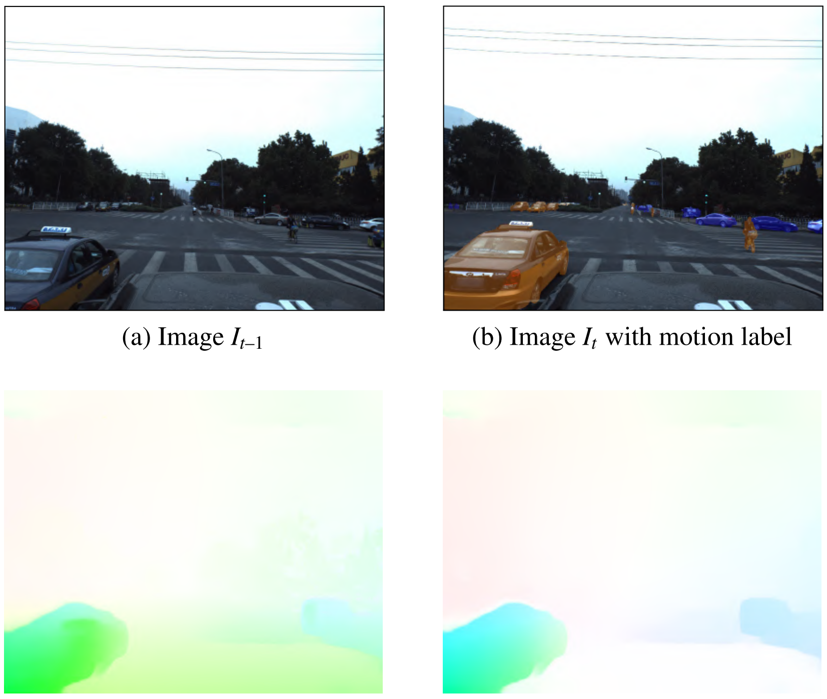
(a) Image $I_{t-1}$ (b) Image $I_t$ with motion label overlay

(c) Generated optical flow (d) Ego-motion suppressed flow

**Figure 6.7:** An example scene from the KITTI dataset showing a pair of consecutive images with the correspondingly generated optical flow field and the ego-motion compensated flow field. The image from the current timestep $I_t$ also shows the groundtruth static (blue) and moving (orange) object labels as an overlay.

Cityscapes dataset is highly challenging for dynamic scene understanding as it contains many dynamic objects including *person, car, truck, bus, tram, motorcycle, bicycle, caravan* and *trailer*. However, we only labeled the category of cars on this dataset with motion annotations. We plan to extend the annotations to other dynamic object categories in the future. Figure 6.6 shows an example scene with the RGB images of the current frame $I_t$ and the preceding frame $I_{t-1}$, the groundtruth motion label overlaid on the current frame, the computed optical flow and the corresponding flow with EFS applied. Note that we compute the reverse flow mapping from the current frame $I_t$ with semantic motion annotations to the previous frame $I_{t-1}$.

**KITTI-Motion:** The KITTI vision benchmark suite [247] contains 200 training images and 200 testing images with semantic groundtruth annotations, however, it does not provide any moving object annotations. Existing techniques that address the problem of semantic motion segmentation have been benchmarked on the 200 motion annotations created by Reddy *et al.* [239], yet no training data is available with both semantic and motion groundtruth labels. Therefore, we introduce the KITTI-Motion dataset consisting of 255 training and 200 testing images with manually annotated pixel-level semantic as well as motion labels. The images were captured at a resolution of 1280 × 384 pixels in a diverse set of driving scenarios including freeways, residential and inner-city areas. This dataset contains a large amount of moving and parked cars, often partially occluded due to trees or street lamps, making moving object segmentation equally challenging as the Cityscapes dataset. We manually annotated the 255 training images with pixel-level semantic class labels for the same object categories that are present in the Cityscapes dataset



(a) Image $I_{t-1}$ (b) Image $I_t$ with motion label

(c) Generated optical flow (d) Ego-motion suppressed flow

**Figure 6.8:** An example scene from the ApolloScape dataset showing a pair of consecutive images with the correspondingly generated optical flow field and the ego-motion compensated flow field. The image from the current timestep $I_t$ also shows the groundtruth static (blue) and moving (orange) object labels as an overlay.

and additionally with moving object annotations for cars. The 200 testing images are taken from the dataset provided by Reddy *et al.*. In addition, we combine two community annotated KITTI semantic segmentation datasets [248, 249] consisting of 253 images for pretraining the semantic stream of our network. These images also do not overlap with the training or testing images in the KITTI-Motion dataset that we introduced. Figure 6.7 shows an example scene from this dataset with the preceding frame $I_{t-1}$, the groundtruth motion label overlaid on the current frame $I_t$, the optical flow map that was generated and the corresponding flow with EFS.

**ApolloScape-Motion:** ApolloScape [240] is one of the most recent and largest scene understanding datasets till date. The images were captured using two VMX-CS6 cameras at a resolution 3384 $\times$ 2710 pixels and the depth maps were generated from laser measurements acquired using a Riegl VMX-1HA scanner. All the images were tagged with IMU/GNSS measurements and were time synchronized. A total of 143,906 video frames and corresponding pixel-level semantic annotations were released, including 89,430 instance-level semantic object annotations. The dataset was collected in easy, moderate, and heavy scene complexities, where the complexity is computed from the amount of movable objects such as person and vehicles in the scene. This dataset is substantially more challenging than Cityscapes and KITTI datasets, as it contains extreme lighting conditions caused by driving under overpasses, glare on the camera optics and reflections of multiple nearby vehicles that are visible on other vehicles. Semantic annotations are provide for 21 object classes including *car, motorcycle, bicycle, person, rider, truck, bus, tricycle, road, sidewalk, traffic cone, bollard, fence, traffic light, pole, traffic sign, wall, trash can, billboard, building, bridge, tunnel, overpass* and *vegetation*.

Depth maps have only been released for a subset of the labeled images. Therefore, we chose the images for which the depth maps are available for our ApolloScape-Motion dataset, which corresponds to 40,960 training images and 8327 testing images. Similar to the Cityscapes and KITTI datasets, we manually annotated each of these images with pixel-level motion labels. Unlike the other datasets where only *cars, trucks* and *buses* were annotated with motion labels, in this dataset, we annotated all the different types of dynamic objects, thereby individually classifying static/moving car, static/moving person and static/moving cyclist. We use the same semantic object classes as the Cityscapes and KITTI datasets for consistency and in order to be able to combine the datasets. Figure 6.8 shows an example scene from the ApolloScape dataset with the previous image $I_{t-1}$, the current image $I_t$, the current image with the groundtruth motion labels overlaid, the generated optical flow image and the corresponding optical flow with EFS.

Despite the reasonable amount of training data, we perform a series of data augmentations to introduce more diversity into the training set. Data augmentation is critical to prevent overfitting and to enhance the generalization ability of the network. We apply spatial transformations such as rotation (–13° to 13°), scaling (0.5 to 2.0), flipping, translation (–20% to +20%) and cropping (0.8 to 0.9), as well as chromatic transformations such as color (0.5 to 2), contrast (0.5 to 1.5) and brightness (–40 to 40) modulation. As our networks take two consecutive images as input, we augment the pair and the corresponding groundtruth label jointly with the same parameters.

## 6.4 Experimental Evaluation

In this section, we first describe the training protocol that we employ in Section 6.4.1, followed by comprehensive quantitative results for semantic motion segmentation using our proposed architectures in Section 6.4.2 and an analysis on the influence of motion parallax in Section 6.4.3. We then present detailed ablation studies that describe our architectural decisions in Section 6.4.4 and extensive qualitative semantic motion segmentation results on each of the datasets in Section 6.4.5. Finally, we present evaluations of the generalization ability of our models to new environments in Section 6.4.6.

We use the TensorFlow [159] deep learning library for the implementations and all the experiments were carried out on a system with an Intel Xeon E5, 2.4 GHz and an NVIDIA

TITAN X GPU. We quantify the performance using the standard Jaccard Index which is commonly known as average intersection-over-union (IoU) metric. It can be computed for each object class as $IoU = TP/(TP + FP + FN)$, where $TP$, $FP$ and $FN$ correspond to true positives, false positives and false negatives respectively. We also report the mean intersection-over-union (mIoU), pixel-wise accuracy (Acc) and average precision (AP) for the empirical analysis.

### 6.4.1 Network Training

We train our networks with an input image resolution of $768 \times 384$ pixels. We use bilinear interpolation for resizing the RGB images and the nearest-neighbour interpolation for resizing the groundtruth labels. We employ the same multi-stage training protocol to effectively train both our SMSnet and SMSnet++ architectures. We first train the Semantic Feature Learning stream of SMSnet and SMSnet++ that encompass the AdapNet and AdapNet++ architectures respectively for learning the semantic features corresponding to the $C$ number of object classes in the datasets. We use the training scheme described in Section 4.4.3 of Chapter 4 for this purpose. Subsequently, in the second stage, we leverage transfer learning to train the joint model in either the SMSnet or the SMSnet++ framework by initializing the Semantic Feature Learning stream with weights from the previous stage and the embedded optical flow generation network in the Motion Feature Learning stream with weights pre-trained on the KITTI Flow dataset [247]. While, we initialize the other layers using the He initialization [93] scheme.

We train the entire joint architecture while keeping the weights of the Semantic Feature Learning stream and the flow generation network fixed. This enables our joint architecture to exploit the stable optical flow field features and semantic features, while learning deeper discriminative motion representations at the high-level in the second stage. We use the Adam solver for optimization with $\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-10}$. We train the joint model for a maximum of 100,000 iterations using an initial learning rate of $\lambda_0 = 10^{-4}$ with a mini-batch size of 8 and a dropout probability of 0.5. In the final stage, we fix the weights of the Semantic Feature Learning stream, the Motion Feature Learning stream and the layers in the Semantic-Motion Feature Fusion until the first deconvolution layer, while only training the decoder for the motion segmentation with a learning rate of $\lambda_0 = 10^{-5}$ and a mini-batch size of 12 for 50,000 iterations. This enables us to train the network with a larger batch size, while focusing more on the upsampling stages to yield the high-resolution motion segmentation output.

### 6.4.2 Comparison with the State-of-the-Art

In this section, we present comprehensive benchmarking results of our proposed SMSnet and SMSnet++ architectures on the Cityscapes-Motion, KITTI-Motion and Apolloscape-

Motion datasets. We compare the performance of our models with the state-of-the-art semantic motion segmentation architectures including CRF-M [232] and CNN-MCA [237], as well as architectures that only perform end-to-end motion segmentation, namely, MP-net [235], MODNet [250], AHCRF-Motion [234] and GEO-M [251]. Additionally, we also compare against a baseline technique that we term SMSnet++$_{\text{FLO}}$, for which we use the same topology as our proposed SMSnet++ architecture, but we remove the Semantic Feature Learning stream. Therefore, this baseline performs motion segmentation only based on the optical flow and in comparison with the SMSnet++ architecture, it highlights the improvement that our model achieves by encoding semantic features into the motion segmentation stream. Furthermore, we report results of two model variants for each of our proposed networks; one that is trained on a label set consisting of all the semantic object classes in the dataset for motion segmentation, regardless of whether the specific object class is movable, and secondly, the model that is trained on the label set consisting of only the movable semantic objects classes in the dataset for motion segmentation. Intuitively, providing the network with this prior about the movable objects will help the network learn the diverse motion patters more accurately, as instead of considering objects such as *building* and *fence* that would never move in reality, it can focus more on learning the motion patters of movable objects such as *car* and *person*. We denote the model that is trained on the label set containing all the semantic objects as movable with the suffix *FULL* and the model that is trained on the label set consisting of only the movable semantic objects for motion segmentation with the suffix *MOV*. In the ablation study presented in Section 6.4.4.5, we demonstrate performance comparison of models trained on both these label sets for different moving object distances. As our proposed networks jointly predict both motion segmentation labels as well as semantic segmentation labels, we first present the results for motion segmentation using class-wise IoU scores for the static and moving classes for each of the datasets in Section 6.4.2.1, and subsequently present the class-wise results for semantic segmentation in Section 6.4.2.2.

#### 6.4.2.1 Motion Segmentation

Table 6.1 shows the benchmarking results on the Cityscapes-Motion dataset. Among the existing approaches, the CNN-MCA model achieves the highest mIoU score of 82.23%, followed by MODNet which trails by 1.90%. It should be noted that CNN-MCA jointly predicts both the semantic object class and motion labels, whereas MODnet is purely a motion segmentation network. Our proposed SMSnet model outperforms the state-of-the-art achieving an mIoU score of 85.78%, which constitutes a moving object IoU score of 78.69% and an IoU score of 92.87% for the static class. Furthermore, our SMSnet++ architecture sets the new state-of-the-art on this dataset by achieving a mIoU score of 89.92%, thereby amounting to a large improvement of 7.69% over the previous state-of-the-art CNN-MCA model. Analyzing the individual class IoU scores in comparison to

**Table 6.1:** Comparison of semantic motion segmentation on the Cityscapes-Motion dataset.

| Approach | IoU (%) | | mIoU | Acc. | AP |
|---|---|---|---|---|---|
| | Moving 🟧 | Static 🟦 | (%) | (%) | (%) |
| MPnet [235] | 54.19 | 89.68 | 71.93 | 90.80 | 89.18 |
| MODNet [250] | 69.53 | 92.76 | 81.14 | 93.89 | 92.51 |
| CNN-MCA [237] | 72.27 | 92.18 | 82.23 | 93.78 | 93.16 |
| SMSnet++$_{\text{FLO}}$ | 67.42 | 92.90 | 80.16 | 93.81 | 94.69 |
| SMSnet$_{\text{FULL}}$ (Ours) | 76.56 | 92.38 | 84.47 | 93.90 | 94.20 |
| SMSnet$_{\text{MOV}}$(Ours) | 78.69 | 92.87 | 85.78 | 94.21 | 94.75 |
| SMSnet++$_{\text{FULL}}$ (Ours) | 83.11 | 94.38 | 88.75 | 95.59 | 95.37 |
| SMSnet++$_{\text{MOV}}$ (Ours) | **85.02** | **94.82** | **89.92** | **95.99** | **94.85** |

CNN-MCA, our SMSnet++ model demonstrates a improvement of 12.75% and 2.64% for the moving and static object classes respectively. Although our SMSnet++$_{\text{FULL}}$ model achieves a performance 1.17% lower than SMSnet++$_{\text{MOV}}$, it still substantially outperforms existing methods. Interestingly, the SMSnet++$_{\text{FLO}}$ baseline only achieves a moving object IoU score of 67.42%. This signifies that our SMSnet++ model achieves an improvement of 17.60% by adaptively fusing the semantic features into the motion segmentation stream.

We present benchmarking results on the KITTI-Motion dataset in Table 6.2. All the semantic motion segmentation approaches that have been proposed thus far have been benchmarked on the KITTI-Motion dataset. Similar to the observation that we made for the results on the Cityscapes-Motion dataset, the CNN-MCA model demonstrates the highest performance among the existing methods, achieving a mIoU score of 82.50%, followed by MODNet achieving a mIoU score which is 1.94% lower than the CNN-MCA model. The results shown in Table 6.2 are chronologically ordered and it can be seen that the methods that jointly predict the semantic object class and motion labels such as CNN-MCA and CRF-M substantially outperform motion segmentation approaches that were proposed before them. This can be attributed to the fact that these approaches learn to correlate motion features with the learned semantic representations, which improves the overall motion segmentation accuracy. Intuitively, these methods learn that there is a higher probability of a car moving than a building or a pole. Although Fan *et al.* [233] also propose an approach for semantic motion segmentation, the KITTI scene flow dataset that they evaluate on has inconsistent class labels, which does not allow for a meaningful comparison. Our proposed SMSnet model achieves a mIoU score of 86.78%, which exceeds the state-of-the-art by 4.28%. Furthermore, our SMSnet++ model achieves a higher mIoU of 93.97% and sets the new state-of-the-art on the KITTI-Motion benchmark. This constitutes to a significant improvement of 16.62% in the IoU score for the moving object class and 6.34%

**Table 6.2:** Comparison of semantic motion segmentation on the KITTI-Motion dataset.

| Approach | IoU (%) | | mIoU | Acc. | AP | Time |
|---|---|---|---|---|---|---|
| | Moving 🟧 | Static 🟦 | (%) | (%) | (%) | (ms) |
| GEO-M [251] | 46.50 | 49.80 | 48.15 | N/A | N/A | N/A |
| AHCRF-Motion [234] | 60.20 | 75.80 | 68.00 | N/A | N/A | N/A |
| CRF-M [232] | 73.50 | 82.40 | 77.95 | N/A | N/A | 240000 |
| MPnet [235] | 67.41 | 86.49 | 76.95 | 89.44 | 92.08 | 156 |
| MODNet [250] | 72.68 | 88.45 | 80.56 | 90.96 | 92.94 | 326 |
| CNN-MCA [237] | 75.36 | 89.63 | 82.50 | 91.64 | 93.17 | 413 |
| SMSnet++$_{\text{FLO}}$ | 69.29 | 87.40 | 78.34 | 90.18 | 93.30 | 109 |
| SMSnet$_{\text{FULL}}$ (Ours) | 78.59 | 90.48 | 84.54 | 92.95 | 93.74 | 134 |
| SMSnet$_{\text{MOV}}$ (Ours) | 81.78 | 91.78 | 86.78 | 93.99 | 94.47 | 134 |
| SMSnet++$_{\text{FULL}}$ (Ours) | 90.00 | 94.94 | 92.47 | 96.52 | 92.47 | 176 |
| SMSnet++$_{\text{MOV}}$ (Ours) | **91.98** | **95.97** | **93.97** | **96.78** | **92.53** | **176** |

in the IoU score for the static object class, over the previous state-of-the-art CNN-MCA model. Analyzing the performance of the SMSnet++$_{\text{FLO}}$ model for the moving object segmentation, we observe that our SMSnet++ model achieves an improvement of 22.69% in the IoU score by encoding semantics into the motion segmentation stream. Other performance metrics such as the pixel accuracy (Acc) and average precision (AP) show a similar improvement in comparison to existing methods.

Additionally, Table 6.2 also shows the inference time for each of the models. Fast prediction time is one of the most essential requirements for adoption in real-world robotic applications. Therefore, we designed the topology of our architectures keeping this critical factor in mind. Run-time of existing semantic motion segmentation techniques vary from 4 s for CRF-M to 413 ms for CNN-MCA. Motion segmentation models have a faster run-time than semantic motion segmentation approaches, as the complexity of these models are simpler and they have lesser number of parameters on account of only needing to distinguish between the static and moving pixels. Our proposed SMSnet architecture performs inference in 134 ms which is 67.55% faster than the previous state-of-the-art. While our proposed SMSnet++ architecture consumes 176 ms for the inference, it improves the moving object segmentation performance over SMSnet by 10.20% in the IoU score and still remains over twice as fast as existing semantic motion segmentation architectures.

Finally, we present results on the newer and more challenging ApolloScape-Motion dataset in Table 6.3. Unlike the other datasets that we benchmark on, the ApolloScape-Motion dataset contains moving object annotations for multiple semantic object classes, namely, *car, person* and *cyclist*. Therefore, we report the IoU scores of the individual semantic moving and static object classes, as well as the global Intersection over Union

**Table 6.3:** Comparison of semantic motion segmentation on the ApolloScape-Motion dataset.

| Approach | Moving | | | | Static | | | | mIoU |
|---|---|---|---|---|---|---|---|---|---|
| | Car 🟧 | Person 🟦 | Cyclist 🟥 | gIoU (%) | Car 🟦 | Person 🟥 | Cyclist ⬛ | gIoU (%) | (%) |
| MPnet [235] | 26.95 | 5.46 | 6.47 | 23.89 | 51.96 | 79.02 | 90.21 | 58.32 | 43.34 |
| MODNet [250] | 35.79 | 14.96 | 14.38 | 37.21 | 52.78 | 79.22 | 90.44 | 60.34 | 47.93 |
| CNN-MCA [237] | 47.20 | 14.21 | 18.76 | 43.38 | 53.45 | 79.09 | 90.71 | 62.83 | 50.57 |
| SMSnet++$_{\text{FLO}}$ | 28.47 | 15.15 | 10.28 | 25.54 | 51.28 | 79.76 | 89.90 | 58.69 | 45.81 |
| SMSnet$_{\text{FULL}}$ (Ours) | 49.67 | 14.17 | 23.92 | 50.24 | 54.59 | 79.44 | 90.81 | 63.54 | 52.10 |
| SMSnet$_{\text{MOV}}$ (Ours) | 56.90 | 16.38 | 25.07 | 53.25 | 64.92 | 79.35 | 91.06 | 64.78 | 55.61 |
| SMSnet++$_{\text{FULL}}$ (Ours) | 58.21 | 18.45 | 27.82 | 56.95 | 66.35 | 80.31 | 91.31 | 67.53 | 56.90 |
| SMSnet++$_{\text{MOV}}$ (Ours) | **60.32** | **20.31** | **28.36** | **59.00** | **68.65** | **81.32** | **91.56** | **68.37** | **58.42** |

(gIoU) score for the moving classes as a whole and for the static classes as a whole. The gIoU is computed by aggregating the true positives, false positives and false negatives, across all the semantic moving object classes together and the static objects classes together, followed by computing the intersection over union with these aggregated values. This gives us a measure of the overall motion segmentation performance. The ApolloScape dataset is significantly more challenging than the Cityscapes and KITTI datasets, as it contains scenes in a densely populated urban city and therefore has large open spaces with a substantial amount of moving objects.

Among the existing methods, the CNN-MCA model achieves the highest performance in the overall mIoU score and the class-wise IoU scores as well as the gIoU scores, except for the *person* class for which our SMSnet++$_{\text{FLO}}$ baseline model achieves the highest IoU score. The *person* and *cyclist* classes are the hardest to accurately segment in this dataset due to their relatively small size which is further exaggerated by the far away distances at which they appear within the image. Our proposed SMSnet model achieves an mIoU score of 55.61%, thereby outperforming the existing the approaches. The largest improvement in the moving object segmentation over the previous state-of-the-art is observed for the *car* class for which SMSnet achieves an improvement of 9.70%.

Our proposed SMSnet++ further outperforms SMSnet by achieving an mIoU score of 58.42%, as well as consistently outperforming each of the static and moving object classes, thereby achieving state-of-the-art performance. In comparison to the previous state-of-the-art CNN-MCA model, SMSnet++ achieves an improvement of 15.62% in the gIoU for the moving classes and 5.54% in the gIoU for the static classes. Similar to the SMSnet model, the largest improvement achieved by SMSnet++ over the CNN-MCA model is for the *car* class, followed by the *person* class. A notable improvement of 11.10% is also observed in the pixel accuracy. By comparing the performance of our SMSnet++

model with SMSnet++$_{\text{FLO}}$ which does not incorporate the semantics, we observe an overall improvement of 12.61% in the mIoU score and a significant improvement of 33.46% in the gIoU for the moving classes. These results corroborate the fact that adaptively encoding semantic information using our SSMA fusion module enables the motion segmentation network to effectively learn heterogeneous motion patterns of different semantic objects.

#### 6.4.2.2 Semantic Segmentation

In order to evaluate the performance of our proposed architectures on semantic segmentation, we benchmark against both the previous state-of-the-art semantic motion segmentation network CNN-MCA, as well as state-of-the-art semantic segmentation networks including DeepLab v3 [132], ParseNet [168], FCN-8s [24] and AdapNet++. Our proposed SMSnet architecture adopts the topology of our AdapNet architecture for the semantic segmentation stream and the weights of the Semantic Feature Learning stream are kept fixed while training the joint semantic motion segmentation architecture, therefore the performance of the joint model is identical to that of the AdapNet model. Whereas, in our proposed SMSnet++ architecture, we build upon our AdapNet++ model and incorporate our temporal representational warping technique using an edge-enhanced optical flow representation to simultaneously improve the semantic segmentation performance in the joint model. Therefore, we also compare against the performance of the AdapNet++ architecture in order to quantify the improvement due to temporal representational warping that we introduce. Comprehensive comparisons from this experiment are shown in Table 6.4.

Results on the Cityscapes dataset shown in Table 6.4 (a) demonstrates that our SMSnet model achieves a mIoU score of 77.56%, thereby outperforming the state-of-the-art semantic segmentation model DeepLab v3 by 2.35% and exceeding the performance of the semantic motion segmentation model CNN-MCA by 11.29%. However, the performance of SMSnet is still lower than our AdapNet++ model introduced in Chapter 4. Nevertheless, our proposed SMSnet++ model outperforms SMSnet as well as all the baselines networks by achieving the state-of-the-art performance of 82.31% for the mIoU score, while also consistently achieving a higher performance in each of the individual object class IoU scores. The largest improvement over DeepLab v3 is observed for the object classes that have thin structures such as the *fence* and *pole* classes for which SMSnet++ achieves an improvement of 21.23% and 10.87% respectively. Comparing the performance of our SMSnet++ model with AdapNet++ to analyze the impact of introducing the temporal warping, we observe an improvement of 1.51% in the overall mIoU score. This demonstrates the benefit of exploiting the complementary motion features to improve the learning of semantics with a minimal additional overhead in the computation. Note that AdapNet++ and DeepLab v3 are specialized semantic segmentation architectures. In comparison to CNN-MCA which is a semantic motion segmentation architecture, our proposed SMSnet++ achieves a much larger improvement of 16.02% in the mIoU score.

**Table 6.4:** Comparison of the semantic segmentation performance of our proposed joint semantic motion segmentation architectures.

| | Approach | Sky | Building | Road | Sidewalk | Fence | Veg | Pole | Car | Sign | Person | Cyclist | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 🟦 | ⬛ | 🟪 | 🟪 | 🟫 | 🟩 | ⬜ | 🟦 | 🟨 | 🟥 | ⬛ | (%) |
| (a) Cityscapes | FCN-8s [24] | 76.51 | 83.97 | 93.82 | 67.67 | 24.91 | 86.38 | 31.71 | 84.80 | 50.92 | 59.89 | 59.11 | 59.97 |
| | ParseNet [168] | 77.57 | 86.81 | 95.27 | 74.02 | 33.31 | 87.37 | 38.24 | 88.99 | 53.34 | 63.25 | 63.87 | 69.28 |
| | CNN-MCA [237] | 80.76 | 80.69 | 96.71 | 76.40 | 40.01 | 77.00 | 41.93 | 86.35 | 39.69 | 60.33 | 49.35 | 66.29 |
| | DeepLab v3 [132] | 92.40 | 89.02 | 96.74 | 78.55 | 41.00 | 90.81 | 49.74 | 91.02 | 64.48 | 66.52 | 66.98 | 75.21 |
| | AdapNet++ | 94.18 | 91.49 | 97.93 | 84.40 | 54.98 | 92.09 | 58.85 | 93.86 | 72.61 | 75.52 | 72.90 | 80.80 |
| | SMSnet (ours) | 92.45 | 89.98 | 97.43 | 81.43 | 49.93 | 91.44 | 53.43 | 92.23 | 65.32 | 69.86 | 69.62 | 77.56 |
| | SMSnet++ (ours) | **94.46** | **92.25** | **98.19** | **86.28** | **62.32** | **92.56** | **60.61** | **94.23** | **73.48** | **76.95** | **74.08** | **82.31** |
| (b) KITTI | FCN-8s [24] | 89.38 | 84.56 | 83.59 | 61.46 | 61.45 | 82.25 | 27.85 | 84.33 | 42.66 | 37.80 | 40.28 | 63.24 |
| | ParseNet [168] | 89.45 | 83.48 | 86.41 | 66.73 | 62.88 | 82.22 | 34.25 | 85.62 | 35.16 | 39.66 | 36.75 | 63.87 |
| | CNN-MCA [237] | 87.36 | 81.78 | 79.40 | 55.69 | 63.01 | 65.67 | 26.15 | 81.09 | 23.12 | 36.33 | 29.91 | 57.23 |
| | DeepLab v3 [132] | 88.46 | 83.69 | 84.55 | 60.71 | 61.66 | 80.69 | 34.42 | 86.06 | 47.17 | 42.99 | 36.54 | 64.27 |
| | AdapNet++ | 89.47 | 85.65 | 86.89 | 70.71 | 67.25 | 83.63 | 36.81 | 86.51 | 49.61 | 46.11 | 49.17 | 68.34 |
| | SMSnet (ours) | 88.90 | 83.77 | 86.38 | 64.16 | 63.82 | 80.96 | 34.12 | 85.75 | 50.34 | 41.87 | 40.04 | 65.47 |
| | SMSnet++ (ours) | **89.24** | **85.59** | **88.04** | **70.91** | **68.90** | **82.36** | **39.28** | **85.45** | **44.10** | **50.96** | **54.58** | **69.04** |
| (c) ApolloScape | FCN-8s [24] | 90.85 | 78.93 | 93.71 | 19.88 | 50.23 | 90.26 | 26.68 | 77.67 | 59.66 | 6.60 | 42.93 | 57.94 |
| | ParseNet [168] | 92.38 | 82.06 | 95.61 | 33.85 | 59.35 | 92.12 | 41.99 | 84.55 | 70.20 | 14.65 | 54.01 | 65.53 |
| | CNN-MCA [237] | 91.38 | 79.00 | 94.45 | 19.33 | 55.83 | 90.12 | 32.18 | 81.70 | 61.29 | 8.42 | 40.39 | 59.46 |
| | DeepLab v3 [132] | 93.17 | 81.82 | 96.32 | 40.23 | 60.61 | 92.58 | 50.36 | 86.25 | 74.75 | 16.51 | 55.81 | 68.03 |
| | AdapNet++ | 93.99 | 84.71 | 96.52 | 42.51 | 63.57 | 93.54 | 54.00 | 90.04 | 79.38 | 21.70 | 63.09 | 71.19 |
| | SMSnet (ours) | 93.15 | 81.86 | 96.02 | 40.45 | 60.97 | 92.76 | 46.21 | 85.18 | 72.76 | 11.28 | 54.14 | 66.80 |
| | SMSnet++ (ours) | **93.58** | **89.01** | **97.49** | **60.77** | **67.55** | **93.02** | **58.11** | **90.95** | **68.48** | **24.06** | **65.82** | **73.53** |

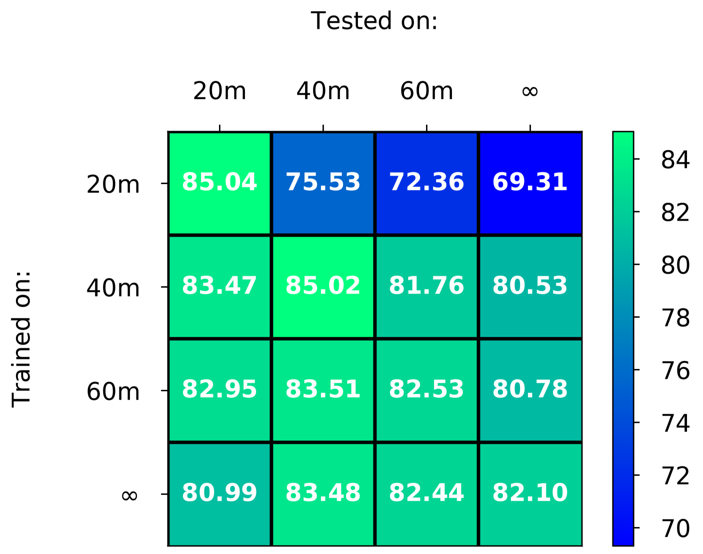
Table 6.4 (b) shows the comparison of the semantic segmentation performance on the KITTI dataset. Our proposed SMSnet model achieves a mIoU score of 65.47%, thereby outperforming each of the baseline models, excluding our AdapNet++ architecture. Compared to the state-of-the-art DeepLab v3 model, it achieves an improvement of 1.2% in the mIoU score, while we observe a substantial improvement of 8.24% in the mIoU score, in comparison to the CNN-MCA architecture. While our improved SMSnet++ architecture sets the new state-of-the-art on this dataset by achieving a mIoU score of 69.04%, which constitutes to an improvement of 4.77% over the previous state-of-the-art DeepLab v3 network and an improvement of 11.81% over the CNN-MCA model. The KITTI dataset consists of images containing *sidewalks* with outgrown grass that are labeled as *sidewalk* as opposed to *vegetation*. This causes significant misclassifications which is evident from the low IoU score of the baselines for this object class. Moreover, *person* and *cyclist* objects that appear in the KITTI dataset are often occluded by other objects such as *cars* and *vegetation* in the scene, which also causes a substantial amount of false positives. However, Our SMSnet++ architectures improves the prediction of these classes by enforcing temporal consistency between consecutive frames using our representational warping layer. Due to our temporal warping, we observe an improvement of 5.41% in the IoU score of the *cyclist* class and a similar improvement of 4.85% in the IoU score of the *person* class. While compared to the previous state-of-the-art DeepLab v3, we observe a substantially larger improvement of 18.04% and 7.97% in the IoU scores of the *cyclist* and *person* classes respectively.

We also present the results on the challenging ApolloScape dataset in Table 6.4 (c). Unlike the Cityscapes and KITTI datasets, our SMSnet model is outperformed by DeepLab v3 by 1.23% in the mIoU score, although it exceeds the performance of the CNN-MCA architecture by 7.34%. However, our proposed SMSnet++ architecture outperforms DeepLab v3, achieving the state-of-the-performance of 73.53% in the mIoU score. It demonstrates an improvement of 5.5% in the mIoU in comparison to the performance of the DeepLab v3 model and a larger improvement of 14.02% in the mIoU score compared to the CNN-MCA architecture. As we described in Section 6.4.2.1, the *person* and *cyclist* classes are the hardest to accurately predict in this dataset. This can be observed in the low IoU score of 16.51% for the *person* class and 55.81% for the *cyclist* class in the results achieved by the DeepLab v3 model. Nevertheless, our SMSnet++ model achieves an improvement of 7.55% and 10.01% in the IoU scores of *person* and *cyclist* classes respectively. Comparing the performance of the SMSnet++ model with AdapNet++, we observe an improvement of 2.34% in the mIoU, due to temporal representational warping that we introduced. The results demonstrate that on the ApolloScape dataset, the temporal warping benefits the *sidewalk* class the most as it improves the performance by 18.32% in the IoU score, followed by object classes such as *building, fence* and *pole*. This demonstrates that the temporal consistency that our representational warping layer enforces not only benefits object classes having thin pole-like structure but also large objects that occupy a significant portion of the image.

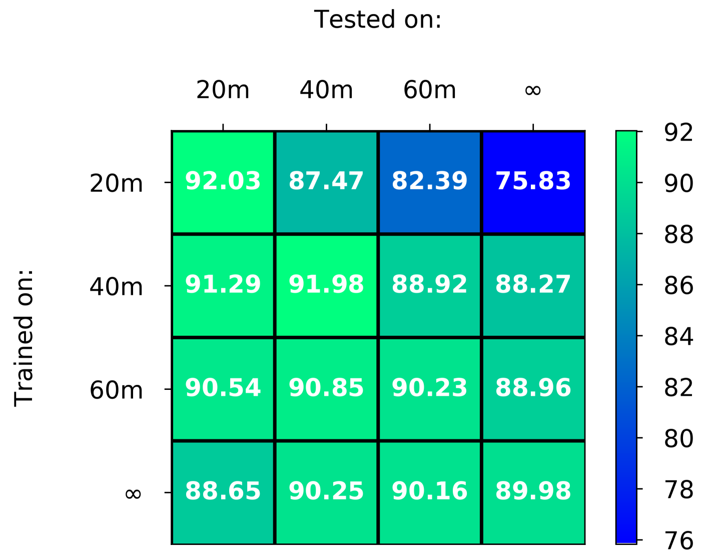
### 6.4.3 Influence of the Motion Parallax Effect

In this section, we investigate the performance of motion segmentation using our proposed SMSnet++ architecture considering different maximum ranges within which the moving objects might lie. One of the primary challenges of motion segmentation is to predict the motion of moving objects that are at far away distances from the camera. As the distance of moving objects from the camera increases, the pixel displacements of those objects in consecutive frames decreases due to the motion parallax effect. Therefore, segmenting distant moving objects from camera images is extremely challenging. On the one hand, including training examples that contain distant moving objects might enable learning of more multiscale features that can cover a wide variety of motion appearances. While, on the other hand, these highly difficult training examples can also restrict the network from training effectively, if it is unable to learn features that can distinguish the state of distant moving objects. In order to quantify this influence on the performance of our models for motion segmentation, we trained individual models on the entire training data but only considering moving objects within a certain maximum distance from the camera in the label set and objects that lie beyond this distance are ignored for training. We then evaluated these models on the different test sets, each containing moving objects within certain maximum distances. For this experiment, we considered moving objects within four discrete set of maximum distances. Specifically, for 20 m, 40 m, 60 m and over 60 m that denote as $\infty$.

Figure 6.9 shows results from this experiment on the Cityscapes-Motion and KITTI-Motion datasets. As we hypothesized, our models achieve the best performance while training on the label set containing moving objects within 20 m and testing with the corresponding label set with same configuration. On the Cityscapes-Motion dataset, our SMSnet++ model achieves a IoU of 85.02% for the moving object class, whereas on the KITTI-Motion dataset, it achieves a mIoU of 91.98% for the moving object class. The models demonstrate the best trade-off considering moving objects within a maximum distance of 40 m and subsequently for larger distances, the performance drops by approximately 3% in the IoU score. However, this score is still over 12% higher than the best performance achieved by the previous state-of-the-art networks that we presented in Table 6.1. Furthermore, we observe that there is a larger performance drop when the model is trained with labels consisting of moving objects within 20 m and tested on the label set with moving objects at increasingly larger distances. This is to be expected, as the network has not learned features that represent distant moving objects. Nevertheless, we observe that training the model on the label sets with moving objects at increasingly larger distances makes the model perform well across each of the four discrete label sets with different maximum distances. The model trained with the maximum distance at infinity



(a) Cityscapes-Motion dataset



(b) KITTI-Motion dataset

**Figure 6.9:** Comparison of moving object segmentation performance of our proposed SMSnet++ architecture, while training on label sets with moving objects within different maximum ranges on the Cityscapes-Motion and KITTI-Motion datasets. The model that was trained with the label set consisting of moving objects within 40 m offers the right tradeoff. However, the model trained with no bounds on the distance of the moving object (denoted by $\infty$) still performs impressively well and the performance is substantially higher than the previous state-of-the-art.

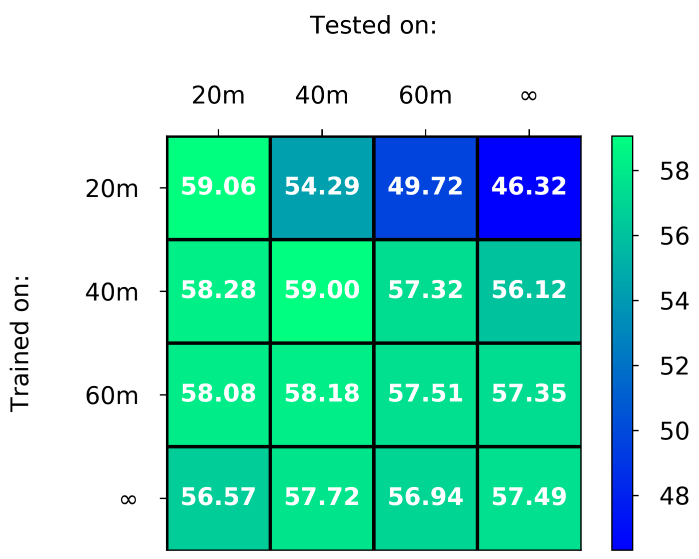


**Figure 6.10:** Comparison of moving object segmentation performance of our proposed SMSnet++ architecture, while training on label sets with moving objects within different maximum ranges on the ApolloScape-Motion dataset. Our model trained on the label set with moving objects within 40 m achieves the right trade-off. However, even our model trained on the label set with no bounds, effectively segments distant moving objects.

performs impressively well even for challenging moving object examples that are at far away distances.

Finally, we present results on the ApolloScape-Motion dataset in Figure 6.10. We obtain a similar performance as the results reported on the Cityscapes-Motion and KITTI-Motion datasets. The SMSnet++ model trained on the label set consisting of moving objects within 20 m and evaluated on the corresponding test set with the same configuration achieves the highest moving object segmentation IoU of 59.00%. However, the model trained on the 40 m label set, demonstrates the right trade-off by achieving a moving object segmentation mIoU of 57.68% over all the four discrete label sets with different maximum distances. The SMSnet++ model trained on the label set that does not contain any bounds on the moving object distances demonstrates a drop in the mIoU of 0.5% over all the four discrete label sets with different maximum distances. Nevertheless, our SMSnet++ architecture trained on the label set with no bounds on the moving object distances, achieves an improvement of 15.62% over the previous state-of-the-art CNN-MCA [237] architecture, thereby demonstrating the efficacy of our model in effectively segmenting distant moving objects.

### 6.4.4 Ablation Study

In this section, we present detailed ablation studies on the various architectural design choices that we made in our proposed SMSnet and SMSnet++ architectures with concrete empirical evaluations on each of the related model configurations. We first present results on the major components of our SMSnet++ architecture in Section 6.4.4.1, followed by an analysis on the introduction of ego-flow suppression in Section 6.4.4.2 and the utility of temporal representational warping in Section 6.4.4.3. We then present evaluations of different topologies of the Motion Feature Learning stream in Section 6.4.4.4 and finally, in Section 6.4.4.5, we present an analysis on the performance of motion segmentation with the label set that contains all the semantic objects in the dataset and the label set that contains only the movable semantic object classes.

#### 6.4.4.1 Detailed Study on the Architectural Components

Table 6.5 shows various configurations of the major components of our proposed architectures using models trained on the Cityscapes-Motion dataset. For this experiment, we consider all the labels in the Cityscapes-Motion dataset for learning the moving object segmentation, regardless of whether the object class is movable, therefore this corresponds to the models with the suffix *FULL* in the benchmarking results that we presented. The model M1 has a topology similar to our proposed SMSnet architecture, but we remove the Semantic Feature Learning stream, therefore the model only learns motion features from the learned optical flow maps. We employ the residual units from the ResNet-50 architecture in the various network blocks and we embed the FlowNet2 architecture in the Motion Feature Learning stream to generate the optical flow maps from which our network learns the motion features. The resulting optical flow maps are scaled to the 0 to 255 range and the flow vectors are represented as integers, which is indicated by (i) in the *Flow Arch* column of Table 6.5. This M1 model demonstrates an IoU of 67.42% and 92.90%, for the moving and static classes respectively. In the subsequent M2 model, we build upon the M1 model and add the Semantic Feature Learning stream, which has the topology of our AdapNet architecture that we introduced in Chapter 4. Feature maps from the Semantic Feature Learning stream are then fused with the learned flow field features in the Semantic-Motion Fusion stream using simple concatenation. This model exactly resembles the topology of our proposed SMSnet architecture and hence, learns joint semantic motion segmentation. It achieves a semantic segmentation performance of 77.58% in the mIoU score and demonstrates an improvement of 9.14% and 0.52%, for the moving and static classes respectively. This improvement can be attributed to the benefit of encoding semantic representations in the motion segmentation stream.

In the M3 model, we replace the standard residual units that are in the M2 model, with full pre-activation residual units and we also replace the AdapNet architecture in the Semantic Feature Learning stream, with our improved AdapNet++ architecture. This model achieves an improvement of 3.78% for the moving class and 1.01% for the static class, in addition to an improvement of 3.22% in the mIoU score for semantic segmentation. This increase in the scores is primarily due to the multiscale feature learning and long-range

**Table 6.5:** Influence of the various architectural contributions proposed in the SMSnet and SMSnet++ architectures. The performance is shown for the models trained on the Cityscapes-Motion dataset. *Base Arch* denotes that the various residual units follow either the standard standard ResNet-50 or the full pre-activation ResNet-50 architecture. *Flow Arch* denotes the architecture embedded for learning optical flow field features and the symbols (i) or (f) denote if the flow displacements are integers or floating point values. The symbols $\oplus F_w$ denote warping of intermediate encoder representations from the previous timestep that are dynamically fused using our SSMA module, with representations of the current timestep. The *Fusion* column denotes if the motion and semantic feature are concatenated ($\|$) or adaptively fused using our SSMA module. The subsequent columns denote if class balancing is applied, if our proposed new decoder is employed and if the skip refinement is preformed in the motion segmentation stream. The final model M10 is our proposed SMSnet++ architecture trained on the full label set.

| Model | Base Arch | Flow Arch | Semantic Arch | Fusion | Class Bal | New Dec | Skip | Moving IoU (%) | Static IoU (%) | Semantic mIoU (%) |
|---|---|---|---|---|---|---|---|---|---|---|
| M1 | ResNet-50 | FlowNet2 (i) | - | - | - | - | - | 67.42 | 92.90 | - |
| M2 | ResNet-50 | FlowNet2 (i) | AdapNet | $\|$ | - | - | - | 76.56 | 92.38 | 77.58 |
| M3 | PA ResNet-50 | FlowNet2 (i) | AdapNet++ | $\|$ | - | - | - | 80.34 | 93.39 | 80.80 |
| M4 | PA ResNet-50 | FlowNet3 (i) | AdapNet++ | $\|$ | - | - | - | 80.87 | 93.42 | 80.80 |
| M5 | PA ResNet-50 | FlowNet3 (f) | AdapNet++ | $\|$ | - | - | - | 81.05 | 93.51 | 80.80 |
| M6 | PA ResNet-50 | FlowNet3 (f) | Adapnet++ $\oplus F_w$ | $\|$ | - | - | - | 81.08 | 93.49 | 82.31 |
| M7 | PA ResNet-50 | FlowNet3 (f) | AdapNet++ $\oplus F_w$ | SSMA | - | - | - | 80.55 | 94.78 | 82.31 |
| M8 | PA ResNet-50 | FlowNet3 (f) | AdapNet++ $\oplus F_w$ | SSMA | ✓ | - | - | 81.95 | 94.08 | 82.31 |
| M9 | PA ResNet-50 | FlowNet3 (f) | AdapNet++ $\oplus F_w$ | SSMA | ✓ | ✓ | - | 82.49 | 94.05 | 82.31 |
| M10 | PA ResNet-50 | FlowNet3 (f) | AdapNet++ $\oplus F_w$ | SSMA | ✓ | ✓ | ✓ | **83.11** | **94.38** | **82.31** |

context aggregation in the semantic segmentation stream. In the subsequent M4 model, we replace the FlowNet2 architecture in the beginning of the Motion Feature Learning stream, with the FlowNet3 architecture with the aim of improving the learning of optical flow field features. This model achieves a IoU of 80.87% for the moving class and 93.42% for the static class. However, as the improvement in performance is not significant, we further removed the rounding-off to the nearest decimal that was being applied on the optical flow maps and maintained the optical flow vectors are floating point numbers in the M5 architecture. We indicate the floating point flow maps with the suffix (f) in the *Flow Arch* column of Table 6.5. This model in comparison to the M3 model, achieves an improvement of 0.71% in the IoU for the moving class and 0.12% in the IoU of the static class.

In an effort to simultaneously improve the semantic segmentation performance, we employ our temporal representational warping layer that also transforms the optical flow to the edge enhanced flow for improving the warping in the Semantic Feature Learning stream of the M6 model. This improves the semantic segmentation performance by 1.51% in the mIoU score. Furthermore, a comparison of warping directly using the optical flow maps, instead of using the edge-enhanced flow is presented in Section 6.4.4.3. In the models that we presented thus far, we concatenate the learned semantic features with the motion features in the Semantic-Motion Fusion stream. This direct concatenation may not be ideal as the motion boundaries learned from the optical flow maps do not exactly correspond to the object boundaries in the semantic feature maps. Moreover, the motion features do not contain much information about distant moving objects. Therefore, in order to exploit complementary information from both semantic and motion features, we employ our SSMA fusion module that we introduced in Chapter 5 to fuse the semantic representations with the motion features in the M7 model. However, this model demonstrates a reduction in the IoU score of the moving object class with a concurrent increase in the IoU score of the static object class. Upon closer inspection of the performance of this model, we observed that static and moving object classes are highly imbalanced. The images often contain a substantial amount of pixels that belong to static objects while only a few number of pixels representing the moving objects. In order to account for this imbalance, in the M8 model, we employ the normalized inverse class frequency weighting to balance the loss. Most networks employ the inverse class frequency weighting, however, as the moving object class is not present in each training example, it leads to large weight updates and causes the gradients to explode. In order to alleviate this problem, we re-normalize the frequency weightings by a factor to make the sum of the weights for each training example as one. This model achieves an improved performance of 81.95% and 94.05%, for IoU of the moving and static classes respectively.

Finally, in order to improve the granularity of the motion segmentation along object boundaries, we employ our new multistage decoder that we introduced in Chapter 5 in the Semantic-Motion Feature Fusion stream of the M9 model and we subsequently add two skip refinement stages from the encoder of the Semantic Feature Learning stream to the decoder of the Semantic-Motion Feature Fusion stream in the M10 model. This leads to an IoU improvement of 0.54% in the M9 model and an additional 0.62% in the M10 model for the moving object class. We denote this final model M10, which achieves a IoU of 83.11% for the moving class, 94.38% for the static class and a mIoU of 82.31% for the semantic segmentation as our proposed SMSnet++ architecture.

#### 6.4.4.2 Influence of Ego-Flow Suppression

As the main goal of this work is to enable a robot to detect moving objects in dynamic environments, the movement of the robot itself causes discontinuities in the optical flow magnitude, where nearby static objects can appear to have a larger optical flow magnitudes even though they are static in the environment. In order to alleviate this problem, we compute the ego-motion of the robot and subtract the resulting ego-flow from the generated optical flow maps. In this section, we compare the performance of the SMSnet++ model with and without Ego-Flow Suppression (EFS) on the Cityscapes-Motion dataset with only the movable object labels in the motion segmentation network. Results from this experiment shown in Table 6.6 demonstrate that the E1 model that directly uses the optical flow maps in floating point precision from the embedded FlowNet3 network to learn motion features, achieves a mIoU of 79.44% for the motion segmentation. In the subsequent E2 model, we scale the optical flow maps to the 0-255 range and maintain the floating point precision. This model achieves an improvement of 3.68% and 2.24% in the IoU score of the moving and static classes. Although our Motion Feature Learning stream contains convolution layers with batch normalization that normalizes the output of the previous activation layer, we find that explicitly normalizing the optical flow maps to the 0-255 range yields a notable performance improvement. In our final E3 model, we employ our EFS technique to compensate for the flow induced due to the ego-motion of the robot. The E3 model achieves a mIoU score of 89.92%, which accounts for a large improvement in the IoU score of 11.53% for the moving class and 3.5% for the static class, in comparison to the E2 model. The improvement is more apparent when compared with the E1 model which directly uses the optical flow maps from the embedded flow generation network. More specifically, this amounts to an improvement of 15.23% in the moving object class and 5.74% for the static object class.

Furthermore, we show a comparison of the motion segmentation performance using our SMSnet++ model trained on each of the aforementioned optical flow transformations on all the datasets that we benchmark on in Figure 6.11. The results demonstrate that our E3 model, which scales the optical flow maps to the 0-255 range in floating point precision and employs our EFS technique, consistently achieves the highest performance on the Cityscapes-Motion, KITTI-Motion and ApolloScape-Motion datasets. The largest improvement of 15.23% in the IoU score for the moving object class is observed on the

**Table 6.6:** Evaluation of the motion segmentation performance with different optical flow transformations in the SMSnet++ architecture. Results are shown on the Cityscapes-Motion dataset.

| Model | Scaling | EFS | IoU (%) | | mIoU | Acc. | AP |
|---|---|---|---|---|---|---|---|
| | (0-255) | | Moving | Static | (%) | (%) | (%) |
| E1 | - | - | 69.79 | 89.08 | 79.44 | 91.28 | 88.68 |
| E2 | ✓ | - | 73.47 | 91.32 | 82.40 | 93.14 | 93.47 |
| E3 | ✓ | ✓ | **85.02** | **94.82** | **89.92** | **95.99** | **94.85** |

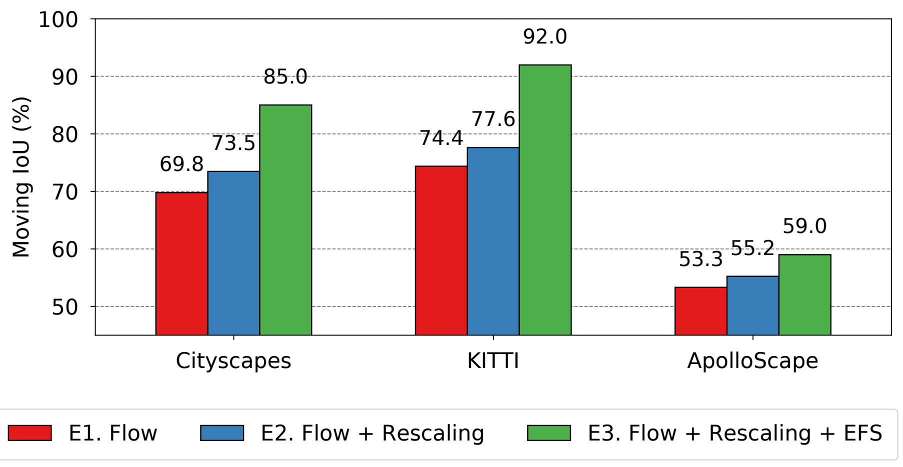


**Figure 6.11:** Performance comparison of motion segmentation with different optical flow transformations in our SMSnet++ architecture.

Cityscapes-Motion dataset and the smallest improvement of 5.68% in the IoU score for the moving object class is observed on the ApolloScape-Motion dataset. The comparatively smaller improvement on the ApolloScape-Motion dataset is due to the substandard optical flow maps that were obtained on this dataset, attributable to the significantly different scene structure compared to the other datasets. Since this dataset does not provide groundtruth optical flow maps for training the flow network, we employed the FlowNet3 model trained on the KITTI dataset and kept the weights fixed, while training the rest of the Motion Feature Learning stream. Employing an unsupervised optical flow learning network would yield a larger improvement in the final E3 model. Nevertheless, these results demonstrate the utility of employing our EFS technique to learn more effective motion representations in our proposed architectures.

#### 6.4.4.3 Influence of Warping Semantic Features

In this section, we evaluate the improvement due to temporal representational warping by introducing our warping module at different stages of the segmentation stream in our

**Table 6.7:** Evaluation of warping semantic features in the SMSnet++ architecture at various stages of the network using the edge-enhanced optical flow. The M3 model is our final SMSnet++ architecture. Results are shown on the Cityscapes dataset.

| Model | Warping Stage | Edge Enhance | mIoU (%) | Acc. (%) | AP (%) |
|---|---|---|---|---|---|
| M1 | - | - | 80.80 | 96.04 | 90.37 |
| M2 | eASPP | - | 81.96 | 96.26 | 91.22 |
| M3 | eASPP | ✓ | **82.31** | **96.37** | **90.67** |
| M4 | eASPP + Res3 | ✓ | 81.36 | 96.11 | 90.54 |
| M5 | eASPP + Res4 | ✓ | 81.89 | 96.22 | 90.56 |
| M6 | eASPP + Res3 + Res4 | ✓ | 81.19 | 96.11 | 90.59 |

SMSnet++ architecture. Table 6.7 shows the results from this experiment on the Cityscapes dataset. In the M2 model, we directly use the optical flow maps from the embedded FlowNet3 network to warp the feature maps at the end of the semantic encoder after the eASPP, where the feature maps are of dimensions $24 \times 48$. This model achieves a mIoU of 81.96%, accounting for an improvement of 1.16% compared to the M1 model that has the topology of our AdapNet++ architecture. In order to further improve the semantic segmentation performance using representational warping, we employ the edge-enhanced flow for warping in the M3 model, as opposed to the optical flow directly. Figure 6.12 shows examples of the edge-enhanced flow in comparison to the original optical flow from our network. We perform the warping at the same stage as the M2 model, after the eASPP. This model achieves a mIoU 82.31% which accounts for an improvement of 1.51% over the standard AdapNet++ segmentation model M1.

Subsequently, we experiment with adding another temporal warping layer to the M3 model at the end of the previous downsampling stages in an effort to enforce multiscale temporal consistency. Specifically, we add the second warping layer at the end of the *Res3* block in the M4 model and at the end of the *Res4* block in the M5 model. However, the results demonstrate that both these models do not improve the performance over the M3 model with the temporal warping layer after the eASPP module. Furthermore, we also experimented with introducing the warping layer at more than two downsampling stages. In the M6 model, we introduce a warping layer at the end of the eASPP module and at the end of the last residual unit of the *Res3* as well as the *Res4* blocks. This model achieves a lower performance than the M4 and M5 models with two temporal warping layers. Therefore, we employ the M3 model configuration in the Semantic Feature Learning stream of our proposed SMSnet++ architecture to introduce temporal consistency in the semantic segmentation output with respect to the previous frame.

Analyzing the results presented in Table 6.7, we observe that the performance of our model increases considerably using the edge-enhanced flow for warping at the end of

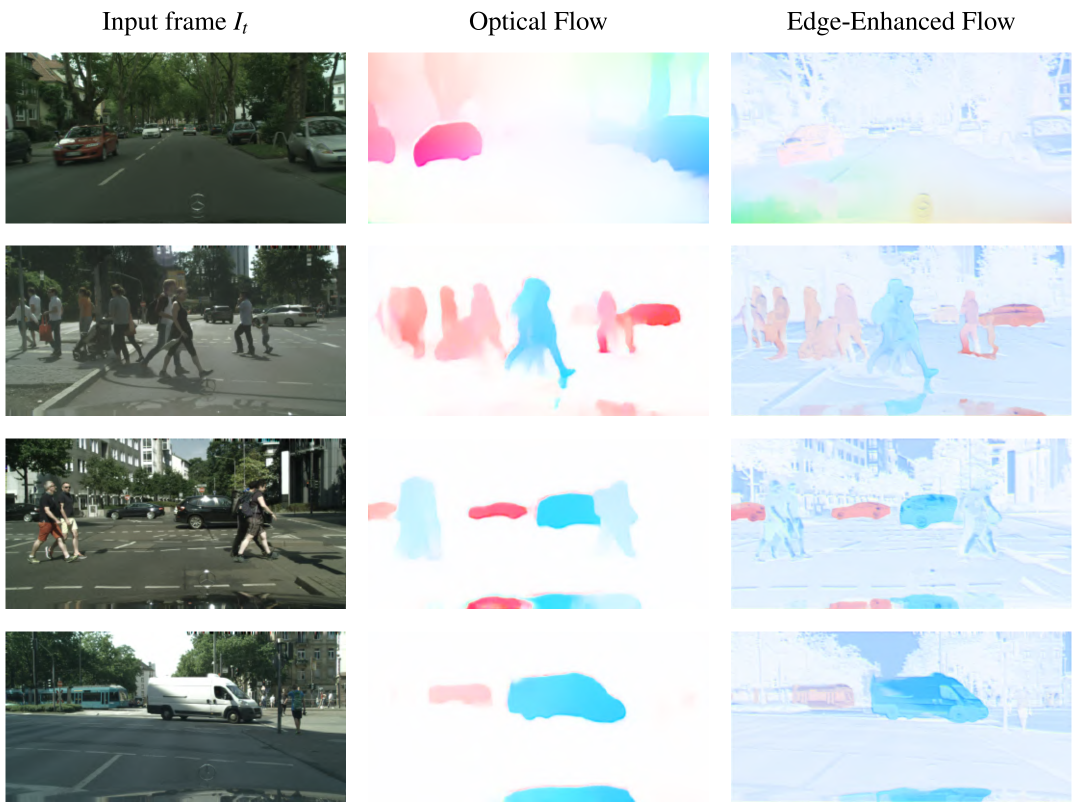


**Figure 6.12:** Examples from the Cityscapes dataset showing the input image $I_t$, the original optical flow and the corresponding edge-enhanced flow from our SMSnet++ architecture. The edge-enhanced flow clearly shows more emphasis on the structure of objects in the scene.

the eASPP but thereafter, the increase in performance is lower than that obtained from introducing only one temporal warping layer. This can be attributed to the max pooling layer that we employ to downsample the optical flow map to the desired warping resolution. Pooling the optical flow maps at multiple scales causes the downsampled flow maps to not align with the corresponding semantic feature maps and therefore, when the semantic feature maps from the previous timestep are warped with the unaligned optical flow maps, it negatively affects the performance of the model. Hence, computing the optical flow maps at multiple scales or employing a scaling technique that maintains the point correspondences could potentially alleviate this problem.

### 6.4.4.4 Evaluation of Motion Feature Learning Configuration

In order to learn optical flow field features in the Motion Feature Learning stream of our proposed architectures, we experiment with adding different number of residual blocks after the embedded optical flow network. Table 6.8 presents the results from this experiment on the Cityscapes-Motion dataset with the full label set. The optical flow

**Table 6.8:** Evaluation of various network configurations for learning optical flow field features in the SMSnet++ architecture. The F4 model is our final SMSnet++ architecture trained on the full label set in the Cityscapes-Motion dataset.

| Model | Configuration | IoU (%) | | mIoU | Acc. | AP |
|---|---|---|---|---|---|---|
| | | Moving | Static | (%) | (%) | (%) |
| F1 | Res1 | 81.93 | 93.56 | 87.74 | 95.12 | 94.89 |
| F2 | Res1 + Res2 | 82.47 | 94.23 | 88.35 | 95.45 | 95.12 |
| F3 | Res1 + Res2 + Res3 | 82.79 | 93.51 | 88.15 | 95.38 | 95.01 |
| F4 | Res1 + Res2 + Res3(2) | **83.11** | **94.38** | **88.75** | **95.59** | **95.37** |
| F5 | Res1 + Res2 + Res3(2) + Res4 | 82.34 | 93.67 | 88.00 | 95.33 | 95.07 |

network in the beginning of the Motion Feature Learning stream yields an output with the same dimensions as the input image resolution. Therefore, in the F1 model, we employ the *Res1* block from the full pre-activation ResNet-50 architecture, followed by a $1 \times 1$ convolutional layer with striding to downsample the features maps to match the resolution of the semantic encoder output. This model achieves a mIoU of 87.74% for the motion object segmentation.

In the subsequent two models, we build upon the F1 model and add an additional *Res2* block the in F2 model and an additional *Res2* as well as a *Res3* block in the F3 model. While the F2 model improves the IoU of the moving object class by 0.54% and 0.67% for the moving object class, the F3 model further improves the moving object IoU by 0.32% but yields a lower performance for the static class. Therefore, in the M4 model, we build upon the topology of the M3 model and reduce the number of residual units in the *Res3* block by employing only the first two units (indicated by (2) next to Res3 in the *Configuration* column of Table 6.8) to prevent the model from overfitting to the training data. This model further improves the performance of both the moving and static classes by 1.18% and 0.82% respectively, compared to the first F1 model. We also experimented with adding the fourth residual block from the full pre-activation ResNet-50 architecture to the F4 model, however, this lowered the overall mIoU score by 0.65%. Therefore, we employ the F4 model configuration to effectively learn flow field features in both our proposed SMSnet and SMSnet++ architectures.

#### 6.4.4.5 Evaluation on Full Labels vs. Only Movable Objects

In the motion segmentation benchmarking results that we reported in Section 6.4.2.2, we presented the performance of two variants of our models, one that is trained on the label set containing all the semantic objects in the dataset, despite all of the objects being not movable (SMSnet++$_{\text{ALL}}$) and one that is trained on the label set consisting of only movable objects (SMSnet++$_{\text{MOV}}$). In this section, we compare the performance of

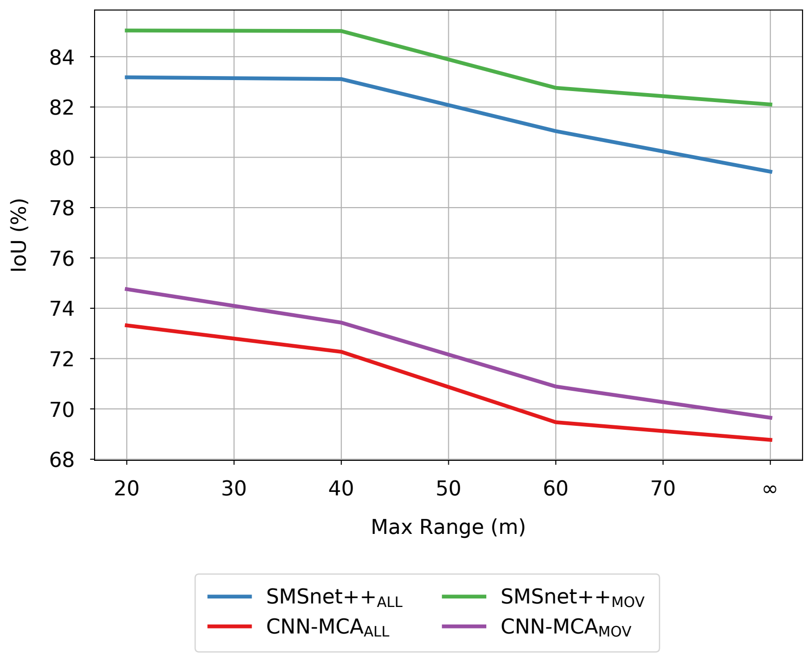


**Figure 6.13:** Comparison of motion segmentation of our SMSnet++ model with the previous state-of-the-art CNN-MCA model, each trained on a label set consisting of only movable objects and the label set that contains all the objects in the dataset, irregardless of their movability. We report the results of these model variants for a set of discrete ranges within which the moving object lies in the test set.

these two model variants with the corresponding variants of the previous state-of-the-art CNN-MCA architecture on the Cityscapes-Motion dataset with moving objects at different maximum ranges. Figure 6.13 shows the results from this experiment for four discrete set of maximum ranges within which the moving object lies. Specifically, for 20 m, 40 m, 60 m and over 60 m that we denote as $\infty$.

We observe that both the SMSnet++$_{\text{ALL}}$ and SMSnet++$_{\text{MOV}}$ model variants, significantly outperform the corresponding variants CNN-MCA$_{\text{ALL}}$ and CNN-MCA$_{\text{MOV}}$ models for each of the discrete maximum ranges. The highest performance is achieved by our SMSnet++$_{\text{MOV}}$ model trained only the movable objects that are within 20 m from the ego-vehicle equipped with the camera. It can also be seen that the performance of each of these models slowly decreases as the distance from the moving object increases. This can be attributed to the motion parallax effect, where the pixel displacements of moving objects in consecutive images become smaller as the distance from the camera increases and thereby, detecting the motion of these objects becomes increasingly hard.

More importantly, we see that our SMSnet++$_{\text{MOV}}$ model trained only on the movable object labels outperforms the SMSnet++$_{\text{ALL}}$ model that is trained on the entire label set consisting of all the semantic object classes in the dataset for each of the discrete maximum ranges. Although the difference in performance of these two models is not significant, this demonstrates that the motion segmentation network learns the heterogeneous motion patterns of various dynamic objects more effectively, if the network is provided with the prior of defining what types of semantic categories are potentially movable in the label set while training. In contrast, when the network is trained on all the semantic object labels in the dataset that contain both permanently static objects and movable objects, it is additionally required to learn which of the semantic objects are movable which causes the performance to decrease by 1.8% in the mIoU over all the discrete maximum ranges. Nevertheless, our proposed SMSnet++ architecture demonstrates a smaller decrease in performance, while compared to the previous state-of-the-art CNN-MCA architecture.

### 6.4.5 Qualitative Evaluations

In this section, we present qualitative results of semantic motion segmentation using our proposed SMSnet and SMSnet++ architectures in comparison with the previous state-of-the-art CNN-MCA [237] architecture on diverse scenes from the Cityscapes-Motion, KITTI-Motion and ApolloScape-Motion datasets. Figure 6.14 shows the results on the Cityscapes-Motion dataset, where the first two rows shows the comparison of SMSnet++ with the CNN-MCA model as the baseline, while the last two rows show the comparison of SMSnet++ with our SMSnet model as the baseline. Furthermore, we also show the improvement / error map where the improvement in moving object segmentation of SMSnet++ in comparison to the baseline is indicated with cyan pixels, while the improvement in semantic segmentation of SMSnet++ in comparison to the baseline model is indicated with green pixels and finally, the misclassifications in the SMSnet++ output in comparison to the groundtruth is indicated with red pixels. The color legend for the segmentation labels correspond to those shown in the motion and semantic benchmarking results in Section 6.4.2.1.

In Figure 6.14 (a), we show a scene with heavy traffic. We observe one moving *car* in the same direction as our ego-vehicle and four moving *cars* in the opposite direction. The CNN-MCA model mispredicts the parked truck in front of the ego-vehicle as well as the parked *car* in front of the parked truck as moving and does not detect the distant moving *car* in the front of the opposite lane. Additionally, it also mispredicts the *person* standing behind the truck as a *car* and does not detect the bumper of the *car* that is just entering the frame on the right side of the image. Whereas, our proposed SMSnet++ model accurately predicts the motion status as well the semantic category of these objects. The multiscale residual units in our SMSnet++ model enables accurate detection of entire moving objects of different scales. Moreover, it can be observed that the granularity of the motion segmentation along the moving object boundaries is much finer than the CNN-MCA model. The overall improvement in the semantic segmentation can also be seen in the more refined segmentation of the two *people* on the sidewalk, accurate estimation of the object boundaries of the *sidewalk* class as well as refined segmentation of *pole* and

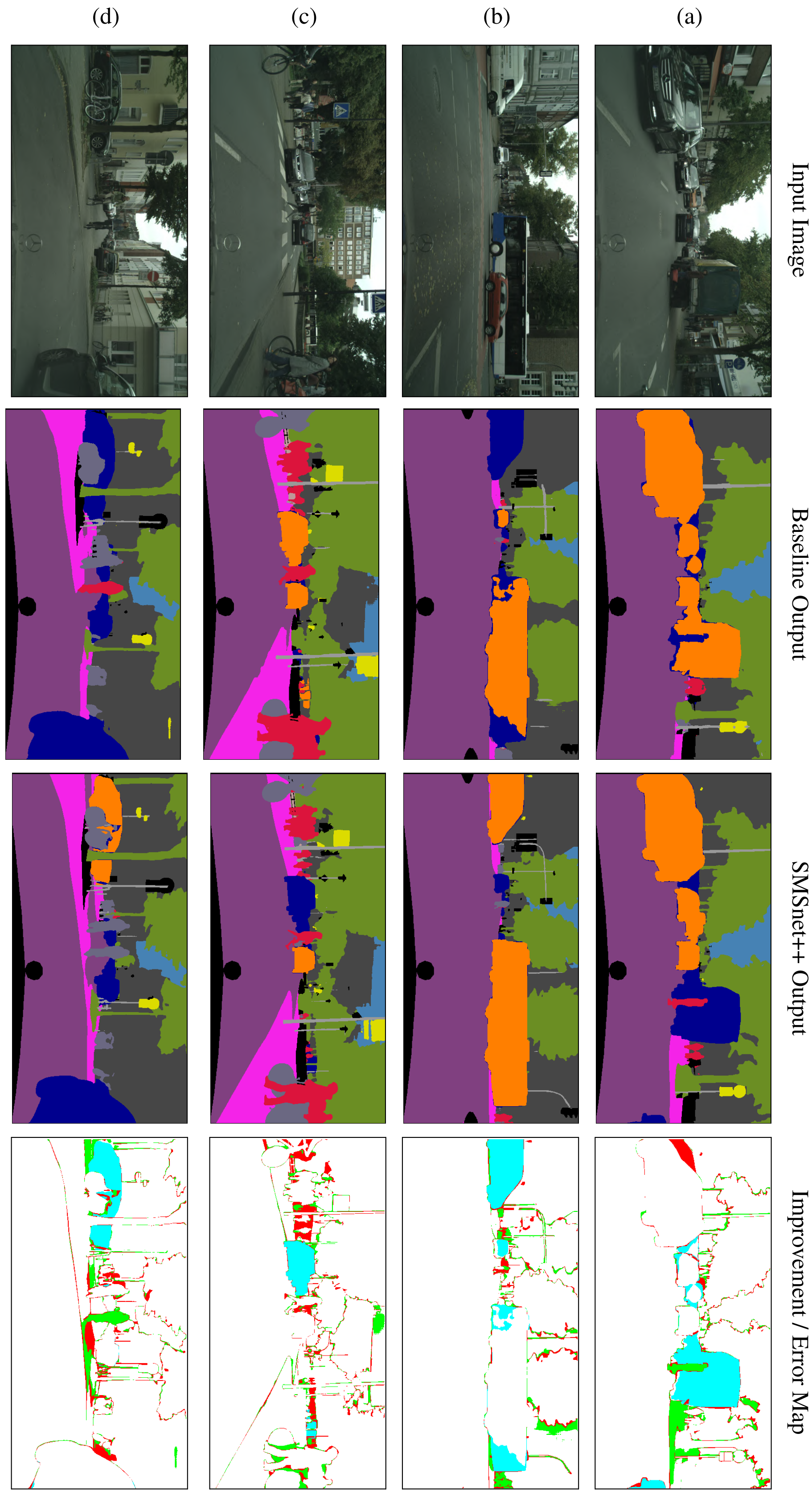


**Figure 6.14:** Qualitative results of our SMSnet++ model in comparison to (a, b) CNN-MCA [237] and (c, d) SMSnet on the Cityscapes-Motion dataset. In addition to the segmentation output, we also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the baseline model but correctly predicted by SMSnet++ for semantic segmentation in green and for motion segmentation in cyan. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 6.4.2.

*sign* objects.

In the second example shown in Figure 6.14 (b), we observe that the ego-vehicle is waiting at the traffic light, while a *bus*, *car* and a *van* are moving in the perpendicular direction to the ego-vehicle. It can be seen that the *bus* and the *car* are moving in the opposite directions and are adjacent to each other. This causes the baseline CNN-MCA model to not entirely detect all the pixels that belong to the *bus* as moving due to its small receptive field that does not encompass the entire moving object. Moreover, in the CNN-MCA output, the *van* entering the frame on the left side of the image is also not detected as moving and the static *car* waiting at the traffic light on the opposite side of the ego-vehicle is incorrectly detected as moving. Whereas, we observe that our proposed SMSnet++ model accurately predicts the semantic object class and motion status of all the pixels in the these cases. In the first two examples, the improvement seen in the motion segmentation is due to the large effective receptive field of our SMSnet++ architecture that enables the detection of entire large moving objects that occupy a substantial portion of the image. The improvement in semantic segmentation output of our SMSnet++ model in comparison to CNN-MCA can be observed in the accurate detection of *person* and *cyclist* in the scene, as well as in the boundaries of the *vegetation* class.

We show a comparison of the output of our SMSnet++ architecture with SMSnet as the baseline in Figure 6.14 (c). We observe that the four static *cars* waiting at the pedestrian crossing on the opposite side of the ego-vehicle are incorrectly segmented as moving in the output of the SMSnet model and additionally, the distant *bus* behind the trees on the right side of the image is also incorrectly segmented as moving in the output of SMSnet model. Whereas, our SMSnet++ model accurately predicts the motion status of these objects as static. We observe an improvement in the semantic segmentation output of our SMSnet++ model in comparison to SMSnet, for objects such as *pole, sign* and *pedestrians*, as well as a part of the *building* which is overexposed due to the sun. This improvement can be attributed to the representational warping layer that leverages semantic features from the previous frames to enforce temporal consistency. In the results shown in Figure 6.14 (d), we see an interesting scenario where a moving *car* is partly occluded by a tree that splits the *car* in two halves. This causes the SMSnet model to predict the pixels of the *car* as static. whereas our SMSnet++ model accurately predicts the entire *car* as moving due to its large effective receptive field which encompasses the parts of the car on either side of the tree. Furthermore, a *cyclist* is incorrectly predicted as a *person* in the segmentation output of the SMSnet model. An improvement in the semantic segmentation output of SMSnet++ can be observed in the accurate prediction of the semantic category of objects such as *sidewalk, road, cyclist, pole* and *sign* classes. Most of these improvements can be attributed to the strong decoder with skip refinement stages that accurately captures the boundaries of these objects.

Figure 6.15 shows qualitative results on the KITTI-Motion dataset which contains both scenes from residential areas where *cars* are moving with low velocities, and in highway

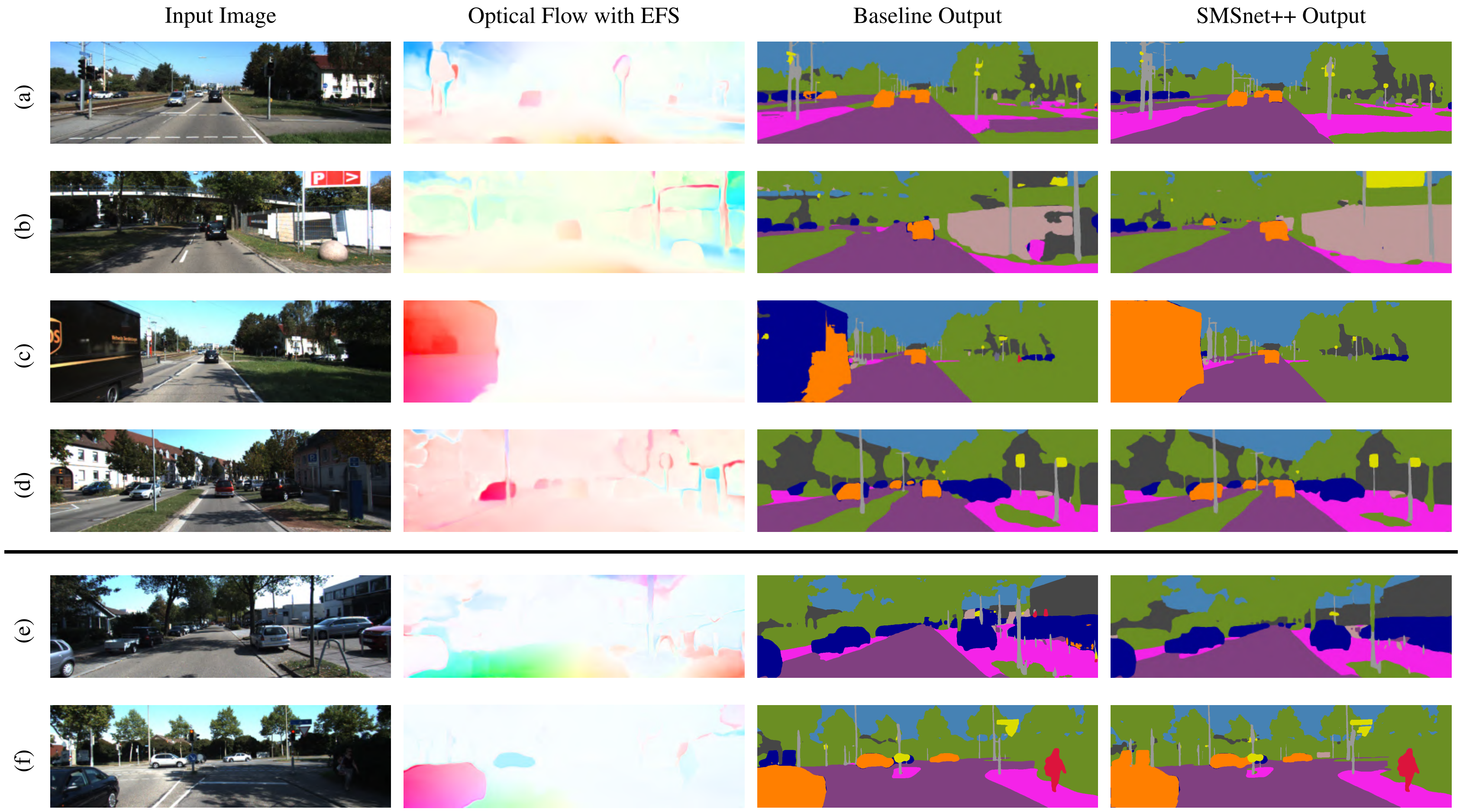


**Figure 6.15:** Qualitative semantic motion segmentation results of our SMSnet++ model in comparison to (a, b) CNN-MCA [237] and (c, d) SMSnet on the KITTI-Motion dataset. The last two rows show the failure modes in comparison to (e) SMSnet and (f) CNN-MCA models as the baseline. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 6.4.2.

driving scenarios where *cars* are moving at high velocities, therefore causing a substantial amount of motion blur in the captured images. These scenes also have moving objects of different scales and lighting conditions that cause significant shadows on the ground. The first two rows show the comparison of SMSnet++ with the previous state-of-the-art CNN-MCA model as the baseline, while the next two rows show the comparison of SMSnet++ with the SMSnet model as the baseline model. In addition to the semantic motion segmentation outputs, we also show the optical flow maps with EFS for each of the examples. In Figure 6.15 (a), we observe that the CNN-MCA model incorrectly segments the static *cars* on the opposite lane from the ego-vehicle as moving and also segments the shadow of the first moving *car* in the opposite direction as belonging to the moving *car*. Whereas, our SMSnet++ architecture accurately predicts the semantic category and motion status of each of these objects. Moreover, the improvement in the semantic segmentation output of SMSnet++ can be observed in the accurate segmentation of *sidewalk, pole* and *sign* classes. In the second example shown in Figure 6.15 (b), we see that the CNN-MCA model incorrectly predicts the moving *car* on the opposite side of the lane as static and a distant moving *car* turning on the same side of the lane as the ego-vehicle as static, while our SMSnet++ model accurately predicts these *cars* as moving. It can also be seen that the CNN-MCA model misclassifies a large portion of objects such as *sign, fence* and *vegetation*. The accurate prediction of object classes such as the *sidewalk* and the *fence* in the above examples can be attributed to the representational warping and the improvement in segmenting thin pole-like structures is enabled by our new decoder that fuses mid-level encoder features to yield a high-resolution segmentation output.

In the example shown in Figure 6.15 (c), we observe that the SMSnet output shown as the baseline does not segment the entire *van* as moving and it also incorrectly segments a parked *car* on the opposite side of the road adjacent to the *van* as moving. This primarily occurs in the segmentation output of SMSnet, when there is a moving vehicle that is partly visible due to it entering or leaving the frame. Note that both our SMSnet and SMSnet++ models employ the same optical flow map with EFS to learn the motion features and it can be observed that even though there are high flow gradients in the optical flow maps, SMSnet does not entirely segment the moving *van*. In the subsequent example shown in Figure 6.15 (d), we observe a residential scene, in which a moving *car* on the opposite side of the lane is only partly segmented as moving in the output of the SMSnet model due to an occluding light pole dividing the moving *car* into two parts. Moreover, it can be seen that as the distance to the moving objects increases, the region of segmentation on the moving *cars* decreases. In the output of the SMSnet model, we observe that most part of the second moving *car* is accurately segmented as moving, but only a small part of the third moving *car* is segmented as moving and the fourth moving *car* is entirely segmented incorrectly as static. However, we observe that the multiscale receptive fields of the SMSnet++ architecture enable it to efficiently distinguish between static background and moving objects of different scales, thus achieving accurate motion segmentation.

The last two rows of Figure 6.15 shows failure cases in comparison to SMSnet in (e) and in comparison to the CNN-MCA model in (f). In Figure 6.15 (e), we observe a distant moving *car* on the road among static *cars* being parked on both sides of the street. Both the SMSnet output shown as the baseline and the SMSnet++ output incorrectly segment the moving *car* as static. As the scene is not well lit near the distant moving *car*, the network is unable to detect the change in the pixel displacements. While in the second failure case shown in Figure 6.15 (f), we observe a scene in which a moving *car* is merely visible due to the occlusion caused by another moving *car*, in addition to a traffic light *pole* and a *sign* covering most of the vehicle. This causes both the networks to detect the pixels of the moving *car* as static. However, it can be seen that a moving *van* on the left side of the image is accurately segmented as moving in the output of the SMSnet++ model, even though a large portion of the object is occluded by a tree trunk, while the CNN-MCA model incorrectly predicts the *van* as being static. Nevertheless, we remark that these corner cases can be overcome by aggregating temporal features of moving objects from a sequence of frames, while our network currently only learns the motion patterns from a pair of subsequent images.

Finally, Figure 6.16 shows the qualitative semantic motion segmentation results on the recently introduced ApolloScape-Motion dataset. As this dataset consists of images in high resolution, they contain objects such as *cars* and *pedestrians* that are captured at extremely far away distances, which makes motion segmentation highly challenging. Moreover, it contains multiple types of moving objects such as *cars, person* and *cyclist* The first two rows show the comparison of our proposed SMSnet++ architecture with the previous state-of-the-art CNN-MCA baseline and the last two rows show a comparison of partial failure modes of our SMSnet++ model with SMSnet as the baseline. For each of the examples, we show the input image, the corresponding optical flow maps with EFS, the segmentation outputs and the improvement / error map.

In the first example shown in Figure 6.16 (a), we see a scene with multiple distant moving *cars* in several directions. Out of the six *cars* that are moving in the same direction as the ego-vehicle, the CNN-MCA model segments two of them as moving and does not capture the last two *cars* in the segmentation output. In addition, the CNN-MCA model misclassifies the moving *motorcycle* as static and it does not capture the *truck* that is moving adjacently in the opposite direction of the ego-vehicle. Whereas, our SMSnet++ model that incorporates our multiscale residual units and the eASPP, accurately predicts the semantic category and motion status of pixels belonging to the aforementioned objects of different scales. It can also be observed that the CNN-MCA model falsely predicts *vegetation* on the bottom left of the road due to motion blur in the image and it does not entirely detect the two static *cars* on the left lane. Comparing the quality of the segmentation, we can see that our SMSnet++ model has a more refined segmentation output due to the multistage refinement strategy that it incorporates, while the segmentation of the CNN-MCA model is coarse and has several discontinuities or missing structures of pole-like objects. In

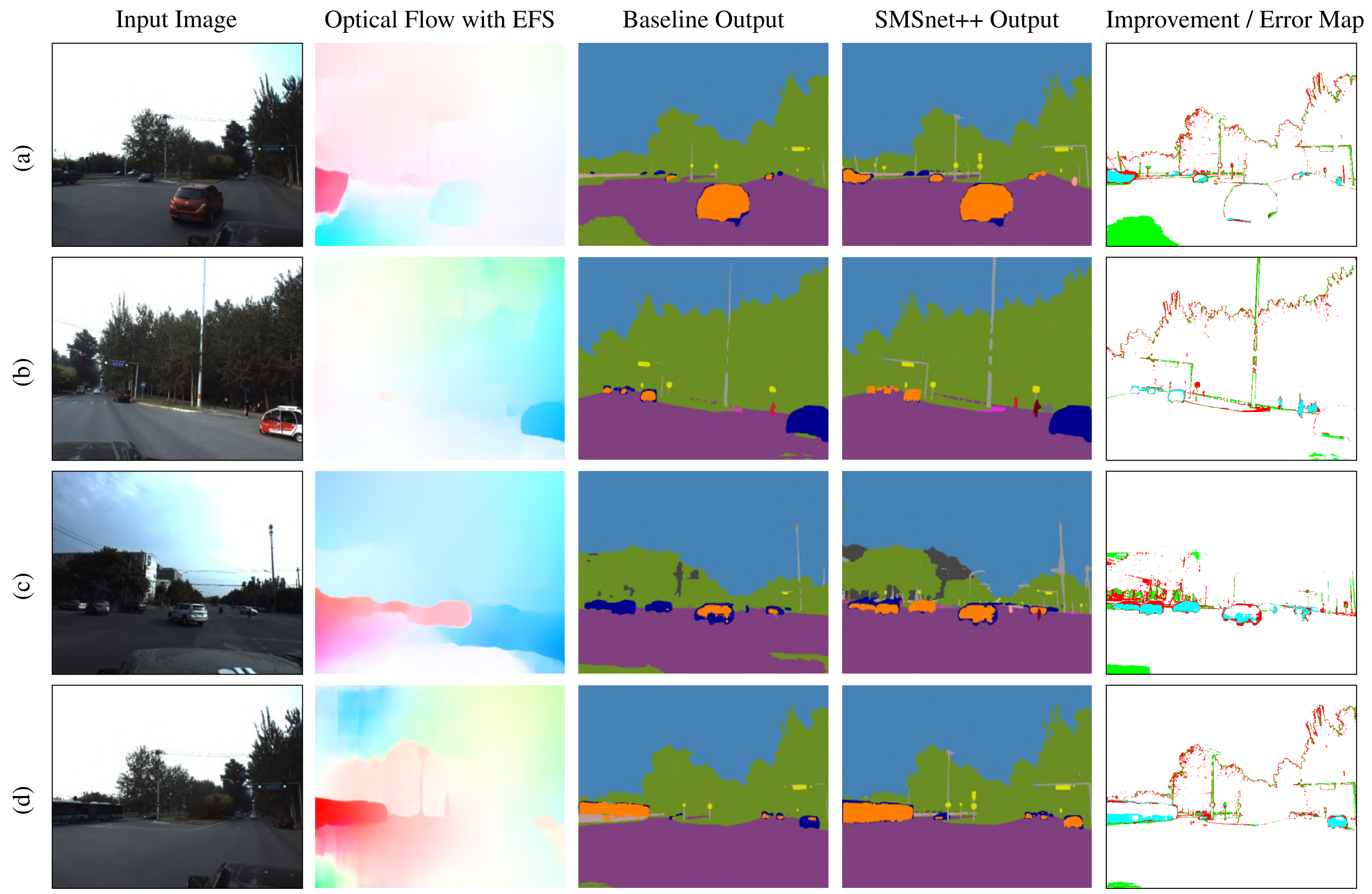


**Figure 6.16:** Qualitative semantic motion segmentation results of our SMSnet++ model in comparison to (a, b) CNN-MCA [237] and (c, d) SMSnet on the ApolloScape-Motion dataset. We also show the improvement / error map which indicates the misclassified pixels in red and the pixels that are misclassified by the baseline model but correctly predicted by SMSnet++ for semantic segmentation in green and for motion segmentation in cyan. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 6.4.2.

the second example shown in Figure 6.16 (b), there are eight moving *cars* on the same lane and a moving *person* on the right side of the image. The CNN-MCA model only segments three of the eight *cars* as moving and misclassifies the moving *person* as static. Additionally, it does not detect a static *person* on the right side of the image and again misclassifies parts of the road on the bottom right of the image as *vegetation* due to motion blur. Analyzing the segmentation output of our SMSnet++ model, we see that it accurately predicts the motion status and the semantic object category of the pixels in this image.

In the next example shown in Figure 6.16 (c), we observe five moving *cars* on the same direction as the ego-vehicle, one moving *car* on the opposite direction of the ego-vehicle, one moving *person* and one moving *motorcycle*. The output of the SMSnet model shown as the baseline only partially segments two of the moving *cars* out of the total six present in the scene, and does not detect both the moving *person* and the *motorcycle*. Moreover, the numerous parked static *cars* along the road in front of the ego-vehicle are all left undetected in the segmentation output and most parts of the *buildings* are misclassified as *vegetation*. In addition static *pedestrians* and *poles* that are at far away distances are not captured in the segmentation output of the SMSnet model. Once again, we also observe parts of the *road* on the bottom corners of the image are segmented incorrectly, as *vegetation* due to motion blur in the images. Whereas, our proposed SMSnet++ model does not demonstrate the misclassifications due to motion blur as it effectively leverages semantic information from the previous frames using our representational warping to yield temporally smoother predictions. Although SMSnet++ accurately segments the scene, it can be seen that certain pixels of the moving *car* are incorrectly classified as static. This is primarily due to the optical flow maps that do not capture the accurate boundaries of the moving objects on this dataset, in addition to artifacts such as trails of ghost regions that are observed behind moving objects. As there are no groundtruth optical flow labels to train the flow generation network in a supervised fashion on this dataset, we employ the network trained on the KITTI flow dataset and keep the weights fixed while training the rest of the motion segmentation stream. Therefore, employing an unsupervised optical flow network will alleviate the aforementioned problem.

In the last example shown in Figure 6.16 (d), we observe seven moving *cars* on the same lane as the ego-vehicle and a moving *bus* as well as a moving *car* on the opposite lane. It can be seen that the SMSnet model only partially segments four out of the eight moving *cars* in the same lane, while also partially segmenting the moving *bus* and the *car* on the opposite lane. However, our SMSnet++ model segments all the *cars* but misclassifies three out of the seven distantly moving *cars* on the same lane as static. While it accurately segments the *bus* on the opposite lane as moving, it only partially classifies the pixels on the *car* behind the *bus* as moving. Observing the optical flow map, we see that the distant objects are not captured in the scene which causes the motion segmentation network to misclassify distant moving cars as static. Comparing the semantic segmentation output of the SMSnet++ model with SMSnet, we observe a significant improvement in the granularity of the segmentation for object classes such as *vegetation, pole* and *sign*. Nevertheless, we remark that this dataset is extremely challenging and the results that we presented in this chapter is the first method that has been benchmarked thus far.

### 6.4.6 Generalization Evaluations

In this section, we demonstrate the generalization ability and the platform independence of our proposed SMSnet++ architecture in comparison to the previous state-of-the-art CNN-MCA [237] and our SMSnet architectures. We trained each of these networks on the Cityscapes-Motion dataset and evaluated them on the images that we collected in Freiburg using a ZED stereo camera. Note that the Cityscapes dataset was collected with an automotive grade large baseline stereo camera, while we employed a consumer grade stereo camera for collecting the images that we evaluate on in this section. The images that we collected consists of challenging perceptual conditions such as low-lighting, glare, shadows and motion blur, which make semantic motion segmentation more challenging. Qualitative semantic motion segmentation results from this experiment are presented in Figure 6.17, where the first two rows show a comparison of the segmentation output from SMSnet++ with CNN-MCA as the baseline and the subsequent two rows show the comparison of the output from our SMSnet++ architecture with the SMSnet architecture as the baseline.

Figure 6.17 (a) shows a scene with a moving *car* in the adjacent lane and static *cars* that are parked along the sidewalk. Although the CNN-MCA model accurately detects the moving *car*, it also segments parts of the static *cars* that are parked along the road as moving. Additionally, due to glare from the sun, it misclassifies pixels on *buildings* as *sign* and segments the entire *sky* as a *building*. It also incorrectly classifies sections of the *road* near the vanishing point as *sidewalk* and it does not detect the *sidewalk* that the *cars* are parked on in the left side of the image. Whereas, our SMSnet++ model accurately segments the scene and classifies the moving objects precisely, even in the presence of glare from the sun. The second example shown in Figure 6.17 (b) shows a similar scene in which there is one moving *car* on the same lane as the ego-vehicle and there are two moving *cars* approaching from the opposite direction, with parked *cars* along the sidewalk on either sides of the road. Analyzing the output of the CNN-MCA model, we observe that it reasonably segments the *car* ahead of the ego-vehicle as moving, however, it misclassifies the parked *car* along the right side of the road as moving. Interestingly, it can be observed that the model classifies the two moving *cars* that are approaching from the opposite direction as static and simultaneously classifies the entire lane that the *cars* are traversing on as *sidewalk*. Our hypothesis is that as a consequence of classifying the two moving *cars* as static, it incorrectly classifies the lane that they are traversing on as *sidewalk*. Conversely, as a consequence of classifying the the adjacent lane as a *sidewalk*, it incorrectly classifies the moving *cars* as static. However, the segmentation output of

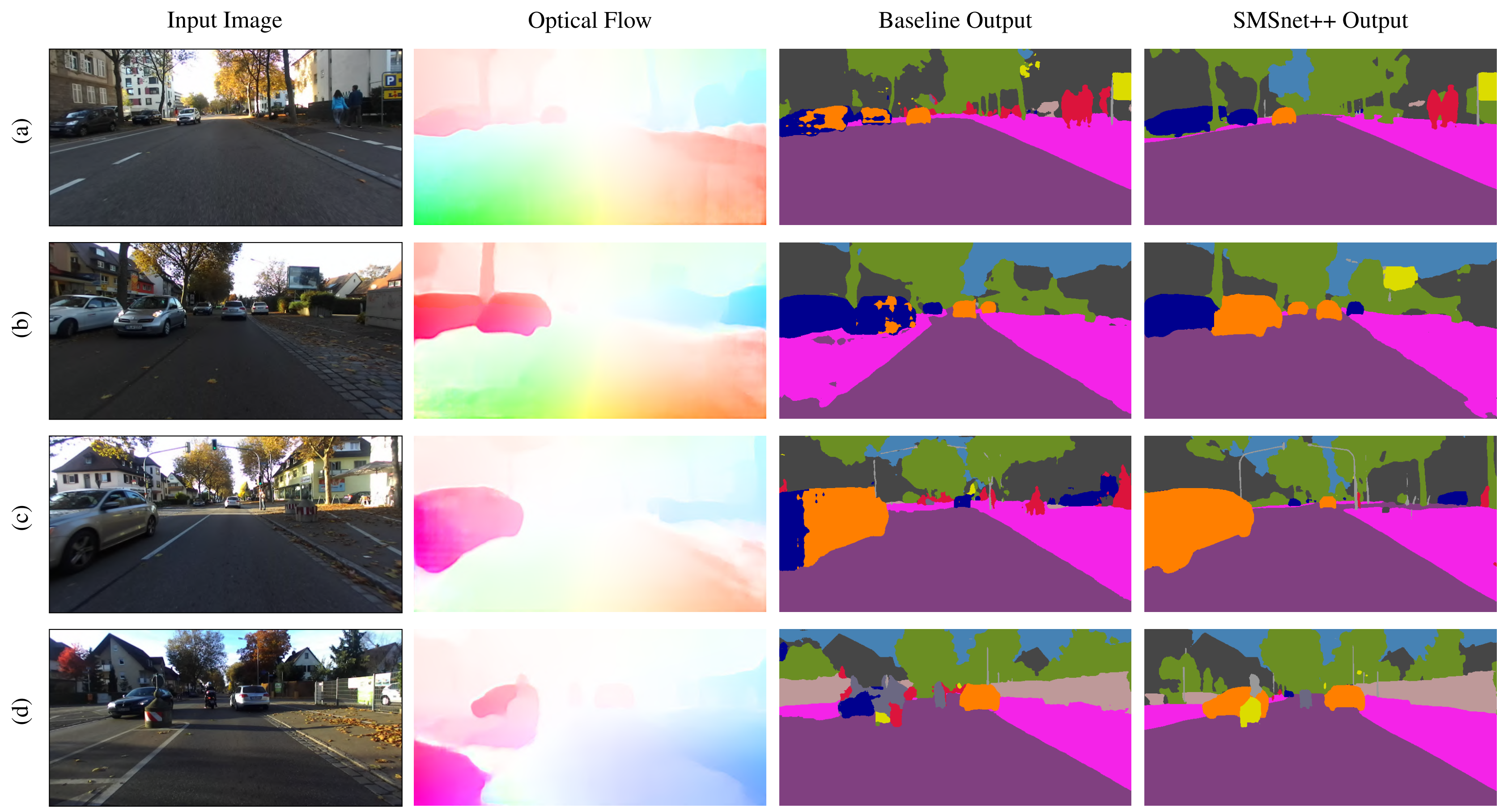


**Figure 6.17:** Qualitative semantic motion segmentation results of various architectures trained on the Cityscapes-Motion dataset and evaluated on images collected in Freiburg. The results of our SMSnet++ model are shown in comparison to (a, b) CNN-MCA [237] and (c, d) SMSnet. The color legend for the segmentation labels correspond to those shown in the benchmarking tables in Section 6.4.2.

our SMSnet++ model does not show these incorrect classifications, rather it accurately distinguishes between the static and moving objects in the scene.

In Figure 6.17 (c), we observe a scene with a moving *car* ahead of the ego-vehicle on the same lane and another moving *car* approaching on the adjacent lane. The output of the SMSnet model shown as the baseline, segments most of the *car* on the adjacent lane as moving, however, it misclassifies pixels along the edge of the image where the *car* is exiting the frame as static. Additionally, it incorrectly classifies the *car* ahead of the ego-vehicle as static. Whereas, our SMSnet++ model accurately assigns the pixels to the static and moving object categories. Comparing the semantic segmentation output of both models, we observe that due to the glare from the sun, the SMSnet model misclassifies most of the pixels near the vanishing point of the *road* and also misclassifies objects on the section of the *sidewalk* on the right side of the image that is brightly lit by sunlight. However, our SMSnet++ model leverages semantic information from the previous view using our temporal warping layer to more accurately segment these areas that are over exposed due to the sunlight. The improvement in the semantic segmentation output of SMSnet++ can be observed in precise segmentation of the object boundaries as well as in the segmentation of thin pole-like structures.

The final example shown in Figure 6.17 (d) depicts a scene where there is a moving *car* as well as a *motorbike* ahead of the ego-vehicle and another moving *car* is approaching on the opposite side in the adjacent lane. The output of the SMSnet model demonstrates accurate segmentation of the *car* and most pixels belonging to the *motorbike* ahead of the ego-vehicle as moving. However, as the *car* approaching on the opposite side is partly occluded due to a bollard on the traffic island, the section of the *car* on one side of the bollard is incorrectly classified as static, while the other section is incorrectly classified as *motorbike*. Observing the output of our SMSnet++ model, we see that both the *cars* and the *motorbike* are accurately predicted as moving and more interestingly, both sections of *car* on the either side of the bollard as predicted as moving. This can be attributed to the large effective receptive field of the SMSnet++ architecture that effectively enables the network to incorporate more context information into the final prediction. The improvement in the semantic segmentation of the SMSnet++ model is more evident in the area on the left side of the image, where the distinction between *fence* and *sidewalk* is accurately predicted. In addition, we can see that our SMSnet++ model accurately segments the bollard as a *sign* and a *pole*. However, the SMSnet model misclassifies the bollard as a *person* and a *motorbike*. Overall, we observe that our SMSnet++ model trained on the Cityscapes-Motion dataset performs substantially better in newer environments than the previous state-of-the-art CNN-MCA model and the SMSnet architecture. Our SMSnet++ model accurately segments moving objects according to their semantic classes and demonstrates negligible false positives as well as exceptional generalization of the learned kernels.

## 6.5 Related Work

In this chapter, we addressed the problem of joint semantic motion segmentation using convolutional neural network architectures. The networks that we introduced enables a robot to simultaneously predict both the semantic object category and motion status of each pixel in an image, more accurately than employing individual specialized models for each task. Moreover, as robots require information from both tasks simultaneously in order to plan future actions, a single coherent model is more efficient for deployment. Semantic segmentation and motion segmentation are two fundamental problems in scene understanding that each have substantial amount of prior work in their areas. However, there are only a handful of techniques that have focused on tackling them in a joint framework with the aim of exploiting semantic representations to improve motion segmentation. In Section 4.5 of Chapter 4, we presented a thorough review of the related work in semantic scene segmentation. In this section, we first describe the most relevant work in motion segmentation as well as the closely related task of video segmentation, followed by methods that address the problem of semantic motion segmentation jointly.

There are numerous approaches that have been proposed for segmenting moving objects from stationary camera images [252, 253, 254]. Vidal *et al.* [252] propose a classical approach that minimizes the reprojection error subject to all multibody epipolar constraints for segmenting dynamic scenes containing multiple rigidly moving objects. Sapagnolo *et al.* [253] introduce an approach for foreground moving object segmentation that combines background subtraction with temporal image analysis. While, Gao *et al.* [254] propose a method that combines the Kirsch operator with optical flow for moving object detection. However, these approaches fail in degenerate cases and they cannot be directly applied to moving camera images as the movement causes a dual motion appearance which consists of the background motion and the object motion.

In general, methods that detect motion from freely moving cameras partition the image into coherent regions with homogenous motion. This process splits the image into background and moving clusters. These methods can be categorized into optical flow based and tracking based approaches. Optical flow based techniques [255, 256] check if the motion speed as well as direction of a region is consistent with its radially surrounding pattern and then classifies it as a moving object if the motion of the region deviates from this pattern. Namdev *et al.* [257] propose an approach that combines optical flow and geometric cues to generate dense segmentation using a graph-based clustering algorithm. The approach further demonstrates how both these cues complement each other to segment moving objects that have difficult degenerate motions. In a similar work, Lenz *et al.* [258] introduce a class independent method for object detection that uses stereo images as well as optical flow information and demonstrate superior performance in the presence of unknown objects in traffic scenarios where appearance-based object detectors often fail. Wedel *et al.* [259] propose an energy minimization approach to detect and segment independently moving objects using scene flow and stereo images. Experiments that were performed in challenging scenarios show that their approach accurately localizes independently moving objects where traditional background subtraction techniques fail. Kao *et al.* [260] derive a geometric model that relates 2D motion to a 3D motion field relative to the camera, based on estimated depth and motion of vanishing points in the scene. Spectral clustering is then applied on the recovered 3D motion field to obtain the moving object segmentation. Although qualitative evaluations have been shown on the KITTI benchmark, no quantitative comparisons were reported. The major disadvantage of these methods is that they are prone to occlusion, noise in the optical flow map and edge effects due to optical flow misalignments with true object boundaries.

Tracking based techniques [234, 251, 261, 262] on the other hand, aim to detect and localize target objects in successive frames. Tracking of objects yields movement trajectories and by estimating the ego-motion of the camera, objects can be segmented from the background motion. Romero-Cano *et al.* [263] propose a technique that estimates the likelihood of pixel motion from the fusion of dense optical flow with depth information and temporal consistency is incorporated by tracking the moving objects across consecutive images. Lin *et al.* [264] propose a motion segmentation framework that combines 3D geometric constrains with high-level spatio-temporal features learned from consecutive stereo images. Tourani *et al.* [265] propose an approach in which first motion models are generated and merged using trajectory clustering into different motion affine subspaces. Moving object proposals generated from the prior model then yield a sparse collection of points on the dynamic object. Drayer *et al.* [262] introduce a weakly supervised motion segmentation technique that first extracts temporally consistent object tubes based on an off-the-shelf detector, followed by building a spatio-temporal graph by connecting the detections to segment objects. Ochs *et al.* [230] cluster long term point trajectories for temporally consistent moving object segmentation. A disadvantage of these approaches is that they typically have long processing pipelines resulting in high computation times and coarse segmentations.

Several approaches have extensively explored the use of convolutional neural networks for motion segmentation [235, 236]. Fragkiadaki *et al.* [236] propose an approach that first generates region proposals using multiple segmentations on optical flow and static boundaries, following which a moving objectness detector rejects proposals on static backgrounds and then ranks spatio-temporal segments by mapping clustered trajectories to pixel tubes. Tokmakov *et al.* [235] introduce an encoder-decoder architecture that takes the optical flow as input to first learn a coarse representation of the motion features which are then iteratively upsampled and refined to yield the full resolution motion segmentation. Perazzi *et al.* [266] formulates the motion segmentation as a guided instance segmentation problem, while combining offline and online training on static images to perform video object segmentation. Similarly, Caelles *et al.* [267] propose a video object segmentation network that is first pre-trained on generic datasets for the task of foreground object segmentation and then fine-tuned on the first frame of the test sequence while processing each frame independently. Cheng *et al.* [268] propose a unified architecture consisting of an optical flow learning stream and a semantic segmentation stream. Features from both these streams are propagated bidirectionally at the high-level to improve both segmentation and optical flow generation. Jain *et al.* [269] propose a similar two-stream architecture that fuses motion and appearance features for video object segmentation. Subsequently, Siam *et al.* [250] propose another similar architecture that performs object detection in addition to motion segmentation in a joint model. Most of these approaches are oblivious to object categories and generate coarse object boundary segmentations. Moreover, they have been solely benchmarked on video segmentation datasets that only have one moving object in each frame. Whereas, our proposed networks are capable of segmenting multiple moving and static objects while accounting for their semantic object category.

Recent methods have also explored estimating semantic object labels and motion labels jointly [232, 233, 237, 270]. Reddy *et al.* [232] propose an approach that generates motion likelihoods based on depth and optical flow estimations, while combining them with semantic and geometric constraints within a dense conditional random field. However, the approach has limited generalization ability as it primarily relies on handcrafted features. Chen *et al.* [270] introduce a method that detects object-level motion from a moving camera using two consecutive image frames and provides 2D bounding boxes as the output. They design a robust context-aware motion descriptor that considers moving speed, as well as the direction of objects and combines them with an object classifier. The descriptor measures the inconsistency between local optical flow histograms of objects and their surroundings, giving a measure of the state of motion. More recently, a multi-step framework was proposed [233], in which sparse image features from two consecutive stereo image pairs are first extracted and matched, followed by classifying the matched feature points using RANSAC into inliers caused by the camera and outliers caused by moving objects. Subsequently, the outliers are then clustered in a U-disparity map which provides the motion information of objects and a dense CRF is then used to merge the motion information with the semantic segmentation provided by a FCN. Another closely related work was proposed by Haque *et al.* [237] in which a three stage pipeline is employed. Their approach integrates optical flow as a constraint with semantic features into a dilated convolutional neural network and achieves state-of-the-art performance for joint semantic motion segmentation on the KITTI dataset.

One of the major drawbacks of these approaches is the substantially large interference time, which ranges from a few seconds to even minutes, thereby making them unusable for robotic applications that require near real-time performance such as autonomous driving. More importantly, these approaches only focus on jointly learning both semantic segmentation and motion segmentation in single framework but they do not simultaneously exploit the semantic cues to improve motion segmentation and vice versa. In contrast to these techniques, in this chapter, we presented the novel SMSnet architecture that combines learned multiscale semantic representations with optical flow field features for joint semantic motion segmentation. Our model compensates for the flow induced by the ego-motion of the camera and is several orders faster than existing techniques. Additionally, our improved SMSnet++ architecture also improves the semantic segmentation performance using a representational warping technique and further improves the motion segmentation by incorporating adaptive fusion as well as a multistage refinement strategy for high-resolution motion segmentation.

## 6.6 Conclusions

In this chapter, we addressed the problem of semantic motion segmentation using convolutional neural network architectures that take two consecutive images as input and learns to predict both the semantic object class label and motion status of each pixel in an image. Our proposed SMSnet architecture first learns coarse representations of optical flow field features while compensating for the flow induced due to the ego-motion of the camera and simultaneously learns multiscale semantic representations in a parallel stream. Subsequently, the network combines the learned semantic features with the flow field features and further learns discriminative deep representations while refining the predictions to yield the pixel-wise semantic motion labels. The fusion of semantic features with learned optical flow field features boosts the performance of segmenting moving objects, especially in cluttered and challenging scenes with multiple moving objects of different semantic categories in the scene. Furthermore, we presented the SMSnet++ architecture that achieves deeper synergy between semantics, motion and appearance by incorporating a representational warping technique to improve the temporal consistency of semantic segmentation, while adaptively fusing semantic information with flow field features to improve motion segmentation. Rather than using the learned optical flow directly to warp semantic feature maps, the temporal warping layer transforms the ego-flow suppressed optical flow to an edge-enhanced representation, which further improves the semantic segmentation performance. Concurrently, the adaptive fusion of semantic and optical flow field features discards over-segmentation or parts of semantic objects that are static in the resulting motion segmentation. The proposed SMSnet++ architecture additionally integrates our multistage refinement strategy in both the semantic stream and the motion stream to yield a high resolution segmentation output that accurately captures the object boundaries. Our proposed semantic motion segmentation frameworks are modular and can be easily be adapted with other semantic segmentation or optical flow generation architectures.

We extended three standard semantic segmentation benchmark datasets with motion annotations and made them publicly available to encourage future research in this domain. These large datasets are the first-of-its-kind and enable training of deep convolutional neural networks for joint semantic motion segmentation. We validated the effectiveness of our joint model through extensive experiments on both semantic segmentation as well as motion segmentation tasks and showed that it outperforms individual specialized models. We presented exhaustive quantitative results on the Cityscapes-Motion, KITTI-Motion and ApolloScape-Motion datasets that demonstrate that both our architectures set the new state-of-the-art on these benchmarks and run in an online fashion. We presented comprehensive ablation studies that detail our architectural design choices and also presented extensive qualitative results in autonomous driving scenarios. Furthermore, we presented qualitative evaluations on real-world driving data from Freiburg that contain challenging perceptual conditions and demonstrated that our models generalize effectively to new environments.

# Chapter 7

# Geometrically Consistent Semantic Visual Localization

**Semantic understanding and localization are fundamental key enablers of robot autonomy that have for the most part been tackled as disjoint problems. While deep learning has enabled recent breakthroughs across a wide spectrum of scene understanding tasks, its applicability to state estimation tasks has been limited due to the direct formulation that renders it incapable of encoding scene-specific constrains. In this chapter, we propose two multitask convolutional neural network architectures coupled with a novel Geometric Consistency loss function that utilizes auxiliary learning to leverage relative pose information during training, thereby constraining the search space to obtain consistent pose estimates. We introduce the VLocNet architecture that incorporates hard parameter sharing to enable inter-task learning while regressing the 6-DoF global pose and the relative odometry estimate from consecutive monocular images. We also propose the novel VLocNet++ architecture that exploits complex interdependencies between learning semantics, regressing 6-DoF global pose and odometry, for the mutual benefit of each of these tasks. Our network aggregates motion-specific temporal information and fuses semantic features into the localization stream based on region activations. Extensive experiments on the Microsoft 7-Scenes and DeepLoc datasets demonstrate that our networks set the new state-of-the-art.**

## 7.1 Introduction

Thus far in this thesis, we have addressed several critical scene understanding challenges in the context of robot perception. The techniques that we have introduced enable a

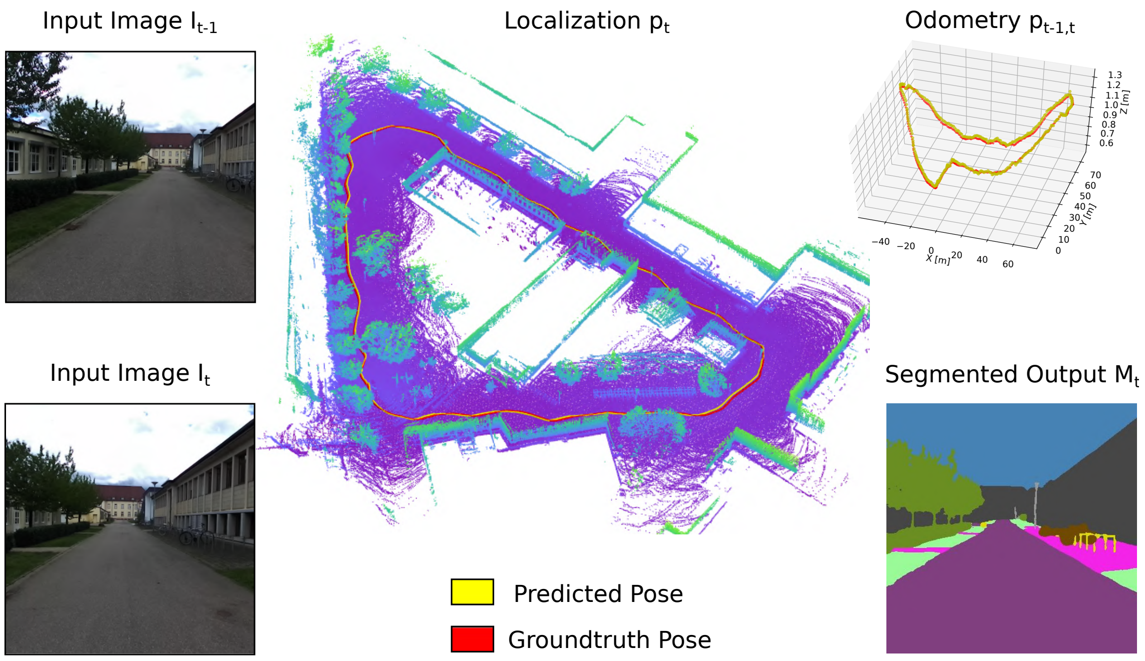


**Figure 7.1:** Output of our multitask VLocNet++ model that simultaneously estimates the 6-DoF global pose $p_t$, semantics of the scene $M_t$ and the 6-DoF odometry estimate $p_{t-1,t}$ from consecutive monocular images $(I_{t-1}, I_t)$ as input. Results are shown for the testing seq-2 from the DeepLoc dataset. VLocNet++ yields accurate pose estimates by leveraging semantic and geometric knowledge from the environment during training and inference. Additionally, the multitask network improves learning of semantics by aggregating context from the previous timestep.

robot that is tasked with navigating in dynamic outdoor environments or complex indoor environments to reliably understand the various elements of its surroundings even in adverse perceptual conditions. In this chapter, we advocate that by utilizing this information about the scene, a robot can accurately estimate its position in the environment. Moreover, visual localization can benefit from this understanding as both these tasks share complex interdependencies that can be exploited for their mutual benefit. Therefore, our goal in this chapter is to enable an autonomous robot to accurately localize itself in the environment, understand its surroundings and precisely estimate its ego-motion in a single coherent framework, while exploiting the synergies between these diverse yet vital tasks that are precursors to action execution or planning.

Visual localization is a fundamental transdisciplinary problem and a crucial enabler for numerous robotics as well as computer vision applications, including autonomous navigation, Simultaneous Localization and Mapping (SLAM), Structure-from-Motion (SfM) and Augmented Reality (AR). More importantly, it plays a vital role when robots lose track of their location, or what is commonly known as the kidnapped robot problem. In order for robots to be safely deployed in the wild, their localization system should be robust to frequent changes in the environment; whether environmental changes such as illumination and seasonal appearance, dynamic changes such as moving vehicles and pedestrians, or structural changes such as renovated buildings. Visual localization techniques can be broadly classified into two categories: topological and metric methods. Topological localization provides coarse estimates of the position, usually by dividing the map into a discretized set of locations and employing image retrieval techniques [271, 272, 273]. While this approach is well suited for large environments, the resulting location accuracy is bounded by the granularity of the discrete set. Metric localization approaches on the other hand, provide a 6-DoF metric estimate of the pose within the environment. Currently, local feature-based approaches that utilize SfM information achieve state-of-the-art performance [274, 275]. However, a critical drawback of these methods is the decrease in speed and increase in complexity of finding feature correspondences as the size of the environment grows. Moreover, most approaches require a minimum number of matches to be able to produce a pose estimate. This in turn causes pose estimation failures when there is large viewpoint changes, motion blur, occlusions or textureless environments.

Inspired by the outstanding performance of Convolutional Neural Networks (CNNs) in a variety of tasks in various domains and with the goal of eliminating manual engineering of algorithms for feature selection, CNN architectures that directly regress the 6-DoF metric pose have recently been explored [276, 277, 278]. However, despite their ability to handle challenging perceptual conditions and effectively manage large environments, they are still unable to match the performance of state-of-the-art local feature-based localization methods [274, 275]. This can be attributed to their inability to internally model the 3D structural constraints of the environment while learning from a single monocular image. As a solution to this problem, we propose a principled approach to embed geometric knowledge into the pose regression model by simultaneously learning to estimate visual odometry as an auxiliary task. We then leverage the complementary relative motion information from odometry to constrict the search space while training the global localization model using our Geometric Consistency loss function which is a result of joint work with Noha Radwan [56]. However, this problem is non-trivial as we need to first determine how to structure the architecture to ensure the learning of this inter-task correlation and secondly, how to jointly optimize the unified model since different task-specific networks have different attributes and different convergence rates.

We address the aforementioned challenges and propose our VLocNet architecture consisting of a global pose regression stream and a Siamese-type relative pose estimation stream. Our network based on the residual learning framework, takes two consecutive monocular images as input and jointly regresses the 6-DoF global pose as well as the 6-DoF relative pose between the images. We incorporate a hard parameter sharing scheme to learn inter-task correlations within the network and present a multitask joint optimization strategy for learning shared features across the different task streams. More importantly, we devise the new Geometric Consistency loss [56] for global pose regression that incorporates the relative motion information during training and enforces the predicted poses to be geometrically consistent with respect to the true motion model. By jointly learning both tasks, our approach is robust to environmental aliasing by utilizing previous pose and relative motion information, thereby combining the advantages of both local feature and deep learning-based localization methods.

Inspired by how humans often describe their location to one another with respect to reference landmarks in the scene and giving their position relative to it, we further explore encoding semantic knowledge into the pose regression model in our proposed VLocNet++ architecture. We now formulate this problem from a multitask learning (MTL) perspective with the goal of learning more accurate localization and semantic segmentation models by leveraging the predicted ego-motion. This problem is even more challenging as it involves simultaneously learning cross-domain tasks that perform pixel-wise classification and regression with different units and scales. However, this joint formulation enables inter-task learning which improves both generalization capabilities and alleviates the problem of requiring vast amounts of labeled training data, which is especially hard to obtain in the robotics domain. Moreover, as robots are equipped with limited resources, a joint model is more efficient for deployment and enables online inference on a consumer grade GPU.

Our motivation for jointly estimating semantics is based on the premise that it can instill structural cues about the environment into the pose regression network and implicitly pull the attention towards more informative regions in the scene. Correspondingly, location-specific information from the localization task can help improve learning of semantics. A popular paradigm employed for semantics-aware localization is to extract predefined features, emphasize on stable features [279] or combine them with local features [280]. Although these handcrafted solutions have demonstrated considerable reliability, their performance suffers substantially when the predefined structures are occluded or not visible in the scene. Therefore, in order to alleviate this problem, we propose a weighted fusion layer to integrate relevant semantic features into the global pose regression stream not only based on the semantic object category, but also the activations in the region.

Predicting consistent semantics is a critical prerequisite for semantic visual localization. Inspired by early cognitive studies in humans showing the importance of learning self-motion for acquiring basic perceptual skills [281], we utilize a novel self-supervised semantic context aggregation technique (joint work with Noha Radwan [34]) leveraging the predicted relative motion from the odometry stream of our network. Using pixel-wise depth predictions from a CNN [243] and differential warping, we fuse intermediate network representations from the previous timestep into the current frame using our proposed weighted fusion layer. This enables our semantic segmentation network to aggregate more scene-level context, thereby improving the performance and leading to faster convergence. Additionally, in order to efficiently utilize the learned motion-specific features in the global pose regression stream from the previous timestep, we also employ the weighting technique to aggregate motion-specific temporal information.

To the best of our knowledge, there is no publicly available localization or pose estimation dataset with pixel-level semantic groundtruth labels tagged with 6-DoF camera poses and containing multiple loops. Therefore, in order to facilitate training of multitask deep learning models for semantic visual localization and odometry estimation, we introduce the new DeepLoc dataset [34] that was gathered using our Obelix robot platform and consists of ten loops amounting a total of 3910 RGB-D images with pixel-level semantic and 6-DoF pose groundtruth labels. The dataset contains repetitive, translucent and reflective surfaces, weakly textured regions and low-lighted scenes with shadows, thereby making it extremely challenging for benchmarking a variety of tasks. We evaluate our proposed VlocNet and VLocNet++ architectures on the challenging indoor Microsoft 7-Scenes benchmark and our outdoor DeepLoc dataset on each of the diverse tasks. Extensive empirical evaluations demonstrate our VLocNet architecture is the first deep learning-based localization method to perform on par with local feature-based techniques while outperforming existing deep learning-based approaches. Furthermore, our proposed VLocNet++ architecture sets the new state-of-the-art outperforming both local feature-based and CNN-based techniques while simultaneously performing multiple tasks and exhibiting substantial robustness in challenging scenarios. The work that we present in this chapter is the first attempt to show that a joint multitask model can precisely and efficiently outperform its task-specific counterparts for global pose regression, semantic segmentation and visual odometry estimation.

In summary, the primary contributions that we make in this chapter are as follows:

- A new residual convolutional neural network architecture for 6-DoF global pose regression.
- A new residual Siamese-type convolutional neural network architecture for 6-DoF odometry estimation.
- The novel VLocNet architecture with a joint optimization strategy for simultaneously regressing the global pose and visual odometry. Our approach presents an efficient, and scalable alternative to learning task specific models.
- The novel VLocNet++ architecture for jointly learning semantics, visual localization and odometry from consecutive monocular images.
- A novel weighted fusion layer for element-wise fusion of feature maps based on region activations to exploit inter/intra-task dependencies.
- A first-of-a-kind urban outdoor localization dataset [34] consisting of multiple loops with pixel-level semantic labels and 6-DoF camera poses.
- Comprehensive quantitative and qualitative comparisons of our task specific networks as well as our joint multitask network against various CNN-based approaches as well as state-of-the-art local feature-based techniques on benchmark datasets.

Additionally, we also detail the following contributions which are an outcome of joint work with Noha Radwan [34, 56]:

- The Geometric Consistency loss function [56] that incorporates the relative motion information during training, thus enabling the network to predict pose estimates that are consistent with the true motion model.

- A self-supervised context aggregation technique [34] based on differential warping that improves the semantic segmentation performance and reduces the training time.

The remainder of this chapter is organized as follows. In Section 7.2, we first detail our Geometric Consistency loss function, followed by the topologies of our proposed VLocNet and VLocNet++ architectures. We then describe the methodology that we employed for collecting and annotating our DeepLoc dataset in Section 7.3. In Section 7.4, we present extensive experimental evaluations, detailed ablation studies and insightful qualitative results. Subsequently, we discuss the related work in Section 7.5 and conclude the chapter in Section 7.6.

## 7.2 Technical Approach

In this section, we first introduce our Geometric Consistency loss function [56] for global pose regression that constricts the search space while training by leveraging the relative motion between two consecutive frames. We then describe our VLocNet architecture for regressing global poses and simultaneously learning to regress relative motion between two camera frames using only pairs of RGB images. Subsequently, we detail our multitask VLocNet++ architecture for jointly estimating the global pose, odometry and semantic segmentation from consecutive monocular images. While the multitask networks presented in this chapter focus on joint learning of the aforementioned tasks, each of the task-specific models can be deployed independently during test-time.

### 7.2.1 The Geometric Consistency Loss Function

Learning both translational and rotational pose components with the same loss function is inherently challenging due to the difference in scale and units between both the quantities. Equations 7.1 and 7.2 describe the loss function for regressing the translational and rotational components in the Euclidean space in which we assume that the quaternion output of the network has been normalized a priori for ease of notation.

$$\mathcal{L}_x (f(\theta \mid I_t)) := \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|_\gamma \tag{7.1}$$

$$\mathcal{L}_q (f(\theta \mid I_t)) := \|\mathbf{q}_t - \hat{\mathbf{q}}_t\|_\gamma , \tag{7.2}$$

where $\theta$ is the parameters of the network, $f(\theta \mid I_t)$ denotes the predicted output of the network for image $I_t$, $\mathbf{x}_t \in \mathbb{R}^3$ and $\mathbf{q}_t \in \mathbb{R}^4$ denote the groundtruth translation and rotation components of the pose, $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{q}}_t$ denote their predicted counterparts and $\gamma$ refers to the $L^\gamma$-norm. In this chapter, we use the $L^2$ Euclidean norm. Recent work [282] has shown that the performance of a model trained to jointly regress the position and orientation components of the pose, outperforms two separate models trained for regressing each quantity. Therefore, as the loss function is required to learn both the position and orientation components simultaneously, a weight regularizer $\beta$ is used to balance each of the loss terms. We represent this loss function as

$$\mathcal{L}_{euc_\beta}(f(\theta \mid I_t)) := \mathcal{L}_x(f(\theta \mid I_t)) + \beta \mathcal{L}_q(f(\theta \mid I_t)). \tag{7.3}$$

Although initial work [276, 277, 283] has shown that by minimizing this function, the network is able to learn a valid pose regression model, it suffers from the drawback of having to manually tune the hyperparameter $\beta$ for each new scene in order to achieve reasonable results. To counteract this problem, we use two learnable weightings to replace the $\beta$ term. As these weightings are learned, their values get updated during the optimization process and consequently do not require manual tuning. The resulting loss function is represented as

$$\begin{aligned}\mathcal{L}_{euc}(f(\theta \mid I_t)) := \mathcal{L}_x(f(\theta \mid I_t)) \exp(-\hat{s}_x) + \hat{s}_x \\ + \mathcal{L}_q(f(\theta \mid I_t)) \exp(-\hat{s}_q) + \hat{s}_q,\end{aligned} \tag{7.4}$$

where $\hat{s}_x$ and $\hat{s}_q$ are the two learnable weighting variables for the translation and rotational components. Although this formulation overcomes the problem of having to manually select a $\beta$ value for each scene, it does not ensure that the estimated poses are consistent with the previous motion.

As a solution to this problem, we utilize our novel Geometric Consistency loss function [56] that incorporates previous motion information, thereby yielding globally consistent pose estimates. We introduce an additional geometric pose constraint between pairs of observations that bootstraps the loss function by penalizing pose predictions that contradict the relative motion. More precisely, in addition to the loss function shown in Eq. (7.4), we add another loss term to constrain the current pose prediction by minimizing the relative motion error between the groundtruth and the estimated motion from the auxiliary odometry stream. We use $\mathcal{R}_x(f(\theta \mid I_t))$ and $\mathcal{R}_q(f(\theta \mid I_t))$ to denote the relative motion between the previous predicted pose $\hat{\mathbf{p}}_{t-1}$ and the pose of the current image $I_t$ as

$$\mathcal{R}_x(f(\theta \mid I_t)) := \hat{\mathbf{x}}_t - \hat{\mathbf{x}}_{t-1} \tag{7.5}$$

$$\mathcal{R}_q(f(\theta \mid I_t)) := \hat{\mathbf{q}}_{t-1}^{-1} \hat{\mathbf{q}}_t. \tag{7.6}$$

The components from Equations 7.5 and 7.6 compute the relative motion in terms of the network's predictions. By utilizing the predictions of the network from the previous timestep along with the current timestep, the relative motion relative motion loss term $\mathcal{L}_{rel}\,(f\,(\theta \mid I_t))$ can be computed as a weighted summation of the translational and rotational errors which minimizes the variance between the predicted poses. The corresponding loss function is formulated as

$$\mathcal{L}_{x_{rel}}(f(\theta \mid I_t)) := \left\|\mathbf{x}_{t-1,t} - \mathcal{R}_x(f(\theta \mid I_t))\right\|_\gamma \tag{7.7}$$

$$\mathcal{L}_{q_{rel}}(f(\theta \mid I_t)) := \left\|\mathbf{q}_{t-1,t} - \mathcal{R}_q(f(\theta \mid I_t))\right\|_\gamma \tag{7.8}$$

$$\begin{aligned}\mathcal{L}_{rel}\,(f\,(\theta \mid I_t)) = {} & \mathcal{L}_{x_{rel}}\,(f\,(\theta \mid I_t))\exp(-\hat{s}_{x_{rel}}) + \hat{s}_{x_{rel}} \\ & + \mathcal{L}_{q_{rel}}\,(f\,(\theta \mid I_t))\exp(-\hat{s}_{q_{rel}}) + \hat{s}_{q_{rel}}\end{aligned} \tag{7.9}$$

where $\mathcal{L}_{x_{rel}}$ computes the difference between the ground-truth relative translational motion and its predicted counterpart, while $\mathcal{L}_{q_{rel}}$ computes a similar difference for the rotational component of the pose. We combine both the relative motion loss term with the loss function from Eq. (7.4) to yield the proposed Geometric Consistency loss, thereby minimizing

$$\begin{aligned}\mathcal{L}_{geo}(f(\theta \mid I_t)) := {} & \left(\mathcal{L}_x(f(\theta \mid I_t)) + \mathcal{L}_{x_{rel}}(f(\theta \mid I_t))\right)\exp(-\hat{s}_x) + \hat{s}_x \\ & + \left(\mathcal{L}_q(f(\theta \mid I_t)) + \mathcal{L}_{q_{rel}}(f(\theta \mid I_t))\right)\exp(-\hat{s}_q) + \hat{s}_q.\end{aligned} \tag{7.10}$$

By minimizing the aforementioned Geometric Consistency loss function [56], our network learns a model that is geometrically consistent with respect to the motion. We hypothesize that the resulting trained model is more robust to perceptual aliasing within the environment by utilizing this relative motion in the loss function.

## 7.2.2 VLocNet Architecture

Our proposed VLocNet architecture consists of three network streams; a global pose regression stream and a Siamese-type double-stream for odometry estimation. An overview of the topology is shown in Figure 7.2. Given a pair of consecutive monocular images $(I_{t-1}, I_t)$, our network predicts both the global pose $\mathbf{p}_t = (\mathbf{x}_t, \mathbf{q}_t)$ and the relative pose $\mathbf{p}_{t-1,t} = \left(\mathbf{x}_{t-1,t}, \mathbf{q}_{t-1,t}\right)$ between the input frames, where $\mathbf{x} \in \mathbb{R}^3$ denotes the translation and $\mathbf{q} \in \mathbb{R}^4$ denotes the rotation in quaternion representation. For ease of notation, we assume that the quaternion outputs of the network have been normalized a priori. The input to the Siamese streams are the images $(I_{t-1}, I_t)$, while the input to the global pose regression stream is $I_t$.

Unlike conventional Siamese architectures, we do not share features across the two temporal streams, instead we share features across the global pose regression stream and the Siamese stream that takes the image $I_t$ as input, upto a certain network depth. As our images do not encompass any spatially centered structure, sharing features across these streams is a viable solution to facilitate auxiliary learning and joint optimization of the task specific network streams. We employ our Geometric Consistency loss function [56] that enforces global consistency among the predicted poses by utilizing the relative pose estimates while training. In the remainder of this section, we detail the constituting components of our VLocNet architecture.

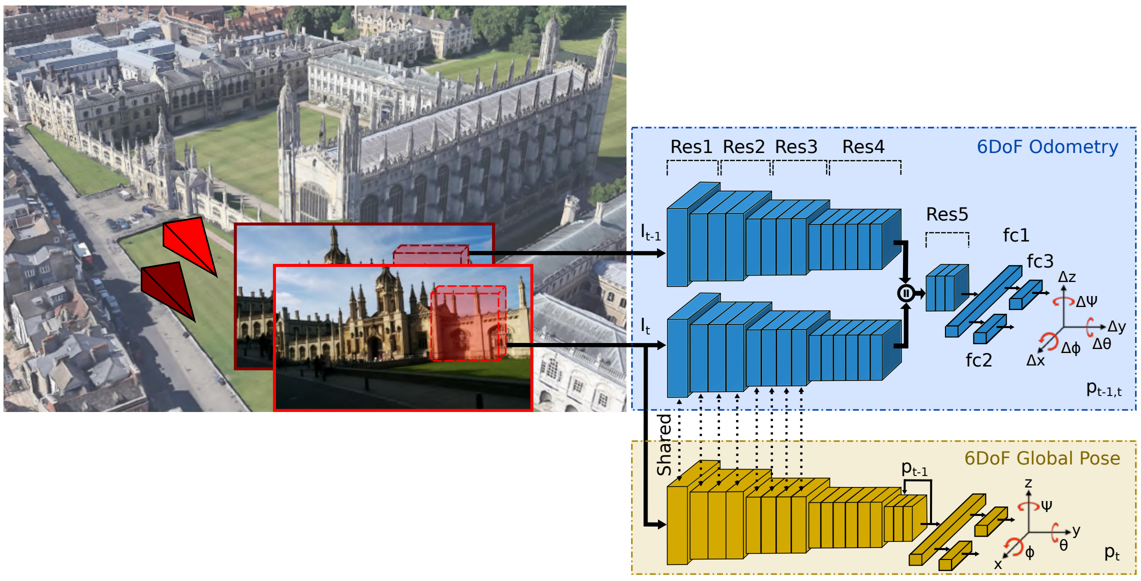


**Figure 7.2:** Topology of our VLocNet architecture for 6-DoF visual localization and odometry estimation. Our network takes two consecutive monocular images $(I_{t-1}, I_t)$ as input and regresses the 6-DoF global pose $p_t$ and 6-DoF odometry $p_{t-1,t}$ simultaneously. The global pose and odometry streams incorporate hard parameter sharing and utilize our Geometric Consistency loss function that is robust to environmental aliasing.

#### 7.2.2.1 Geometrically Consistent Pose Regression

In this section, we describe the topology of our global pose regression stream, which given an input image $I_t$ and the predicted pose from the previous timestep $\hat{\mathbf{p}}_{t-1}$, estimates the 7-dimensional pose $\hat{\mathbf{p}}_t$. Similar to previous works [276, 277], $\mathbf{p}$ is defined relative to an arbitrary global reference frame. We chose the quaternion representation for orientation due to the ease of mapping them back to rotations, since we can easily normalize its four dimensional values to unit length to yield a valid quaternion.

To estimate the global pose, we build upon the ResNet-50 [27] architecture with the following reconfigurations. The topology of our network is similar to ResNet-50 truncated before the last average pooling layer. The architecture is comprised of five residual blocks with multiple residual units, where each unit has a bottleneck architecture consisting of three convolutional layers in the following order: $1 \times 1$ convolution, $3 \times 3$ convolution, $1 \times 1$ convolution. Each of the convolutions is followed by batch normalization, scale and Rectified Linear Unit (ReLU). We modify the standard residual block structure by replacing ReLUs with Exponential Linear Units (ELUs) [66]. ELUs help in reducing the bias shift in the neurons, in addition to avoiding the vanishing gradient and yield faster convergence. We replace the last average pooling layer with global average pooling and subsequently add three inner-product layers, namely *fc1*, *fc2* and *fc3*. The first inner-product layer *fc1* has 1024 outputs and the following two inner-product layers have corresponding output dimensions of 3 and 4 for regressing the translation $\mathbf{x}$ and rotation $\mathbf{q}$ respectively. Our Geometric Consistency loss [56], detailed in Section 7.2.1 ensures that the predicted pose is geometrically consistent with respect to the true motion. Therefore, we feed the predicted pose from the previous timestep to the network so that it can better learn about spatial relations of the environment. In order to effectively train the network, we use the groundtruth pose from the previous timestep during training and the predicted pose from the model during evaluation. We do not incorporate recurrent units into our network as our aim in this work is to localize only using consecutive monocular images and not rely on long-term temporal features. We first feed the previous predicted pose to an inner-product layer *fc4* of dimension $D$ and reshape the outputs to $H \times W \times C$, which corresponds in shape to the output of the last residual unit before the downsampling stage. Both the tensors are then concatenated and fed to the subsequent residual unit. In total, there are four downsampling stages in our network and we experiment with fusing at each of these stages in Section 7.4.5.2. Note that we denote the aforementioned architecture as VLocNet$_{\text{STL}}$ in our experiments.

#### 7.2.2.2 Learning Visual Odometry

Our proposed architecture for relative pose estimation takes a pair of consecutive monocular images $(I_{t-1}, I_t)$ as input and yields an estimate of ego-motion $\mathbf{p}_{t-1,t} = \left(\mathbf{x}_{t-1,t}, \mathbf{q}_{t-1,t}\right)$. We employ a dual-stream architecture in which each of the streams is identically similar in structure and is based on the ResNet-50 model. We concatenate the feature maps of the individual streams before the last downsampling stage at the end of *Res4* and convolve them through the last residual block, followed by an inner-product layer and two regressors for estimating the pose components. During training, we optimize the following loss function by minimizing the Euclidean loss between the groundtruth and the predicted relative poses during training as

$$\begin{aligned}
\mathcal{L}_{vo}\left(f\left(\theta \mid I_{t-1}, I_t\right)\right) &:= \mathcal{L}_x\left(f\left(\theta \mid I_{t-1}, I_t\right)\right) \exp(-\hat{s}_{x_{vo}}) + \hat{s}_{x_{vo}} \\
&\quad + \mathcal{L}_q\left(f\left(\theta \mid I_{t-1}, I_t\right)\right) \exp(-\hat{s}_{q_{vo}}) + \hat{s}_{q_{vo}} \\
\mathcal{L}_x\left(f\left(\theta \mid I_{t-1}, I_t\right)\right) &:= \|\mathbf{x}_{t-1,t} - \hat{\mathbf{x}}_{t-1,t}\|_2 \\
\mathcal{L}_q\left(f\left(\theta \mid I_{t-1}, I_t\right)\right) &:= \|\mathbf{q}_{t-1,t} - \hat{\mathbf{q}}_{t-1,t}\|_2 ,
\end{aligned} \tag{7.11}$$

where $\mathcal{L}_x$ and $\mathcal{L}_q$ refers to the translational and rotational components respectively. We also employ learnable weighting parameters to balance the scale between the translational and rotational components in the loss term. As shown in Figure 7.2, the dual odometry streams have an architecture similar to the global pose regression network. In order to enable the inductive transfer of information between both networks, we share parameters between the odometry stream taking the current image $I_t$ and the global pose regression network as detailed in Section 7.2.2.3.

#### 7.2.2.3 Deep Auxiliary Learning

The idea of jointly learning both the global pose and visual odometry streams from the inherent similarities across both tasks in the feature space. More importantly, sharing features across both networks can enable a competitive and collaborative action as each network updates its own weights during backpropagation in an attempt to minimize the distance to the groundtruth pose. This symbiotic action introduces additional regularization while training, thereby alleviating the problem of overfitting. Contrary to the approaches that use a two stream shared Siamese network for visual odometry estimation, we do not share weights between the two temporal streams, rather we share weights between the stream that takes the image $I_t$ from the current timestep as input and the global pose regression stream. By learning separate discriminative features in each timestep before learning the correlation between them, the visual odometry network is able to effectively generalize to challenging corner cases containing motion blur and perceptual aliasing. The global pose regression network also benefits from this feature sharing, as the shared weights are pulled more towards areas of the image from which the relative motion can be easily estimated. This has a tremendous impact on the accuracy of the predicted global pose in multiple scenarios. Consider the following situation in which the network attempts to estimate the pose of an image in a textureless structurally symmetric environment. Due to the presence of perceptual aliasing, the accuracy of the predicted pose can be substantially lower, while compared to the predictions obtained in an environment with abundant structural variations. However, by jointly training both networks using our Geometric Consistency loss function [56] coupled with hard parameter sharing between both network streams, the global pose regression network encodes relative motion information from the odometry stream, thereby yielding more accurate pose estimates. We denote the aforementioned architecture as VLocNet++$_{\text{MTL}}$ in our experiments.

While sharing features across multiple networks can be inferred as a form of regularization, it is not clear a priori for how many layers should we maintain a shared stream. Sharing only a few initial layers does not have any additive benefit to either network, as early layers learn very generic feature representations. On the other hand, maintaining a shared stream too deep into the network can negatively impact the performance of both tasks, since the features learned at the stages towards the end are more task specific and have lesser feature location information. In this work, we studied the impact of sharing features across both network streams and experimented with varying the amount of feature sharing. We detail the results from this experiment in Section 7.4.5.3. Another critical aspect of auxiliary learning relates to the strategy to be employed for joint optimization. We detail our optimization procedure in Section 7.4.2. Finally, during inference, the joint model can be deployed coherently or each task-specific network individually, since the relative pose estimates are only used in the loss function and there is no inter-network dependency in terms of concatenating or adding features from either task-specific streams.

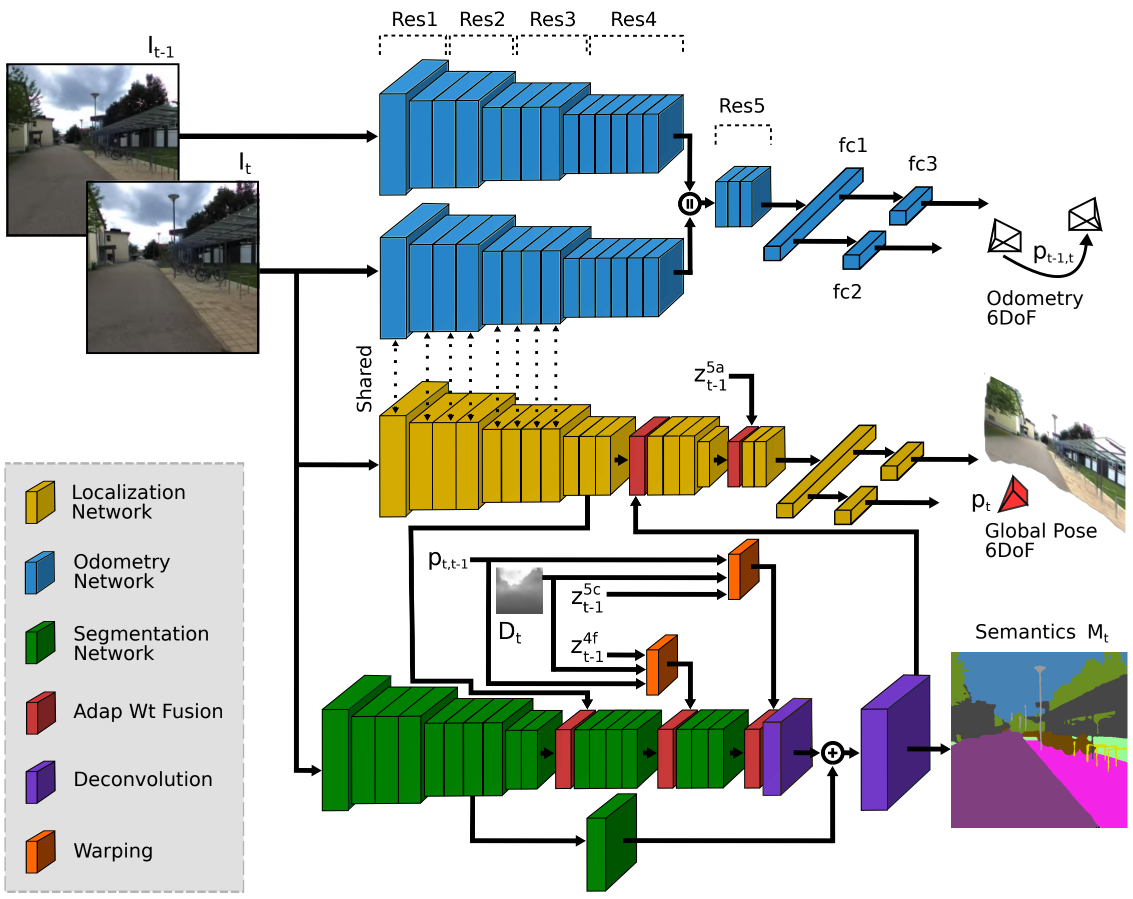


**Figure 7.3:** Schematic representation of our proposed VLocNet++ architecture. The network takes two consecutive monocular images $(I_t, I_{t-1})$ as input and simultaneously predicts the 6-DoF global pose $p_t$, 6-DoF odometry $p_{t-1,t}$ and semantics $M_t$ of the scene. The variable $z^l_{t-1}$ denotes the feature maps of layer $l$ from the previous timestep and $D_t$ denotes a predicted depth map that is used for representational warping in the semantic stream.

## 7.2.3 VLocNet++ Architecture

In this section, we describe our proposed VLocNet++ architecture that introduces a novel strategy for encoding semantic and geometric constraints into the pose regression network. Namely, by incorporating information from the previous timesteps to leverage motion-specific information and by adaptively fusing semantic features based on the activations in the region using our proposed weighted fusion scheme. As being able to predict robust semantics is an essential prerequisite, we present a new self-supervised warping technique [34] for aggregating scene-level context in the semantic segmentation model. Our architecture, depicted in Figure 7.3 consists of four CNN streams; a global pose regression stream, a semantic segmentation stream and a Siamese-type double stream for visual odometry estimation.

Given a pair of consecutive monocular images $I_{t-1}, I_t \in \mathbb{R}^{\rho}$, the pose regression stream predicts the global pose $\mathbf{p}_t = (\mathbf{x}_t, \mathbf{q}_t)$ for image $I_t$, where $\mathbf{x} \in \mathbb{R}^3$ denotes the translation and $\mathbf{q} \in \mathbb{R}^4$ denotes the rotation in quaternion representation, while the semantic stream predicts a pixel-wise segmentation mask $M_t$ mapping each pixel $u$ to one of the $C$ semantic classes, and the odometry stream predicts the relative motion $\mathbf{p}_{t-1,t} = \left(\mathbf{x}_{t-1,t}, \mathbf{q}_{t-1,t}\right)$ between consecutive input frames. While, $z^l$ denotes the feature maps from layer $l$ of a particular stream. In order to estimate the odometry in VLocNet++, we employ our proposed Siamese-type double stream architecture that we described in Section 7.2.2.2. However, we use the full pre-activation residual units [72] as opposed to the standard residual units [27] that we used in VLocNet. In the remainder of this section, we describe rest of the constituting components of our network architecture and our multitask learning scheme.

### 7.2.3.1 Geometrically Consistent Pose Regression

Our model for regressing the global pose is built upon our VLocNet architecture described in Section 7.2.2.1. It has five residual blocks that downsample the feature maps by half at each block, similar to the full pre-activation ResNet-50 architecture [72], as opposed to the standard ResNet-50 architecture that we used in VLocNet. VLocNet++ also employs Exponential Linear Units (ELUs) for the activation function and a global average pooling layer is added after the fifth residual block. Subsequently, we add three inner-product layers *fc1, fc2 and fc3* with the number of output units as 1024, 3 and 4 respectively, where *fc2* and *fc3* regress the translational $\mathbf{x}$ and rotational $\mathbf{q}$ components of the pose. Unlike VLocNet which fuses the previous predicted pose directly using inner-product layers, we adopt a more methodological approach in VLocNet++ to provide the network with this prior. Fusing the previous prediction directly, inhibits the network from being able to correlate the critical motion-specific spatial relations with that of the previous timestep as the network does not retain these features thereafter. In this work, we leverage the network's intermediate representation $z_{t-1}^{5a}$ from the last downsampling stage (*Res5a*) of the previous timestep using our proposed weighted fusion layer that we detail in Section 7.2.3.3. Our fusion scheme learns the optimal element-wise weighting for this fusion, and when it is trained end-to-end with the Geometric Consistency loss function, it enables aggregation of motion-specific features across the temporal dimension. We denote the aforementioned architecture as VLocNet++$_{\text{STL}}$ in our experiments.

As opposed to naively minimizing the Euclidean loss between the predicted poses and the groundtruth, we employ our Geometric Consistency loss function [56] described in Section 7.2.1, which in addition to minimizing the Euclidean loss, adds another loss term to constrain the current pose prediction by minimizing the relative motion error between the groundtruth and the estimated motion from the odometry stream. This enables our network to learn a model that is geometrically consistent with respect to the motion. Moreover, by employing a mechanism to aggregate motion-specific features temporally, we enable the Geometric Consistency loss function to efficiently leverage this information.

#### 7.2.3.2 Incorporating Semantics

Our model for learning consistent semantics is comprised of three components: a base segmentation architecture (green and purple blocks in Figure 7.3), our proposed self-supervised warping layer (orange blocks) and weighted fusion layer (red blocks). In the remainder of this section, we first describe the single-task semantic segmentation architecture and then detail our proposed self-supervised warping scheme [34]. The weighted fusion layer is described in the section that follows.

**Network Architecture:** For the single-task base model, we employ two variants based on our AdapNet and AdapNet++ architectures that we presented in Chapter 4. We denote these two variants as VLocNet++(base) and VLocNet++$_{\text{MTL}}$ respectively. The networks follow the general encoder-decoder design principle where the encoder learns highly discriminative semantic features and yields an output 16-times downsampled with respect to the input dimensions. While the decoder upsamples the output of the encoder back to the input image resolution using deconvolution layers and skip refinement stages. Similar to the global pose regression stream, the encoder in the VLocNet++(base) model is built upon the ResNet-50 [27] architecture and the encoder in the VLocNet++$_{\text{MTL}}$ is built upon the full pre-activation ResNet-50 model [72]. Both architectures include skip connections and batch normalization layers that enable training of deep architectures by alleviating the vanishing gradient problem. We also incorporate our multiscale residual units that have dilated convolutions parallel to the $3 \times 3$ convolutions for aggregating features from different spatial scales, while concurrently maintaining fast inference times. In addition, the encoder of VLocNet++$_{\text{MTL}}$ includes our efficient atrous spatial pyramid pooling that captures long-range context and has a large effective receptive field. We refer the reader to Chapter 4 for more details regarding the encoder-decoder topologies.

We represent the training set for semantic segmentation as $\mathcal{T} = \{(I_n, M_n) \mid n = 1, \dots, N\}$, where $I_n = \{u_r \mid r = 1, \dots, \rho\}$ denotes the input frame and the corresponding groundtruth mask $M_n = \{m_r^n \mid r = 1, \dots, \rho\}$, where $m_r^n \in \{1, \dots, C\}$ is the set of semantic classes. Let $\theta$ be the network parameters consisting of weights and biases, and $s_j(u_r, \theta)$ as the score assigned for labeling pixel $u_r$ with label $j$. We obtain the probabilities $\mathbf{P} = (p_1, \dots, p_C)$ for all the semantic classes using the softmax function $\sigma(.)$ as

$$p_j(u_r, \theta \mid I_n) = \sigma\left(s_j\left(u_r, \theta\right)\right) = \frac{exp\left(s_j\left(u_r, \theta\right)\right)}{\sum_k^C exp\left(s_k\left(u_r, \theta\right)\right)}. \tag{7.12}$$

The optimal network parameters are then estimated by minimizing the cross-entropy loss function as

$$\mathcal{L}_{seg}(\mathcal{T}, \theta) = -\sum_{n=1}^{N}\sum_{r=1}^{\rho}\sum_{j=1}^{C} \delta_{m_r^n, j} \log p_j(u_r, \theta \mid I_n), \tag{7.13}$$

for $(I_n, M_n) \in \mathcal{T}$, where $\delta_{m_r^n, j}$ is the Kronecker delta.

**Self-Supervised Warping:** We employ a temporal warping technique [34] to aggregate scene-level context in order to enable our segmentation network to learn consistent semantics. In this method, we first leverage the estimated relative pose from the odometry stream to warp semantic feature maps from the previous timestep into the current view using a depth map obtained from the DispNet [243] network and the warping concept from multi-view geometry. We then fuse the warped feature maps with the corresponding semantic representations from the current timestep using our proposed weighted fusion layer described in the following Section 7.2.3.3. We introduce the warping and fusion layers (red and orange blocks in Figure 7.3) at *Res4f* and *Res5c* to warp the corresponding feature maps $z_{t-1}^{4f}$ and $z_{t-1}^{5c}$ from the previous timestep into the current view and fuse them with the representations $z_t^{4f}$ and $z_t^{5c}$ of the current frame respectively. We experimentally identify these intermediate network stages to perform the semantic warping and report the results from these experiments in Section 7.4.5.4.

We employ bilinear interpolation as a sampling mechanism for warping to facilitate computation of gradients that are necessary for back-propagation. This sampling mechanism is fully differentiable and it does not require any pre-computation for training. In order to enable our self-supervised warping technique to be computationally more efficient, we only compute the warping grid once at the input image resolution and we then use average pooling to downsample the grid to the required scales.

We formulate warping of a pixel $u_r$ into its warped representation $\hat{u}_r$ as mentioned in the work of Radwan *et al.* [34] using the relative pose $\mathbf{p}_{t-1,t}$ obtained from the odometry stream, the estimated depth map $D_t$ of the current image $I_t$, and the projection function $\pi$ as

$$\hat{u}_r := \pi\left(T\left(\mathbf{p}_{t-1,t}\right)\pi^{-1}\left(u_r, D_t\left(u_r\right)\right)\right). \tag{7.14}$$

The function $T\left(\mathbf{p}_{t-1,t}\right)$ denotes the homogenous transformation matrix of $\mathbf{p}_{t-1,t}$, $\pi$ denotes the projection function transforming from camera to image coordinates as $\pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ and $\pi^{-1}$ denotes the transformation from image to camera coordinates using the depth $D_t\left(u_r\right)$ at the pixel $u_r$. Our proposed temporal warping scheme enables our network to fuse feature maps from multiple views and resolutions thereby making our semantic segmentation model invariant to camera angle deviations, object scale and frame-level distortions. Moreover, it also implicitly introduces feature augmentation during training which facilitates the faster convergence of our network.

### 7.2.3.3 Deep Multitask Learning

Our main motivation for jointly learning the semantics, regressing the 6-DoF global pose and odometry is twofold: to enable inductive transfer by leveraging domain specific information and to enable the global pose regression network to encode geometric and semantic knowledge of the environment while training. In order to achieve this goal, we structure our multitask learning network to be interdependent on the outputs as well as the intermediate representations of the task-specific network streams. More concretely, as shown in Figure 7.3, we employ hard parameter sharing until the end of the *Res3* block between the global pose regression stream and the odometry stream that both receive the image $I_t$ from the current timestep. This enables the network to exploit similarities among these pose regression tasks and influences the shared weights of global pose regression network to integrate motion-specific features due to inductive bias from odometry estimation, in addition to effectuating implicit attention on regions that are more informative for relative motion estimation. We quantify the influence of sharing features between the global pose regression stream and the odometry stream for different network depths in Section 7.4.5.3.

A common practice employed for combining features from multiple layers or multiple networks is to perform concatenation of the tensors or element-wise addition/multiplication. Although this might be effective when both tensors contain sufficient relevant information, it often accumulates irrelevant features and its effectiveness highly depends upon the intermediate stages of the network where the fusion is performed. One of the key components of our multitask learning network is the proposed weighted fusion layer which learns optimal element-wise weightings for the fusion based on the activations in the region, followed by a non-linear feature pooling over the weighted tensors. As opposed to spatial pooling, pooling in the feature space is a form of coordinate-dependent transformation which yields the same number of filters as the input tensor. While, suppressing or enhancing the features using learned weightings according to the region activations enables our fusion approach to discard irrelevant information.

For ease of notation, we formulate the mathematical representation of the weighted fusion layer with respect to two activation maps $z^a$ and $z^b$ from layers $a$ and $b$, while extending the notation to multiple activation maps is straightforward. The activation maps can be obtained from different layers in the same network or from layers of different task-specific networks. The output of the weighted fusion layer can be formulated as

$$\hat{z}_{fuse} = \max\left(\mathbf{W} * \left((w^a \odot z^a) \oplus \left(w^b \odot z^b\right)\right) + \mathbf{b}, 0\right), \tag{7.15}$$

where $w^a$ and $w^b$ are learned weightings having the same dimensions as $z^a$ and $z^b$; $\mathbf{W}$ and $\mathbf{b}$ are the parameters of the non-linear feature pooling; with $\odot$ and $\oplus$ representing per-channel scalar multiplication and concatenation across the channels; and $*$ representing the convolution operation. In other words, each channel of the activation map $z^a$ is first weighted and linearly combined with the corresponding weighted channels of the activation map $z^b$. Non-linear feature pooling is then applied which can be easily realized with existing layers in the form of a $1 \times 1$ convolution with a non-linearity such as ReLUs. As shown in Figure 7.3, we incorporate the weighted fusion layers (red blocks) at *Res4c* to fuse semantic features into the global pose regression stream. In addition, we also employ the proposed weighted fusion layer to fuse warped semantic feature maps from the previous timestep into the segmentation stream at the end of *Res3* and *Res4* blocks. In

Section 7.4.4, we demonstrate that over simple concatenation, our weighted fusion layer learns what features are relevant for both inter-task and intra-task fusion.

In order to jointly learn all the tasks in our network, we minimize the following combined loss function.

$$\begin{aligned}\mathcal{L}_{mtl} := \mathcal{L}_{loc}\exp(-\hat{s}_{loc}) + \hat{s}_{loc} + \mathcal{L}_{vo}\exp(-\hat{s}_{vo}) + \hat{s}_{vo}\\ + \mathcal{L}_{seg}\exp(-\hat{s}_{seg}) + \hat{s}_{seg},\end{aligned} \tag{7.16}$$

where $\mathcal{L}_{loc}$ is the global pose regression loss for which we use our Geometric Consistency loss function [56] defined in Eq. (7.10); $\mathcal{L}_{vo}$ is the visual odometry loss function from Eq. (7.11), and $\mathcal{L}_{seg}$ is the cross-entropy loss function for semantic segmentation defined in Eq. (7.13). Due to the inherent nature of the diverse tasks at hand, each of the associated task-specific loss terms has a different scale. If the task-specific losses were to be naively combined, the task-specific network with the highest scale would dominate the training and there would be little if no gain for any of the other tasks. To counteract this problem, we use learnable scalar weights $\hat{s}_{loc}, \hat{s}_{vo}, \hat{s}_{seg}$ to balance the scale of the loss terms.

## 7.3 DeepLoc Dataset

We introduce a large-scale urban outdoor localization dataset collected around the university campus in Freiburg. The dataset was collected using our Obelix robot equipped with a ZED stereo camera, an XSens IMU, a Trimble GPS Pathfinder Pro and several LiDARs. Stereo image pairs and depth images were captured at a resolution of $1280 \times 720$ pixels, at a frame rate of 20 Hz. The dataset was collected in an area spanning $110 \times 130$ m and contains multiple loops amounting to a total of ten sequences. Each sequence was collected with a different driving pattern to increase the diversity of the captured data and to have non-overlapping trajectories. The dataset was captured at different times of the day, therefore the images contain varying lighting conditions, reflective glare from the sun, orange dawn-sky and shadows, in addition to the motion-blur caused by the moving robot platform. Moreover, the environment that the dataset was collected in, contains structures such as buildings with similar facades, repetitive structures and translucent as well as reflective buildings made of glass. This renders the dataset challenging for a number of perception and localization tasks such as global localization, camera relocalization, semantic segmentation, visual odometry and loop closure detection. Additionally, many objects in the scene cause partial occlusion from different viewpoints which increases the difficulty of the semantic segmentation task. Example images from the dataset are shown in Figure 7.4.

In order to tag the images with 6-DoF camera poses, we used the sub-centimeter accurate LiDAR-based SLAM system from Kümmerle *et al.* [45]. The resulting pose labels are used as the groundtruth for training our deep networks. Furthermore, we

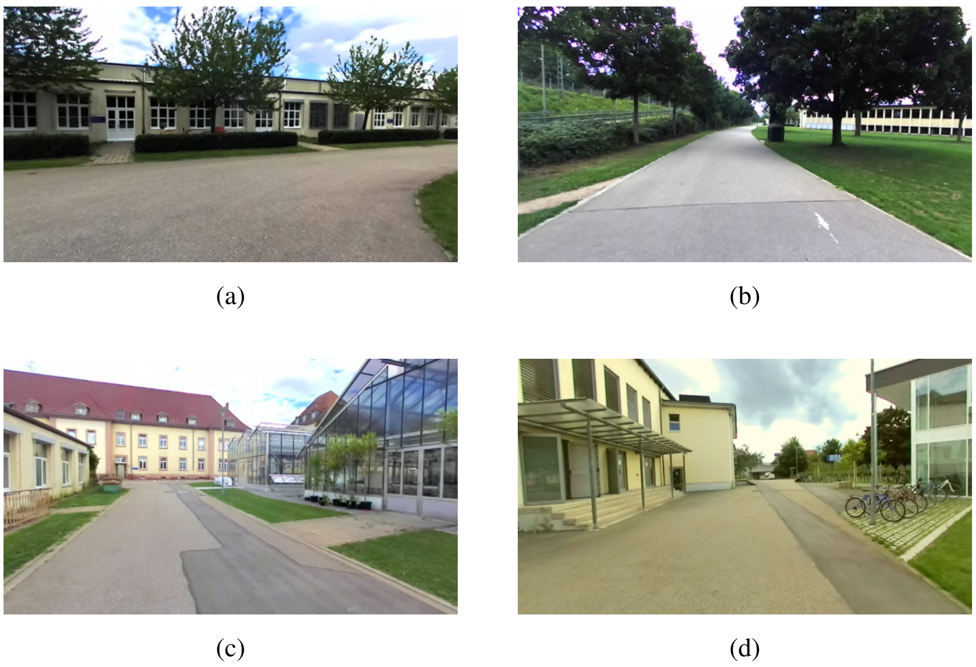


**Figure 7.4:** Example images from our DeepLoc dataset [34] that show challenging scenarios for global pose regression, visual odometry estimation and semantic segmentation. Several sections of the traversed area contain buildings with (a) repetitive patterns, (b) very few distinctive features, (c) structures made solely of glass and (d) large reflective glass surfaces as well as partially occluded structures such as bikes attached to bike-stands.

manually annotated each image with pixel-level semantic groundtruth labels for ten object categories: *background, sky, road, sidewalk, grass, vegetation, building, poles & fences, dynamic and other objects*. To the best of our knowledge, this is the first publicly available dataset containing images tagged with 6-DoF camera poses and pixel-level semantic segmentation labels for an entire scene with multiple loops. We divide the dataset into training and testing splits such that the training set consists of seven loops amounting to a total of 2737 images, while the test set consists of three loops with a total of 1173 images. We made the dataset publicly available at `http://deeploc.cs.uni-freiburg.de` and we hope that it enables future research in multitask and multimodel learning.

## 7.4 Experimental Evaluation

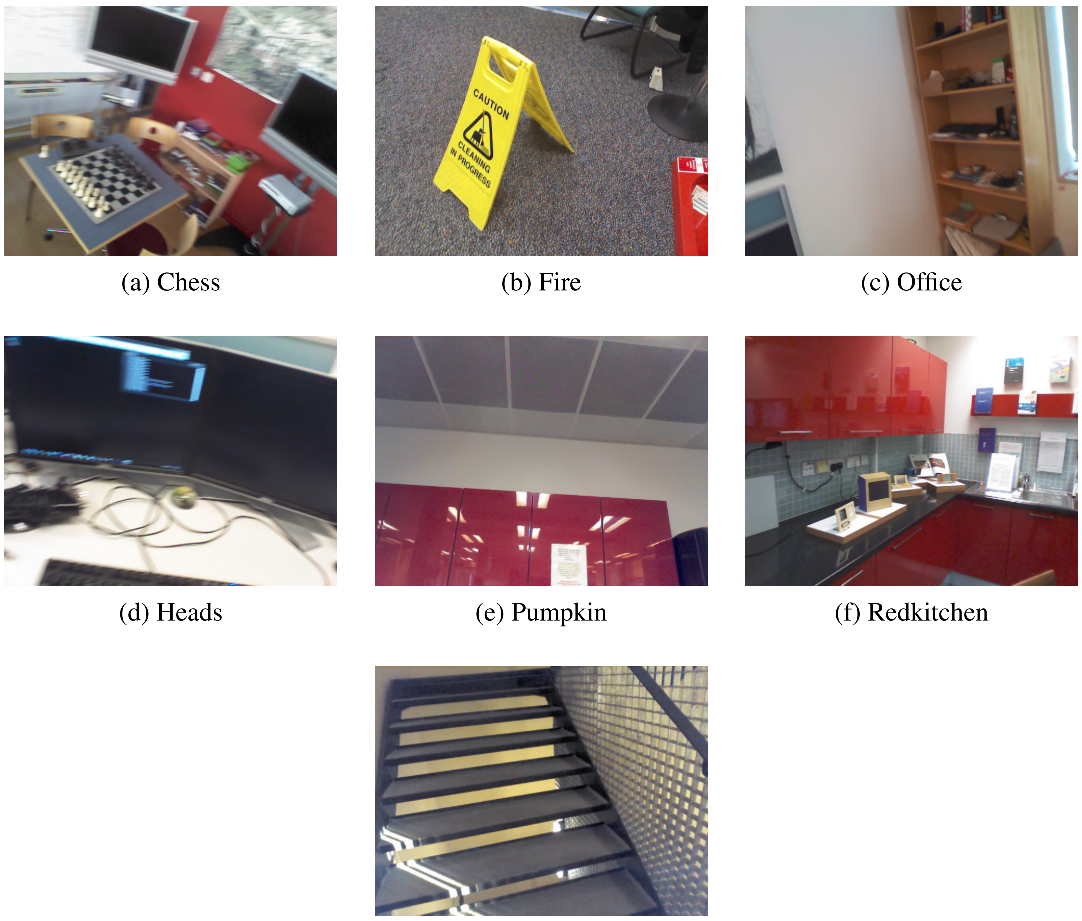
In this section, we first describe the datasets that we benchmark on in Section 7.4.1, followed by the training protocol that we employ in Section 7.4.2 and detailed comparisons on the performance of our single-task models against state-of-the-art methods in each corresponding task in Section 7.4.3. We then present comprehensive empirical evaluations of multitask learning in Section 7.4.4 and extensive ablation studies on the various architectural configurations of our models in Section 7.4.5. Furthermore, we present visualizations on the level of feature similarity in Section 7.4.6 as well as visualizations of the regression activation maps in Section 7.4.7. These visualizations demonstrate the efficacy of our approach and provide insights on the representations learned by our network. Finally, we present qualitative evaluations of localization and semantic segmentation in Section 7.4.8.

We use the TensorFlow [159] deep learning library for the implementations and all the experiments were carried out on a system with an Intel Xeon E5, 2.4 GHz and an NVIDIA TITAN X GPU. While we quantify the localization performance primarily using the median pose error metric, we also report the percentage of the predicted poses that are below 5 cm and 5°. We quantify the semantic segmentation performance using the standard Jaccard Index which is commonly known as average intersection-over-union (IoU) metric. It can be computed for each object class as $IoU = TP/(TP + FP + FN)$, where $TP, FP$ and $FN$ correspond to true positives, false positives and false negatives respectively. We also report the mean intersection-over-union (mIoU) for the empirical analysis. Additionally, a live demonstration of our proposed models is available at `http://deeploc.cs.uni-freiburg.de`.

## 7.4.1 Benchmark Datasets

Supervised learning techniques using CNNs require a large amount of training data with groundtruth annotations which is laborious to acquire. This becomes even more critical for jointly learning multiple diverse tasks which necessitate individual task-specific labels. Although there are publicly available task-specific datasets for visual localization and semantic segmentation, to the best of our knowledge there is a lack of a large enough dataset that contains both semantic and localization groundtruth with multiple loops in the same environment. To this end, in Section 7.3, we introduced the challenging *DeepLoc* dataset containing RGB-D images tagged with 6-DoF poses and pixel-level semantic labels of an outdoor urban environment. In addition to our new dataset, we also benchmark the performance of our localization network (without joint semantics learning) on the challenging Microsoft 7-Scenes dataset. We chose these datasets based on the criteria of having diversity in scene structure and environment as well as the hardware with which the images were captured.

**Microsoft 7-Scenes** dataset [284] is a widely used dataset for camera relocalization and tracking. It contains RGB-D image sequences tagged with 6-DoF camera poses of 7 different indoor environments. The data was captured with a Kinect camera at a resolution of $640 \times 480$ pixels and groundtruth poses were generated using KinectFusion [284]. Each of the sequences contains about 500 to 1000 frames. This dataset is very challenging as it contains textureless surfaces, reflections, motion blur and perceptual aliasing due to repeating structures. Figure 7.5 shows example images from the Microsoft 7-Scenes



(a) Chess (b) Fire (c) Office

(d) Heads (e) Pumpkin (f) Redkitchen

(g) Stairs

**Figure 7.5:** Example images from the Microsoft 7-Scenes benchmark showing challenging scenarios. The images exhibit significant motion blur (Chess and Heads), repetitive structures (Stairs), highly reflective surfaces (Pumpkin, Redkitchen) and low-texture regions (Fire, Office).

dataset. The images are representative of the challenges encountered in each scene. The images show scenarios that are challenging for global pose regression and visual odometry estimation such as substantial blur due to camera motion (Figures 7.5 (a) and (d)), perceptual aliasing due to repeating structures (Figure 7.5 (g)), textureless regions (Figures 7.5 (b) and (c)) and highly reflective surfaces (Figures 7.5 (e) and (f)).

We do not perform any pose augmentations such as pose synthesis [283] and synthetic view synthesis [285], as our initial experiments employing them did not demonstrate any improvement in performance on the aforementioned datasets. However, we randomly apply image augmentations on the semantic segmentation training data, similar to the techniques that we presented in Section 4.4.2. This includes rotation, translation, scaling, skewing, cropping, flipping, contrast and brightness modulation.

### 7.4.2 Network Training

In order to train our network on different datasets, we rescale the images maintaining the aspect ratio such that the shorter side is of length 256 pixels. We calculate the pixel-level mean for each of the scenes in the datasets and subtract them with the input images while training. We train our models from random crops of the image as it acts as a good regularizer helping the network generalize better in comparison to the synthetic augmentation techniques and while testing, we take the center crop of the images. We use the Adam solver for optimization with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-10}$. We employ a multi-stage training procedure and first train task-specific models individually using an initial learning rate of $\lambda_0 = 10^{-3}$ with a mini-batch size of 32 and a dropout probability of 0.2. Subsequently, we initialize the five residual blocks of our tasks-specific networks with weights from the ResNet-50 model pre-trained on the ImageNet dataset and the other layers with Xavier initialization [108]. We then initialize the joint MTL architecture with weights from the best performing single-task models and train with a lower learning rate of $\lambda_0 = 10^{-4}$.

Training deep networks in an end-to-end fashion is still a challenging problem, especially for tasks with limited amounts of training data. One approach to overcome this challenge is to leverage transfer learning and initialize the network with pretrained weights from the network trained on a larger dataset. However, the multitask network that we propose in this chapter is the first approach to address such wide range of tasks, therefore existing pretrained models cannot be leveraged to accelerate the training. In order to alleviate this problem, we investigate different weight initializations as bootstrapping methods for our training architecture. Results from this experiment are presented in Section 7.4.5.7.

In order to learn a unified model and to facilitate auxiliary learning, we employ different optimization strategies that allow for efficient learning of shared features as well as task-specific features, namely alternate training and joint training. In alternate training we use a separate optimizer for each task and alternatively execute each task optimizer on the task-specific loss function, thereby allowing synchronized transfer of information from one task to the other. This instills a form of hierarchy into the tasks, as the odometry sub-network improves the estimate of its relative poses, the global pose network in turn uses this estimate to improve its prediction. It is often theorized that this enforces commonality between the tasks. The disadvantage of this approach is that a bias in the parameters is introduced by the task that is optimized the last. In joint training on the other hand, we add each of the task-specific loss functions and use a single optimizer to train the sub-networks at the same time. The advantage of this approach is that the tasks are trained in a way that they maintain the individuality of their functions, but as each of our tasks is of different units and scale, the task with the larger scale often dominates the training. We evaluate the performance of our MTL model trained with different optimization strategies and discussed the results in Section 7.4.5.8. We train the models for a maximum of 120,000

**Table 7.1:** Median localization error for the task of 6-DoF visual localization on the Microsoft 7-Scenes dataset [34].

| Scene | LSTM-Pose [277] | PoseNet2 [282] | NNnet [286] | VLocNet$_{STL}$ (Ours) | VLocNet++$_{STL}$ (Ours) |
|---|---|---|---|---|---|
| Chess | 0.24 m, 5.77° | 0.13 m, 4.48° | 0.13 m, 6.46° | 0.036 m, 1.71° | **0.023** m, **1.44**° |
| Fire | 0.34 m, 8.99° | 0.27 m, 11.3° | 0.26 m, 12.72° | 0.039 m, 5.34° | **0.018** m, **1.39**° |
| Heads | 0.21 m, 13.7° | 0.17 m, 13.0° | 0.14 m, 12.34° | 0.046 m, 6.64° | **0.016** m, **0.99**° |
| Office | 0.30 m, 8.08° | 0.19 m, 5.55° | 0.21 m, 7.35° | 0.039 m, 1.95° | **0.024** m, **1.14**° |
| Pumpkin | 0.33 m, 7.00° | 0.26 m, 4.75° | 0.24 m, 6.35° | 0.037 m, 2.28° | **0.024** m, **1.45**° |
| RedKitchen | 0.37 m, 8.83° | 0.23 m, 5.35° | 0.24 m, 8.03° | 0.039 m, 2.20° | **0.025** m, **2.27**° |
| Stairs | 0.40 m, 13.7° | 0.35 m, 12.4° | 0.27 m, 11.82° | 0.097 m, 6.48° | **0.021** m, **1.08**° |
| Average | 0.31 m, 9.85° | 0.23 m, 8.12° | 0.21 m, 9.30° | 0.048 m, 3.80° | **0.022** m, **1.39**° |

iterations on a single NVIDIA Titan X GPU which approximately took 23 hours for the model to converge.

### 7.4.3 Comparison with the State-of-the-Art

In this section, we present empirical evaluations comparing our single-task models VLocNet$_{STL}$ and VLocNet++$_{STL}$ with other CNN-based methods for the task of visual localization and odometry estimation. We use the original train and test splits provided by each of the datasets. As we employ our semantic segmentation models from Chapter 4 for the single-task architecture, we report the improvement due to multitask learning in VLocNet++ in Section 7.4.4.3. Note that the results presented in this section are not the highest performance achieved by our models, the improvement achieved from multitask learning is presented in Section 7.4.4.

#### 7.4.3.1 Evaluation of Visual Localization

As a primary evaluation criteria, we first report results in comparison to deep learning-based methods on the publicly available Microsoft 7-Scenes (indoor) and DeepLoc (outdoor) datasets. Comparisons with local-feature based techniques are presented with the evaluation of multitask learning in Section 7.4.4.1. We analyze the performance in terms of the median translational and rotational errors for each of the scenes in the datasets. Table 7.1 shows the results for the Microsoft 7-Scenes dataset. Our proposed VLocNet$_{STL}$ model outperforms all the deep learning-based approaches in each of the scenes while achieving a median localization error of 0.048 m and 3.80° which amounts to an improvement of 77.14% in translation and 59.14% in rotational component of the pose. Moreover, our proposed single-task VLocNet++$_{STL}$ further outperforms VLocNet$_{STL}$ by achieving an

**Table 7.2:** Median localization error for the task of 6-DoF visual localization on the DeepLoc dataset [34].

| PoseNet [276] | Bayesian PoseNet [287] | SVS-Pose [285] | VLocNet$_{\mathrm{STL}}$ (Ours) | VLocNet++$_{\mathrm{STL}}$ (Ours) |
|---|---|---|---|---|
| 2.42 m, 3.66° | 2.24 m, 4.31° | 1.61 m, 3.52° | 0.68 m, 3.43° | **0.37** m, **1.93**° |

overall improvement of 54.17% in translational and 63.42% in rotational components of the pose. This substantial improvement can be attributed to the effective fusion of the previous pose information using our proposed weighted fusion layer. The largest improvement was obtained in the perceptually hardest scenes that contain textureless regions and reflective surfaces such as the *Fire* and *Pumpkin* scenes shown in Figures 7.5 (b) and (e) respectively. A significant improvement of 92.22% in the translational and 90.86% in rotational components of the pose over the previous state-of-the-art NNnet model can also be seen for the *Stairs* scene that contains repetitive structures as shown in Figure 7.5 (g). Furthermore, in addition to the comparisons with state-of-the-art models that we presented in this section, comprehensive comparisons with all the deep learning-based networks that have been benchmarked on the Microsoft 7-Scenes dataset thus far are presented in Table 7.5.

Table 7.2 shows the results on the outdoor DeepLoc dataset for which we obtain almost half the localization error as the previous methods. Our proposed VLocNet$_{\mathrm{STL}}$ outperforms the existing deep learning-based approaches by achieving a median localization error of 0.68 m in translational and 3.43° in rotational components of the pose despite the challenging scenarios in this dataset that include transparent and reflective glass structures and substantial lighting changes. Furthermore, our VLocNet++$_{\mathrm{STL}}$ achieves an improved median localization error of 0.37 m and 1.93° in translational and rotational components of the pose respectively. This amounts to an improvement of 77.02% in the translation and 45.17% in rotation respectively over the previous state-of-the-art. This demonstrates that our models perform equally well in outdoor environments where there is a significant amount of perceptual aliasing and lighting changes as well as in indoor environments with textureless and reflective surfaces.

#### 7.4.3.2 Evaluation of Visual Odometry

We evaluate the performance of our single-task VlocNet$_{\mathrm{STL}}$ and VLocNet++$_{\mathrm{STL}}$ for the task of 6-DoF visual odometry estimation using the average translational and rotational error relative to the sequence length on each of the datasets. Table 7.3 shows the quantitative results on the Microsoft 7-Scenes dataset. Our proposed VLocNet$_{\mathrm{STL}}$ model achieves an average error of 1.51% in translation and 1.45deg/m in orientation components which

**Table 7.3:** Average translational and rotational error for the task of 6DoF visual odometry on the Microsoft 7-Scenes dataset [%, deg/m] [34].

| Scene | LBO [288] | DeepVO [289] | cnnBspp [290] | VLocNet$_{\text{STL}}$ (Ours) | VLocNet++$_{\text{STL}}$ (Ours) |
|---|---|---|---|---|---|
| Chess | 1.69, 1.13 | 2.10, 1.15 | 1.38, 1.12 | 1.14, 0.75 | **0.99**, **0.66** |
| Fire | 3.56, 1.42 | 5.08, 1.56 | 2.08, 1.76 | 1.81, 1.92 | **0.99**, **0.78** |
| Heads | 14.43, 2.39 | 13.91, 2.44 | 3.89, 2.70 | 1.82, 2.28 | **0.58**, **1.59** |
| Office | 3.12, 1.92 | 4.49, 1.74 | 1.98, 1.52 | 1.71, 1.09 | **1.32**, **1.01** |
| Pumpkin | 3.12, 1.60 | 3.91, 1.61 | 1.29, 1.62 | 1.26, 1.11 | **1.16**, **0.98** |
| RedKitchen | 3.71, 1.47 | 3.98, 1.50 | 1.53, 1.62 | 1.46, **1.28** | **1.26**, 1.52 |
| Stairs | 3.64, 2.62 | 5.99, 1.66 | 2.34, 1.86 | **1.28**, 1.17 | 1.55, **1.10** |
| Average | 4.75, 1.79 | 5.64, 1.67 | 2.07, 1.74 | 1.51, 1.45 | **1.12**, **1.09** |

**Table 7.4:** Average translational and rotational error for the task of 6DoF visual odometry on the DeepLoc dataset [%, deg/m] [34].

| LBO [288] | DeepVO [289] | cnnBspp [290] | VLocNet$_{\text{STL}}$ (Ours) | VLocNet++$_{\text{STL}}$ (Ours) |
|---|---|---|---|---|
| 0.41, 0.053 | 0.33, 0.052 | 0.35, 0.049 | 0.15, 0.040 | **0.12**, **0.024** |

amounts to a reduction in the error by 0.56% and 16.67% respectively, over the state-of-the-art cnnBspp [290] model. Furthermore, our proposed VLocNet++$_{\text{STL}}$ model outperforms existing approaches as well as the VLocNet$_{\text{STL}}$ by achieving a translational error of 1.12% and rotational error of 1.09deg/m. The largest improvement over VLocNet$_{\text{STL}}$ and other end-to-end networks can be observed for the *Fire* and *Heads* scenes that have substantial textureless regions and motion-blur. Nevertheless, the high representational ability of the learned features using our dual-stream Siamese architecture enable our models to achieve state-of-the-art performance even in these challenging scenarios.

Table 7.4 shows the average translational and rotational error as a function of sequence length on the outdoor DeepLoc dataset. Accurately estimating the ego-motion outdoors is a rather challenging task due to the more apparent motion parallax and dynamic lighting changes that could occur between the consecutive input frames due to factors such as glare from the sun and shadows. Despite these challenges, both our VLocNet$_{\text{STL}}$ and VLocNet++$_{\text{STL}}$ outperform the other end-to-end networks and achieve state-of-the-art performance. Our proposed VLocNet$_{\text{STL}}$ achieves an average translation error of 0.15% and a rotational error of 0.040deg/m, while our VLocNet++$_{\text{STL}}$ further improves upon the performance yielding a translational error of 0.12% and a rotational error of 0.024 deg/m, thereby demonstrating the efficacy of our model for estimating the ego-motion in challenging outdoor environments.

### 7.4.4 Evaluation of Multitask Learning in VLocNet++

In this section, we present comprehensive empirical evaluations of our proposed multitask VLocNet$_{\text{MTL}}$ and VLocNet++$_{\text{MTL}}$ for each of the visual localization, odometry estimation and semantic segmentation tasks. We compare the performance for visual localization with both state-of-the-art local-feature based techniques as well as deep learning-based models. As no semantic segmentation labels are available in the Microsoft 7-Scenes dataset, we only train the multitask model for visual localization and odometry estimation, while we train the models for all the three tasks on the DeepLoc dataset.

#### 7.4.4.1 Evaluation of Visual Localization

We benchmark the performance of our multitask VLocNet$_{\text{MTL}}$ and VLocNet++$_{\text{MTL}}$ models on the Microsoft 7-Scenes and the DeepLoc datasets by comparing against local feature-based pipelines, learning-based techniques as well as our single-task VLocNet$_{\text{STL}}$ and VLocNet++$_{\text{STL}}$ models. Following the standard benchmarking metrics, we report the median localization error and present our main results on the Microsoft 7-Scenes dataset in comparison to the state-of-the-art local feature-based techniques in Table 7.6 and in comparison to other deep learning-based networks in Table 7.5. It can be seen that the proposed VLocNet$_{\text{MTL}}$ that achieves a median localization error of 0.04 m and 3.09° is the first deep learning-based model to perform on a par with state-of-the-art local feature-based pipelines. Utilizing our Geometric Consistency loss function and jointly regressing the relative motion in addition to the global pose enables our network to effectively exploit motion-specific information thereby yielding accurate pose estimates. Moreover, we observe that VLocNet++$_{\text{MTL}}$ achieves a median localization error of 0.013 m and 0.77° which amounts to a significant improvement of 67.50% in the translational and 75.08% in the rotational components of the pose over VLocNet$_{\text{MTL}}$. This can be attributed to the fusion of feature maps from the previous timestep using our proposed weighted fusion layer that enables the Geometric Consistency loss function to effectively incorporate motion-specific temporal features. This enables our network to achieve sub-centimeter and sub-degree accuracy for the majority of the scenes. Furthermore, unlike local feature-based approaches, our proposed VLocNet++ is able to accurately estimate the global pose in environments containing repetitive structures such as the *Stairs* scene and the scenes containing textureless or reflective surfaces such as *Fire*, *Pumpkin* and *RedKitchen*.

Figure 7.6 further shows the median localization error and the percentage of poses for which the error is below 5 cm and 5° in comparison to the state-of-the-art on the Microsoft 7-Scenes benchmark. While our proposed VLocNet$_{\text{MTL}}$ is the first deep learning-based approach to yield an accuracy comparable to local feature-based pipelines achieving higher performance than SCoRe Forests [284] in terms of number of images with pose error below 5 cm and 5°, its performance is still lower than DSAC2 [293] that was proposed by Brachmann *et al.* which is also the current state-of-the-art. However, our proposed

**Table 7.5:** Benchmarking the median localization error of our single-task models on the Microsoft 7-Scenes dataset [34].

| Scene | PoseNet [276] | Bayesian PoseNet [287] | LSTM-Pose [277] | Hourglass-Pose [291] | BranchNet [283] | PoseNet2 [282] | NNnet [286] | VLocNet$_{STL}$ (Ours) | VLocNet++$_{STL}$ (Ours) |
|---|---|---|---|---|---|---|---|---|---|
| Chess | 0.32m, 8.12° | 0.37m, 7.24° | 0.24m, 5.77° | 0.15m, 6.53° | 0.18m, 5.17° | 0.13m, 4.48° | 0.13m, 6.46° | 0.036m, 1.71° | **0.023**m, **1.44**° |
| Fire | 0.47m, 14.4° | 0.43m, 13.7° | 0.34m, 11.9° | 0.27m, 10.84° | 0.34m, 8.99° | 0.27m, 11.3° | 0.26m, 12.72° | 0.039m, 5.34° | **0.018**m, **1.39**° |
| Heads | 0.29m, 12.0° | 0.31m, 12.0° | 0.21m, 13.7° | 0.19m, 11.63° | 0.20m, 14.15° | 0.17m, 13.0° | 0.14m, 12.34° | 0.046m, 6.64° | **0.016**m, **0.99**° |
| Office | 0.48m, 7.68° | 0.48m, 8.04° | 0.30m, 8.08° | 0.21m, 8.48° | 0.30m, 7.05° | 0.19m, 5.55° | 0.21m, 7.35° | 0.039m, 1.95° | **0.024**m, **1.14**° |
| Pumpkin | 0.47m, 8.42° | 0.61m, 7.08° | 0.33m, 7.00° | 0.25m, 7.01° | 0.27m, 5.10° | 0.26m, 4.75° | 0.24m, 6.35° | 0.037m, 2.28° | **0.024**m, **1.45**° |
| RedKitchen | 0.59m, 8.64° | 0.58m, 7.54° | 0.37m, 8.83° | 0.27m, 10.15° | 0.33m, 7.40° | 0.23m, 5.35° | 0.24m, 8.03° | 0.039m, 2.20° | **0.025**m, **2.27**° |
| Stairs | 0.47m, 13.8° | 0.48m, 13.1° | 0.40m, 13.7° | 0.29m, 12.46° | 0.38m, 10.26° | 0.35m, 12.4° | 0.27m, 11.82° | 0.097m, 6.48° | **0.021**m, **1.08**° |
| Average | 0.44m, 10.4° | 0.47m, 9.81° | 0.31m, 9.85° | 0.23m, 9.53° | 0.29m, 8.30° | 0.23m, 8.12° | 0.21m, 9.30° | 0.048m, 3.80° | **0.022**m, **1.39**° |

**Table 7.6:** Benchmarking the median localization error of our multitask models on the Microsoft 7-Scenes dataset [34].

| Scene | Active Search [274] | SCoRe Forest [284] | DSAC [292] | DSAC2 [293] w/o 3D | DSAC2 [293] w/ 3D | VLocNet$_{STL}$ (Ours) | VLocNet++$_{STL}$ (Ours) | VLocNet$_{MTL}$ (Ours) | VLocNet++$_{MTL}$ (Ours) |
|---|---|---|---|---|---|---|---|---|---|
| Chess | 0.04m, 1.96° | 0.03m, **0.66**° | 0.02m, 1.20° | 0.02m, 0.70° | 0.02m, 0.50° | 0.03m, 1.69° | 0.023m, 1.44° | 0.03m, 1.71° | **0.018**m, 1.17° |
| Fire | 0.03m, 1.53° | 0.05m, 1.50° | 0.04m, 1.50° | 0.04m, 1.20° | 0.02m, 0.80° | 0.04m, 5.34° | 0.018m, 1.39° | 0.04m, 4.86° | **0.009**m, **0.61**° |
| Heads | 0.02m, 1.45° | 0.06m, 5.50° | 0.03m, 2.70° | 0.24m, 10.00° | 0.01m, 0.80° | 0.04m, 6.64° | 0.016m, 0.99° | 0.05m, 4.99° | **0.008**m, **0.60**° |
| Office | 0.09m, 3.61° | 0.04m, 0.78° | 0.04m, 1.60° | 0.03m, 0.80° | 0.03m, **0.70**° | 0.04m, 1.95° | 0.024m, 1.14° | 0.03m, 1.51° | **0.016**m, 0.78° |
| Pumpkin | 0.08m, 3.10° | 0.04m, **0.68**° | 0.05m, 2.00° | 0.04m, 1.10° | 0.04m, 1.00° | 0.04m, 2.28° | 0.024m, 1.45° | 0.04m, 1.92° | **0.009**m, 0.82° |
| RedKitchen | 0.07m, 3.37° | 0.04m, **0.76**° | 0.05m, 2.00° | 0.05m, 1.30° | 0.04m, 1.00° | 0.04m, 2.20° | 0.025m, 2.27° | 0.03m, 1.72° | **0.017**m, 0.93° |
| Stairs | 0.03m, 2.22° | 0.32m, 1.32° | 1.17m, 33.1° | 0.27m, 5.40° | 0.10m, 2.50° | 0.10m, 6.48° | 0.021m, 1.08° | 0.07m, 4.96° | **0.010**m, **0.48**° |
| Average | 0.05m, 2.46° | 0.08m, 1.60° | 0.20m, 6.30° | 0.099m, 2.92° | 0.04m, 1.04° | 0.048m, 3.80° | 0.022m, 1.39° | 0.04m, 3.09° | **0.013**m, **0.77**° |

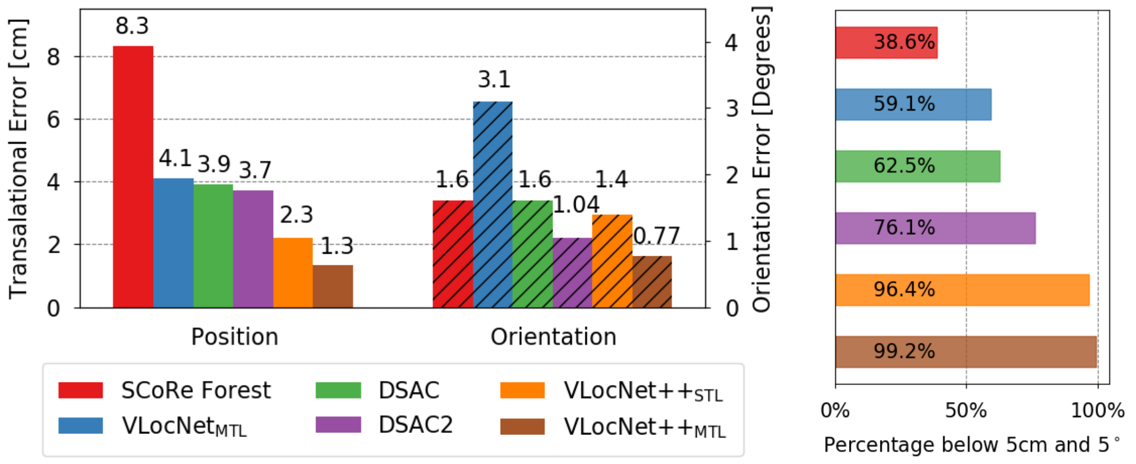


**Figure 7.6:** Benchmarking 6DoF localization on the Microsoft 7-Scenes dataset [34]. We compare against state-of-the-art approaches that utilize RGB or RGB-D data and even with approaches that depend on a 3D model, VLocNet++ only uses RGB images. We report the median localization errors (left) and percentage of test images with a pose error below 5 cm and 5° (right).

**Table 7.7:** Summary of benchmarking the median localization error of VLocNet++$_{MTL}$ in comparison with the state-of-the-art on Microsoft 7-Scenes dataset [34].

| Method | Input | Median Trans Error | Median Rot Error | Pose Acc | Run-time |
|---|---|---|---|---|---|
| DSAC2 [293] | w/ 3D | 0.04 m | 1.04° | 76.1% | 200 ms |
| VLocNet++$_{MTL}$ (Ours) | Monocular | **0.013** m | **0.77**° | **99.2%** | 79 ms |

VLocNet++$_{STL}$ model achieves a localization accuracy of 96.4%, improving over the accuracy of DSAC2 [293] by 20.3% and by over an order of magnitude compared to the other deep learning approaches [282, 286]. Moreover, by employing our proposed multitask framework, VLocNet++$_{MTL}$ further improves on the performance and achieves an accuracy of 99.2%, thereby setting the new state-of-the-art on this benchmark. It is important to note that other than our proposed VLocNet and VLocNet++, the competing methods shown in Figure 7.6 rely on a 3D scene model and hence require RGB-D data, whereas our model only utilizes monocular images. DSAC [292] and its variant DSAC2 [293] that utilize only RGB images demonstrate a lower performance as shown in Table 7.6. The improvement achieved by VLocNet++ demonstrates that the apt combination of employing the Geometric Consistency loss and the weighted fusion layer enables the network to efficiently leverage the motion-specific features in order to learn a geometrically consistent model.

Furthermore, Table 7.7 summarizes the comparison between our VLocNet++$_{MTL}$ and DSAC2 [293] in terms of the median localization error and the percentage of the poses

**Table 7.8:** Benchmarking the median localization error of our multitask models on the DeepLoc dataset [34].

| PoseNet [276] | Bay. PoseNet [287] | SVS-Pose [285] | VLocNet$_{\text{STL}}$ (Ours) | VLocNet++$_{\text{STL}}$ (Ours) | VLocNet$_{\text{MTL}}$ (Ours) | VLocNet++$_{\text{MTL}}$ (Ours) |
|---|---|---|---|---|---|---|
| 2.42m, 3.66° | 2.24m, 4.31° | 1.61m, 3.52° | 0.68m, 3.43° | 0.37m, 1.93° | 0.47m, 2.38° | **0.32**m, **1.48**° |

with localization error below 5 cm and 5° denoted by pose accuracy. It can be seen that our proposed model exceeds the state-of-the-art by 67.5% in the translational and 25.9% in the rotational components of the pose. Unlike DSAC2, our proposed approach does not require a 3D model of the scene which facilitates ease of deployment, in addition to occupying less storage space for the model. While achieving accurate pose estimates is crucial for any localization approach, the run-time requirements and complexity of deploying the models play an important role in its ease of deployment in the real-world on various robotic systems. Our proposed VLocNet++$_{\text{MTL}}$ model only requires 79 ms for a forward-pass on a single consumer grade GPU versus the 200 ms consumed by DSAC2 which makes VLocNet++ 60.5% faster. This renders our method well suited for real-time deployment in an online manner, as well as in resource constrained platforms.

Table 7.8 shows the performance of our multitask VLocNet$_{\text{MTL}}$ and VLocNet++$_{\text{MTL}}$ on the DeepLoc dataset. Although our proposed single-task models outperform the the other deep learning networks by a large margin, our multitask VLocNet$_{\text{MTL}}$ further improves upon the performance and achieves a median localization error of 0.47 m and 2.38° which amounts to an improvement of 70.81% in translational and 32.39% in rotational components of the pose while compared to the best performing SVS-Pose [285] network. Even though perceptual aliasing is more apparent outdoors than indoors, this large improvement demonstrates that our Geometric Consistency loss function is as effective at aggregating motion-specific information outdoors as indoors. Furthermore, our VLocNet++$_{\text{MTL}}$ model that simultaneously encodes geometric and semantic knowledge of the environment into the pose regression network achieves a median localization error of 0.32 m and 1.48° thereby setting the new state-of-the-art on this benchmark. This amounts to an improvement of 31.91% and 37.81% in translation and rotation respectively, over VLocNet$_{\text{MTL}}$ which signifies that our proposed weighted fusion layer enables our network to aggregate motion-specific temporal information as well as effectively encode semantic constraints to learn a geometric and structure-aware pose regression model.

#### 7.4.4.2 Evaluation of Visual Odometry

In this section, we evaluate the performance of our multitask models for the task of visual odometry estimation and report the average translation and orientation error as a function

**Table 7.9:** Average translational and rotational error of our multitask models for the task of 6DoF visual odometry on the Microsoft 7-Scenes dataset [%, deg/m] [34].

| Scene | LBO [288] | DeepVO [289] | cnnBspp [290] | VLocNet$_{\text{STL}}$ (Ours) | VLocNet++$_{\text{STL}}$ (Ours) | VLocNet$_{\text{MTL}}$ (Ours) | VLocNet++$_{\text{MTL}}$ (Ours) |
|---|---|---|---|---|---|---|---|
| Chess | 1.69, 1.13 | 2.10, 1.15 | 1.38, 1.12 | 1.14, 0.75 | 0.99, 0.66 | 1.09, 0.73 | **0.97**, **0.63** |
| Fire | 3.56, 1.42 | 5.08, 1.56 | 2.08, 1.76 | 1.81, 1.92 | 0.99, 0.78 | 1.77, 1.89 | **0.98**, **0.73** |
| Heads | 14.43, 2.39 | 13.91, 2.44 | 3.89, 2.70 | 1.82, 2.28 | 0.58, 1.59 | 1.75, 2.01 | **0.54**, **1.50** |
| Office | 3.12, 1.92 | 4.49, 1.74 | 1.98, 1.52 | 1.71, 1.09 | 1.32, 1.01 | 1.68, 0.99 | **1.31**, **0.97** |
| Pumpkin | 3.12, 1.60 | 3.91, 1.61 | 1.29, 1.62 | 1.26, 1.11 | 1.16, 0.98 | 1.25, 1.08 | **1.11**, **0.94** |
| RedKitchen | 3.71, 1.47 | 3.98, 1.50 | 1.53, 1.62 | 1.46, **1.28** | 1.26, 1.52 | 1.43, 1.29 | **1.24**, 1.40 |
| Stairs | 3.64, 2.62 | 5.99, 1.66 | 2.34, 1.86 | 1.28, 1.17 | 1.55, 1.10 | **1.26**, 1.19 | 1.40, **1.05** |
| Average | 4.75, 1.79 | 5.64, 1.67 | 2.07, 1.74 | 1.51, 1.45 | 1.12, 1.09 | 1.46, 1.31 | **1.08**, **1.03** |

**Table 7.10:** Average translational and rotational error of our multitask models for the task of 6DoF visual odometry on the DeepLoc dataset [%, deg/m] [34].

| LBO [288] | DeepVO [289] | cnnBspp [290] | VLocNet$_{\text{STL}}$ (Ours) | VLocNet++$_{\text{STL}}$ (Ours) | VLocNet$_{\text{MTL}}$ (Ours) | VLocNet++$_{\text{MTL}}$ (Ours) |
|---|---|---|---|---|---|---|
| 0.41, 0.053 | 0.33, 0.052 | 0.35, 0.049 | 0.15, 0.040 | 0.12, 0.024 | 0.15, 0.039 | **0.10**, **0.020** |

of the trajectory length for each of the scenes in the datasets. Table 7.9 shows the results on the Microsoft 7-Scenes benchmark where our proposed VLocNet$_{\text{MTL}}$ outperforms all the other deep learning-based approaches and achieves an average error of 1.46% and 1.31deg/m in translation and orientation components respectively. This demonstrates that by jointly learning to regress the relative motion along with the global pose benefits both tasks and enables the joint model to outperform their single-task counterparts. Additionally, our VLocNet++$_{\text{MTL}}$ further improves upon the performance and achieves an average error of 1.08% in translation and 1.03deg/m in orientation, thereby outperforming each of the single-task models and the other deep learning-based methods, while setting the state-of-the-art on this dataset. Our model is able to better estimate the relative motion by sharing representations with the global pose regression stream which enables it to learn temporally invariant features which can be used to accurately estimate the relative motion between the input frames, while being robust to motion blur, textureless regions and repetitive structures in the scene.

We observe a similar trend from the experimental comparisons on the outdoor DeepLoc dataset shown in Table 7.10, where both our multitask models outperform their single-task counterparts as well the other deep learning-based networks. Our VLocNet++$_{\text{MTL}}$ achieves state-of-the-art performance with an average error of 0.10% and 0.020deg/m for the translation and orientation components, which is an improvement of 0.25% and 59.18% respectively, over the best performing cnnBspp [290] network. An improvement of this scale is especially hard to obtain in outdoor environments due to several factors such as areas with very similar appearance as shown in Figure 7.4 (a), glass structures that cause reflections as shown in Figure 7.4 (c) and variable lighting conditions between the consecutive input images. Despite these factors our proposed models yield reliable performance in challenging outdoor environments demonstrating the efficacy of our multitask leaning scheme.

#### 7.4.4.3 Evaluation of Semantic Segmentation

We present comprehensive empirical evaluations for the task of semantic segmentation using our proposed VLocNet++$_{\text{MTL}}$ model that is built upon the topology of our AdapNet++ architecture as well as the VLocNet++(base) variant that is built upon the topology of our AdapNet architecture. We present results on the DeepLoc dataset and report the Intersection over Union (IoU) score for each of the individual object categories as well as the mean IoU. We benchmark the performance of our model against several state-of-the-art networks including FCN-8s [24], SegNet [136], UpNet [48], ParseNet [168], DeepLab v2 [138] and DeepLab v3 [132]. We also compare with the performance of AdapNet and AdapNet++ that we presented in Chapter 4. Results from this experiment are shown in Table 7.11.

Our proposed VLocNet++(base) model achieves a mIoU score of 80.44%, thereby outperforming the other baseline models in the overall mIoU score as well as consistently

**Table 7.11:** Comparison of the semantic segmentation performance of our multitask models against state-of-the-art networks on the DeepLoc dataset. Our model VLocNet++(base) has a topology built upon AdapNet, while our model VLocNet++$_{\text{MTL}}$ has a topology built upon AdapNet++.

| Approach | Sky | Road | Sidew | Grass | Veg | Bldng | Poles | Dyn | Other | Mean |
|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟪 | 🟪 | 🟩 | 🟩 | ⬛ | ◼ | 🟫 | 🟨 | IoU |
| FCN-8s [24] | 94.65 | 98.98 | 64.97 | 82.14 | 84.47 | 87.68 | 45.78 | 66.39 | 47.27 | 69.53 |
| SegNet [136] | 93.42 | 98.57 | 54.43 | 78.79 | 81.63 | 84.38 | 18.37 | 51.57 | 33.29 | 66.05 |
| UpNet [48] | 95.07 | 98.05 | 63.34 | 81.56 | 84.79 | 88.22 | 31.75 | 68.32 | 45.21 | 72.92 |
| ParseNet [168] | 92.85 | 98.94 | 62.87 | 81.61 | 82.74 | 86.28 | 27.35 | 65.44 | 45.12 | 71.47 |
| DeepLab v2 [138] | 93.39 | 98.66 | 76.81 | 84.64 | 88.54 | 93.07 | 20.72 | 66.84 | 52.70 | 67.54 |
| DeepLab v3 [132] | 93.51 | 98.80 | 77.63 | 85.78 | 88.62 | 93.56 | 24.66 | 67.75 | 53.86 | 76.02 |
| AdapNet | 94.65 | 98.98 | 64.97 | 82.14 | 84.48 | 87.68 | 45.78 | 66.40 | 47.27 | 78.59 |
| AdapNet++ | 96.38 | 99.09 | 80.99 | 89.91 | 92.25 | 95.03 | 48.80 | 73.11 | 61.72 | 81.92 |
| VLocNet++(base) | 95.84 | 98.99 | 80.85 | 88.15 | 91.28 | 94.72 | 45.79 | 69.83 | 58.59 | 80.44 |
| VLocNet++$_{\text{MTL}}$ | **96.77** | **99.19** | **83.65** | **90.38** | **92.92** | **95.10** | **48.93** | **75.76** | **64.02** | **82.96** |

for each of the individual class-wise IoU scores. However, VLocNet++(base) does not exceed the performance of our AdapNet++ architecture. As we employ the topology of our AdapNet architecture for the single-task topology of VLocNet++(base), the improvement of 1.85% observed in comparison to AdapNet. This can be attributed to the self-supervised representational warping scheme that we introduced as well as the inductive transfer that occurs from the training signals of the localization network due to jointly training the models in a MTL framework. Furthermore, our VLocNet++$_{\text{MTL}}$ model outperforms all the other architectures as well as its single-task counterpart and sets the new state-of-the-art on this benchmark by achieving a mIoU score of 82.96%. In addition, the self-supervised warping scheme enables our VLocNet++$_{\text{MTL}}$ model to converge in $26,000$ iterations, whereas the single-task model requires $120,000$ iterations to converge.

Inspecting the individual object class IoU scores of our VLocNet++(base) model in comparison to AdapNet, we observe the largest improvement of 15.88% for the *sidewalk* class, followed by an improvement of 11.32% for the *other* object class. The *other* object class is one of the hardest to accurately segment in our dataset due to the diversity of object contained in that category. A notable improvement of 7.06%, 6.8% and 6.01% can also be observed for the *building, vegetation*, and *grass* categories. Furthermore, the largest improvement in the performance of VLocNet++$_{\text{MTL}}$ over VLocNet++(base) was observed for the *dynamic* object class where VLocNet++$_{\text{MTL}}$ achieves an improvement of 5.93%, followed by an improvement of 5.43% for the *other* object class. The improvement in object categories such as *sidewalk, building, vegetation*, and *grass* can be attributed to our proposed self-supervised warping scheme that effectuates temporal coherence by warping and fusing representations from the previous timestep into the current view. While the

**Table 7.12:** Comparison of the median localization error of different VLocNet++ base architecture topologies on the DeepLoc dataset [34].

| Method | Base Model | Activation | Median Error |
|---|---|---|---|
| M1 | ResNet-18 | ReLU | 0.83 m, 5.96° |
| M2 | ResNet-34 | ReLU | 0.57 m, 4.04° |
| M3 | ResNet-50 | ReLU | 0.65 m, 2.87° |
| M4 | PA ResNet-50 | ReLU | 0.57 m, 2.44° |

improvement for the *other* object class can be primarily attributed to the fusion of learned representations from the localization stream using our weighted fusion layer that facilitates encoding location-specific information. This enables our network to learn a more accurate disambiguation between the diverse set of objects that are categorized under the *other* object class. The improvement in these object classes can also be seen in the extensive qualitative semantic segmentation results presented in Section 7.4.8.2.

## 7.4.5 Ablation Study

In this section, we present comprehensive ablation studies investigating the various architectural design choices that we made in our proposed VLocNet and VLocNet++ architectures while rationalizing the decisions with empirical results. We first describe the base architecture topology of our VLocNet model in Section 7.4.5.1 and subsequently investigate the influence of the previous pose fusion in Section 7.4.5.2.

We then present experiments that describe our multitask learning design choices such as analyzing the effect of hard parameter sharing in Section 7.4.5.3, influence of warping and fusing the semantic features from the previous timestep at different stages of the network in Section 7.4.5.4, influence of fusing semantic features into the localization stream at various stages in Section 7.4.5.5, analyzing the influence of our proposed weighted fusion layer to aggregate motion-specific features and encode semantic information into the localization stream in Section 7.4.5.6, influence of initializing our multitask model with different pre-trained weights in Section 7.4.5.7 and finally, evaluation of the various optimization strategies in Section 7.4.5.8.

### 7.4.5.1 Detailed Study on the Base Architecture Topology

In Table 7.12, we present the localization performance on the DeepLoc dataset for the different variants of the base architecture that we experiment with. Models M1 to M3 employ shallow to deeper residual architectures with the standard ReLU activation function. The M3 model consisting of the ResNet-50 architecture [27] as the backbone, demonstrates an improved performance compared to the shallower ResNet-34 and ResNet-18 architectures.

**Table 7.13:** Comparative analysis of different loss functions and loss weightings in the VLocNet++ architecture in terms of the median localization performance on the DeepLoc dataset [34].

| Model | Activation | Loss Function | Loss Weighting | Median Error |
|---|---|---|---|---|
| PoseNet [276] | ReLU | $\mathcal{L}_{euc}$ | $\beta$ | 0.44 m, 10.40° |
| M4 | ReLU | $\mathcal{L}_{euc}$ | $\beta$ | 0.57 m, 2.44° |
| M41 | ELU | $\mathcal{L}_{euc}$ | $\beta$ | 1.71 m, 2.14° |
| M42 | ELU | $\mathcal{L}_{geo}$ | $\beta$ | 0.56 m, 2.06° |
| M43 (VLocNet++$_{\text{STL}}$) | ELU | $\mathcal{L}_{geo}$ | $\hat{s}_x$, $\hat{s}_q$ | **0.37** m, **1.93**° |

Moreover, the full pre-activation ResNet-50 architecture [72] further increases the performance as it reduces overfitting and improves the convergence of the network. Therefore, we employ the M4 model as our base architecture in the remainder of this chapter.

Using the M4 architecture as the network backbone, we investigate the influence of different activation functions, loss functions and loss weightings on the localization performance. More specifically, we quantitatively analyze the performance improvements for the following architectural configurations:

- M4: Full pre-activation ResNet-50 base architecture with ReLUs, Euclidean loss for translation and rotation with loss weighting $\beta = 1$
- M41: Full pre-activation ResNet-50 base architecture with ELUs, Euclidean loss for translation and rotation with loss weighting $\beta = 1$
- M42: Full pre-activation ResNet-50 base architecture with ELUs and previous pose fusion using $\mathcal{L}_{geo}$ loss with loss weighting $\beta = 1$
- M43: Full pre-activation ResNet-50 base architecture with ELUs and previous pose fusion using $\mathcal{L}_{geo}$ loss with learnable loss weightings $\hat{s}_x$, $\hat{s}_q$. This model corresponds to our proposed single-task VLocNet++$_{\text{STL}}$ architecture.

Table 7.13 shows the median localization error of the aforementioned models on the DeepLoc dataset. We also show the performance of the PoseNet [276] model for reference as it was the one of the first end-to-end pose regression methods that was proposed. Our base M4 model outperforms PoseNet significantly in the rotational component of the pose by 76.54%, however it trails behind PoseNet by 22.81% in the translational component. We replacing the ReLU activation function with ELU in the M41 model as they are more robust to noisy data and further accelerate the training. However, this yields an improvement of 12% in the rotational component of the pose at the cost of a reduction in the accuracy of the translational component by factor of two. In the M42 model, we employ our Geometric Consistency loss function which reduces both the translational and orientation errors. Furthermore, utilizing learnable loss weightings $\hat{s}_x$, $\hat{s}_q$ in the M43 model (VLocNet++$_{\text{STL}}$), instead of the constant weighting $\beta$, yields a substantial improvement of 37.09% and 20.90% in the translational and rotational components over our base M1 model and outperforms the PoseNet model. This validates our hypothesis that constricting the search space with the relative pose information while training using our Geometric Consistency loss function coupled with learnable loss weighting parameters enables the network to learn a model that is more representative of the environment and thus improves the localization performance.

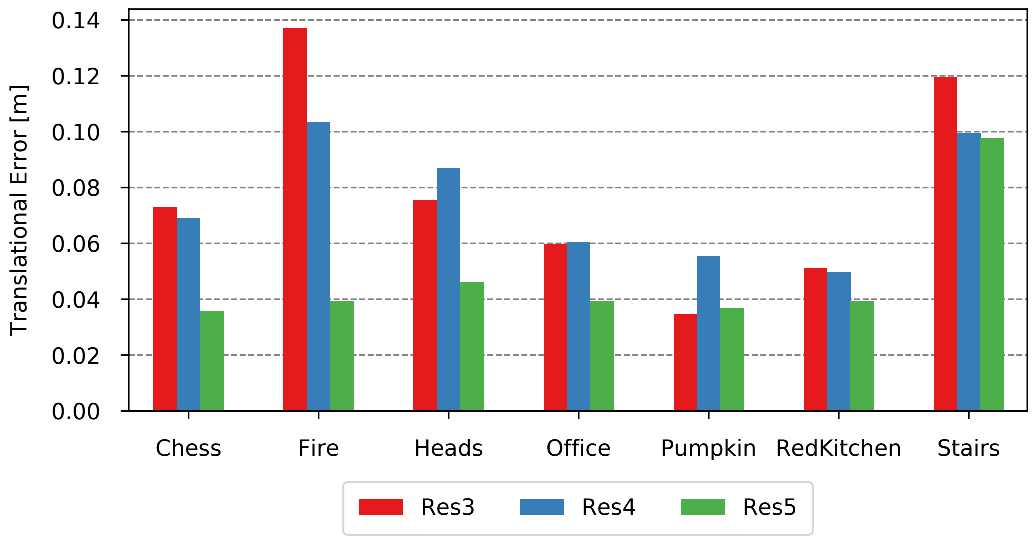
#### 7.4.5.2 Evaluation of Previous Pose Fusion

Fusing the previous predicted pose information is critical to enable the Geometric Consistency loss function to effectively constrict the search space by leveraging the relative motion while training. In order to identify the downsampling stage to fuse the previous predicted pose in the global pose regression stream, we evaluate the localization performance by fusing this information a various stages in our VLocNet architecture. Figure 7.7 shows the median localization error on the Microsoft 7-Scenes dataset while fusing the previous predicted pose at *Res3*, *Res4* and *Res5* in our architecture. The results demonstrate that fusing this information at earlier stages of the network results in an imbalance in the pose error by either reduction in the translational error at the cost of increasing the rotational error or vice versa. However, fusing the previous predicted pose at *Res5*, where the feature maps are of dimensions $7 \times 7$, yields the lowest localization error that is consistently lower than fusing at earlier stages. We hypothesize that this occurs due to fact that the features at the later stages are more mature and task-specific, which enables the network to easily exploit this previous pose prediction information while training.

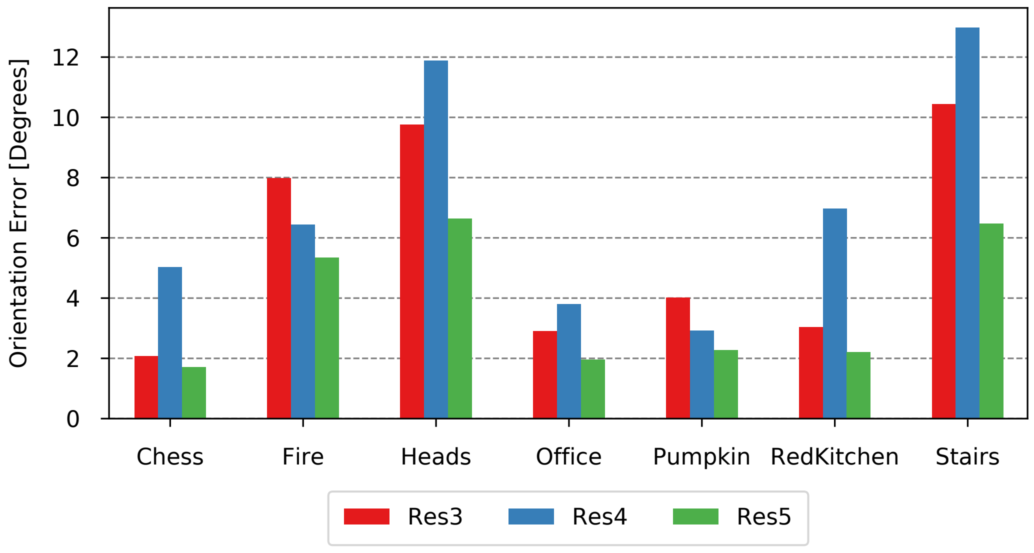
#### 7.4.5.3 Evaluation of Hard Parameter Sharing

Sharing features across the global pose regression and odometry streams can enable a competitive and collaborative action as each task-specific stream updates its own weights during backpropagation in an attempt to minimize the distance to the groundtruth. This symbiotic action introduces additional regularization while training, thereby alleviating overfitting. While sharing features across multiple networks can be inferred as a form of regularization, it is not clear apriori for how many layers should a shared stream be maintained. Sharing only a few initial layers does not have any additive benefit to either network, as early network layers often learn very generic feature representations. On the other hand, maintaining a shared stream too deep into the network can negatively impact the performance for the shared tasks, as the features learned towards the end are more task-specific.

In this section, we investigate the impact of sharing features across the global pose regression and visual odometry streams by experimenting with varying amounts of feature



(a) Translational Error



(b) Orientation Error

**Figure 7.7:** Comparison of median localization error on the Microsoft 7-Scenes dataset from fusing previous predicted pose information at various stages in VLocNet architecture [56]. The results consistently show that the highest localization accuracy is achieved by fusing the previous predicted pose in the *Res5* residual block of the network.

**Table 7.14:** Comparison of the localization performance achieved by VLocNet$_{\mathrm{MTL}}$ with varying amounts of weight sharing between the global pose regression and odometry streams. Results are shown on the Microsoft 7-Scenes dataset [56].

| Scene | Res2 | Res3 | Res4 |
|---|---|---|---|
| Chess | 0.04 m, **1.60**° | **0.03** m, 1.69° | 0.05 m, 1.76° |
| Fire | 0.05 m, **4.40**° | **0.04** m, 4.86° | 0.05 m, 4.59° |
| Heads | 0.05 m, **4.44**° | **0.05** m, 4.99° | 0.06 m, 5.99° |
| Office | 0.04 m, 1.68° | **0.03** m, **1.51**° | 0.04 m, 1.82° |
| Pumpkin | 0.05 m, 1.83° | **0.04** m, 1.92° | 0.04 m, **1.64**° |
| RedKitchen | 0.04 m, 1.89° | **0.03** m, **1.72**° | 0.04 m, 1.75° |
| Stairs | 0.10 m, 5.08° | **0.07** m, 4.96° | 0.09 m, **4.67**° |
| Average | 0.06 m, **2.99**° | **0.04** m, 3.09° | 0.05 m, 3.17° |

sharing. Table 7.14 shows the median localization error achieved by VLocNet$_{\mathrm{MTL}}$ by maintaining a shared stream up to the end of *Res2*, *Res3* or *Res4* blocks of the networks. The localization performance is shown for each of the scenes in the Microsoft 7-Scenes dataset. The results indicate that the lowest localization error is achieved by maintaining a shared stream up to the *Res3* block. This demonstrates that the representations learned after the *Res3* block are highly task-specific and maintaining a shared stream beyond *Res3* negatively impacts both tasks. However, maintaining a shared stream until the end of the *Res2* block also negatively impacts the performance, as the representations learned before the *Res2* block are too generic to provide any benefit to either task. Whereas, maintaining a shared stream until the end of the *Res3* block results in an improvement of 12.5% in translational and 18.49% in rotational components of the pose in comparison to the single-task VLocNet$_{\mathrm{STL}}$ model. We believe that these results further corroborate the utility of jointly learning visual localization and odometry estimation in a multitask learning framework.

#### 7.4.5.4 Where to Warp Semantic Features?

In this section, we quantify the improvement in semantic segmentation due to our proposed self-supervised warping scheme. We conducted experiments on the DeepLoc dataset to determine the network stage at which fusion of warped feature maps from previous timesteps is most beneficial. There are several aspects to this problem, primarily concerning the incorporation of the warping scheme at the beginning of a residual block or the end of the residual block, and secondly, to determine the different residual blocks to introduce the warping in. We hypothesize that warping at the end of the residual block will be more effective as the features are more discriminative than at the beginning where the features are just fused with that of the previous block that are of different dimensions. Moreover, as

**Table 7.15:** Improvement in the segmentation performance due to the fusion of warped feature maps from the previous timestep in the VLocNet++$_{\text{MTL}}$ model. The Warping Layer denotes the layer at which the feature maps from the previous timestep are warped and fused in the segmentation stream. The results are shown on the DeepLoc dataset [34].

| Warping Layer | mIoU (%) |
|---|---|
| No warping | 78.59 |
| Res-3a | 80.03 |
| Res-3d | 80.19 |
| Res-2c, Res-3d | 80.09 |
| Res-3d, Res-5c | 80.34 |
| Res-4f, Res-5c | **80.44** |
| Res-3d, Res-4f, Res-5c | 80.31 |

our encoder architecture has four downsampling stages, the warping can also be introduced in multiple stages. In this case, we hypothesize that warping at multiple downsampling stages will be more beneficial as it would enable the network to aggregate features of different object scales from the previous timestep, thereby implicitly enforcing multiscale temporal consistency.

Table 7.15 shows the mIoU score achieved by fusing warped feature maps from the previous timestep at different stages in the semantic segmentation stream of our VLocNet++$_{\text{MTL}}$ architecture. In order to first determine whether the warping scheme should be introduced at the beginning or the end of a residual block, we experimented with adding the self-supervised warping layer at the *Res3a* unit and in another model at the *Res3d* unit. The results shown in Table 7.15 corroborate our hypothesis that warping the feature maps the end of the residual unit is more beneficial, as warping at *Res-3d* yields an increased mIoU than at *Res-3a*. Subsequently, we experimented introducing the warping layers at the end of the residual blocks at multiple downsampling stages, namely, at (*Res2c*, *Res3d*), (*Res3d*, *Res5c*), (*Res4f*, *Res5c*) and at (*Res3d*, *Res4f*, *Res5c*). The results demonstrate that warping at multiple downsampling stages only marginally increases the mIoU score and introducing the warping at the later stages of the network where the features are semantically more meaningful yields a larger improvement in the mIoU score. From the results shown in Table 7.15, we observe that the model incorporating the warping at the *Res-4f* and *Res-5c* residual blocks achieves an improvement of 1.85% over the model with no warping, thereby demonstrating the efficacy of our proposed self-supervised warping scheme in aggregating scene-level context.

**Table 7.16:** Improvement in the localization performance of VLocNet++$_{\text{MTL}}$ due to the fusion of semantic feature maps. The fusion layer denotes where the semantic feature maps are fused into the localization stream. The results are shown on the DeepLoc dataset [34].

| Fusion Layer | Median Translational Error | Median Rotational Error |
|---|---|---|
| No fusion | 0.37 m | 1.93° |
| Res4a | 0.49 m | 3.10° |
| Res4c | **0.32** m | **1.48**° |
| Res4d | 0.54 m | 1.30° |
| Res4e | 0.46 m | 1.95° |
| Res4f | 0.61 m | 1.45° |

### 7.4.5.5 Where to Fuse Semantic Features?

In order to identify the network stage at which fusing semantic feature maps into the localization stream is most beneficial, we performed experiments fusing the semantic features at various residual units of the *Res4* block of the localization stream. Although the feature maps at the *Res4* and *Res5* blocks have the same dimension as the semantic feature maps, the *Res5* block has a substantially large amount of feature channels that would in turn outweigh the rich semantic features. We do not consider fusing semantic features at earlier stages of the network before the *Res4* block, as the representations learned in the early layers are overly generic and would not benefit from the fusion of high-level semantic features. Therefore, we only experiment with fusing the semantic features at different residual units of the *Res4* block of the localization stream.

Results from this experiment on the DeepLoc dataset are shown in Table 7.16. We show the performance in terms of the median localization error achieved by fusing semantic features into the localization stream at different network stages of the VLocNet++$_{\text{MTL}}$ model. The results demonstrate that the lowest median error of 0.32 m in translation and 1.48° in orientation components of the pose is obtained by fusing the semantic feature maps at the *Res4c* unit of the localization stream which enables our model to exploit the right balance between task-specificity and feature maturity. Note that we do not experiment with fusing semantic features at the *Res4b* unit of the localization stream as we fuse location-specific feature maps from this layer into the semantic segmentation stream, therefore adding this connection would result in a cyclic dependency between these two streams.

### 7.4.5.6 Influence of the Weighted Fusion Layer

In this section, we investigate the effectiveness of employing our proposed weighted fusion layer for encoding semantic information and aggregating motion-specific features in the

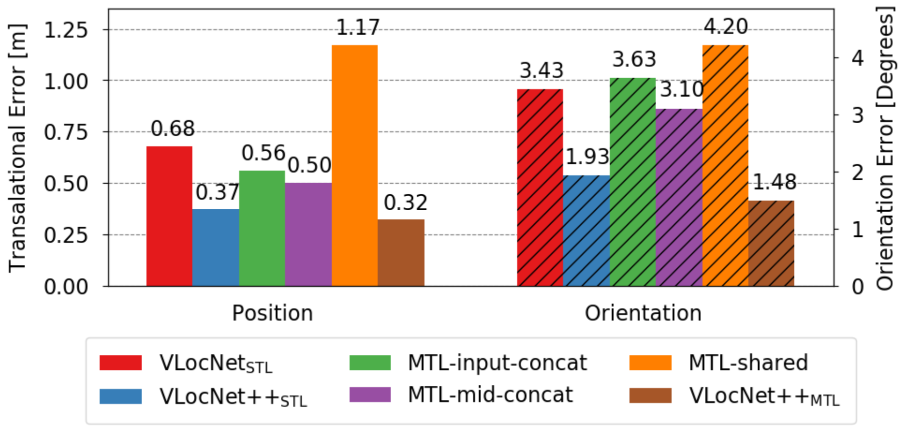


**Figure 7.8:** Comparison of the median localization error of VLocNet++$_{\text{MTL}}$ with baseline multitask models that fuse semantic features into the localization stream. Results are shown on the DeepLoc dataset [34].

global pose regression stream. We compare the localization accuracy of VLocNet++$_{\text{MTL}}$ which incorporates our fusion scheme against the single-task models as well as three competitive multitask baselines. The topologies of the three baselines are as follows:

- **MTL_input-concat**: A simple technique that can be employed for incorporating semantic features into the global pose regression stream is by concatenating the predicted semantic segmentation output $M_t$ with input image $I_t$ as a fourth-channel and feeding the resulting four-channel tensor as input to the localization stream.
- **MTL_mid-concat**: As a second baseline, we concatenate the semantic feature maps with intermediate representations of the global pose regression stream. As the experiments that we presented in Section 7.4.5.5 demonstrated that fusing semantic features with the *Res4c* unit is most beneficial, we similarly fuse semantic feature maps at *Res4c* using simple concatenation as opposed to employing our proposed weighted fusion layer.
- **MTL_shared**: Finally, we compare with an approach [294] that shares the latent spaces of both the semantic segmentation and global pose regression streams. Sharing the latent space can be realized by sharing the weights of a specific layer across different networks. In our implementation, we share the weights of the *Res4c* unit between both the global pose regression and semantic segmentation streams.

Figure 7.8 shows the results from this experiment on the DeepLoc dataset in terms of the median localization error metric. The results clearly show that employing the naive MTL-input-concat approach drastically reduces the performance over our single-task VLocNet++$_{\text{STL}}$ model. Furthermore, we observe that the MTL_mid-concat model achieves lower localization error than our single-task VLocNet++$_{\text{STL}}$ demonstrating that fusing

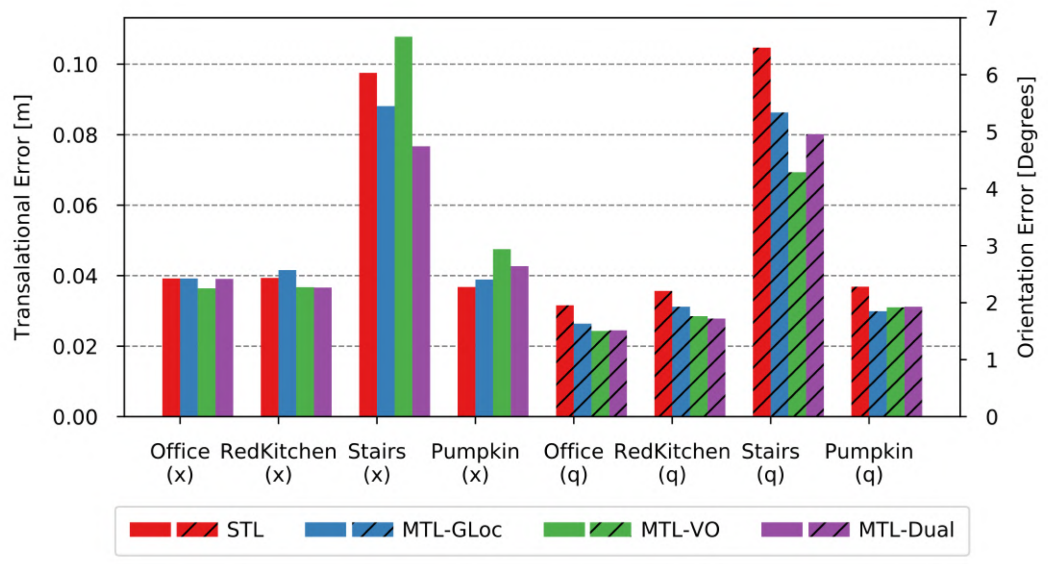


**Figure 7.9:** Localization performance of our single-task model in comparison to the multitask VLocNet$_{\text{MTL}}$ with different pre-trained weight initializations in the layers that are shared between the global pose regression stream and the odometry stream [56]. Results are shown on the Microsoft 7-Scenes dataset. (x) and (q) denote the translation and orientation components.

semantic features at intermediate network stages is more effective than at the input to the network directly. The final multitask baseline model MTL_shared achieves the lowest localization performance in comparison to the other multitask baselines as well as the single-task models. We hypothesize that this is primarily due to the diverse nature of the tasks at hand which causes the learned features to be significantly different and therefore, sharing weights across both network streams does not benefit the pose regression task.

Compared to the best performing MTL-input-concat model, our proposed VLocNet++$_{\text{MTL}}$ achieves an improvement of 36% in translational and 53.87% in rotational components of the pose, thereby outperforming all the baseline multitask models. While in comparison to our single-task VLocNet++$_{\text{STL}}$ model, VLocNet++$_{\text{MTL}}$ achieves an improvement of 13.51% and 23.32% in the translation and rotation components respectively, demonstrating the efficacy of our proposed weighted fusion scheme in learning the most optimal weightings for fusion based on region activations in the feature maps as well as the utility of fusing semantic features into the global pose regression stream.

#### 7.4.5.7 Evaluation of Initialization with Pre-trained Weights

In this section, we evaluate the localization performance of our multitask model with different pre-trained weight initializations in the layers that are shared between the global pose regression stream and the odometry stream. As each task-specific network stream is initially trained separately, it alters the weights of the convolution layers in a way that best minimizes the respective loss function. Combining the task-specific networks in a multitask learning framework to jointly learn multiple tasks requires an initialization strategy that enables effective feature sharing. Therefore, we evaluated various initialization strategies that facilitate the learning of inter-task correlations. Using the model trained on our single-task global pose regression stream $\text{VLocNet}_{\text{STL}}$ (STL) as a baseline, we evaluate the localization performance of the joint model with different pre-trained weight initializations. More precisely, we compare the performance of the following models:

- **MTL-GLoc**: Initializing the shared layers of the global pose regression stream with the weights from the pre-trained task-specific global pose regression model and the remaining layers with the Xavier initialization [108].

- **MTL-VO**: Initializing the shared layers of the global pose regression stream with the weights from the pre-trained task-specific visual odometry model and the remaining layers with the Xavier initialization [108].

- **MTL-Dual**: Utilizing the combined weights from each task-specific network to initialize the joint model.

Figure 7.9 shows the results from this experiment on the Microsoft 7-Scenes dataset in terms of the median localization error of our $\text{VLocNet}_{\text{MTL}}$ model employing the aforementioned initialization strategies. The results demonstrate our multitask model outperforms its single-task counterpart which further validates the efficacy of jointly learning global pose regression and visual odometry estimation tasks in a multitask learning framework. The improvement is clearly seen in the *Stairs* scene which is the most challenging scene to localize in the Microsoft 7-Scenes dataset as it contains repetitive structures and textureless surfaces.

On closer examination of the results, we find that the dual initialization of both network streams with weights from their task-specific models results in the best performance, contrary to initializing only one of the task-specific streams in the joint model and learning the other from scratch. This can be attributed to the training data that is insufficient to train large multitask networks without any pre-trained weight initialization. Furthermore, we observe that initializing only the global pose regression stream (MTL-GLoc) yields the least reduction in the localization error while compared to the single-task model. We hypothesize that this occurs due to the visual odometry stream needing to be trained from scratch and as a result, it does not provide reasonable relative pose estimates at the beginning of the training. Therefore, the localization stream cannot benefit from the motion-specific features from the odometry stream.

**Table 7.17:** Evaluation of different optimization strategies on the median localization performance of VLocNet++$_{\text{MTL}}$ trained on the Microsoft 7-Scenes dataset [34].

| Optimization Strategy | Median Translational Error | Median Rotational Error |
|---|---|---|
| Alternate | 0.042 m | 3.09° |
| Joint constant weights | 0.059 m | 3.79° |
| Joint learnable weights | **0.022** m | **1.39**° |

### 7.4.5.8 Evaluation of Optimization Strategies

In this section, we evaluate the localization performance of our VLocNet++$_{\text{MTL}}$ model using different optimization strategies. Note that we measure the overall utility of the optimization strategy in terms of the performance of the visual localization model as our main goal is to improve the global pose regression by learning auxiliary tasks. In this experiment, we explored using both joint and alternating optimization strategies to minimize the loss function. While employing the joint optimization strategy, we further experimented with using weights for balancing the scales between the different loss terms. More precisely, we compared the performance of utilizing constant equal weighting for each of the loss terms against employing learnable weightings.

Table 7.17 shows the results from this experiment on the Microsoft 7-Scenes dataset in terms of the median localization error of the MTL model employing different optimization strategies. The results demonstrate that using an alternating optimization strategy yields a localization error 28.99% and 18.47% lower in translation and rotation components respectively while compared to a joint optimization strategy with fixed equal weightings. This can be attributed to the difference in scales of the loss values for each task which in turn results in the optimizer becoming more biased towards minimizing the global pose regression error at the cost of having suboptimal relative pose estimates. However, this inadvertently results in worse accuracy for both tasks. Whereas, employing learnable weightings for the loss terms in the joint optimization strategy enables the model to achieve the lowest localization error with an improvement of 47.62% in the translational and 55.02% in the rotational components of the pose, over the alternating optimization strategy. Employing learnable loss weightings enables the network to maintain the individuality of each task during training while prohibiting the task with the largest scale from dominating the training process.

## 7.4.6 Visualization of the Level of Feature Similarity

Despite the recent surge in applying deep learning approaches to various domains, there is still a lack of fundamental knowledge regarding the representations learned by the deep net-

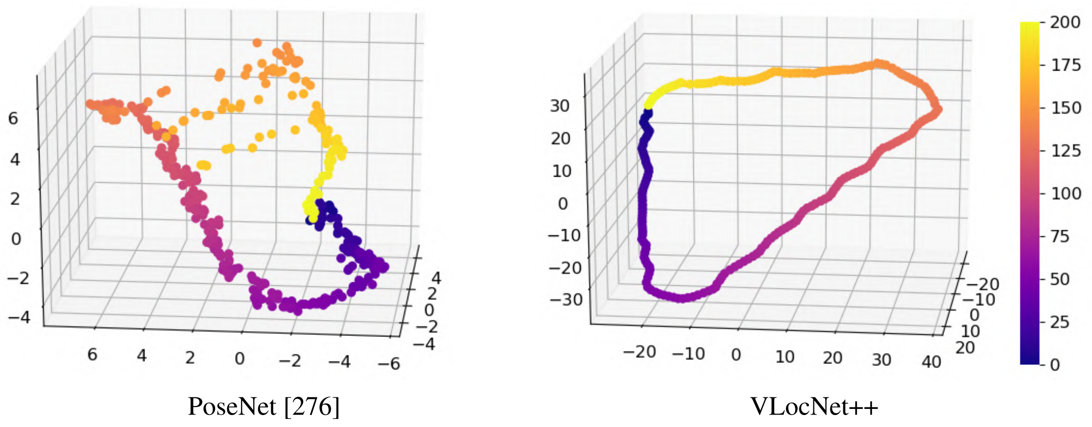


**Figure 7.10:** Comparison of the 3D multi-dimensional scaling (MDS) [295] of features from the penultimate layer of PoseNet [276] and VLocNet++$_{\mathrm{MTL}}$ trained on the DeepLoc dataset [34]. Inputs are images from the testing seq-01 loop and the points on the plot shown are chronologically colored. Features learned by our VLocNet++$_{\mathrm{MTL}}$ show precise correlation with the traversed trajectory (Figure 7.13 (b)), whereas PoseNet fails to capture the distribution especially for the poses near the glass buildings as shown in Figure 7.11 (a).

works. This can be attributed to the high dimensionality of the embedded representations. In order to aid in this understanding, feature visualization and dimensionality reduction techniques when thoughtfully applied can provide helpful insights. Such techniques transform the data from high dimensional spaces to ones of lower dimensions by decomposing the features along a set of principle axis. For the task of visual localization, preserving the global geometry of the features is highly critical over techniques that find clusters and sub-clusters in the data, such as those often applied for classification tasks. Therefore, in this section, we provide visualizations of the underlying distribution of the learned features from the penultimate layer of our multitask VLocNet++$_{\mathrm{MTL}}$ model using the 3D metric Multi-Dimensional Scaling (MDS) [295].

Figure 7.10 shows the down-projected features after applying MDS to the representations of the penultimate layer of VLocNet++$_{\mathrm{MTL}}$ trained on the DeepLoc dataset. For comparison, we also show a similar visualization using the features learned by the PoseNet [276] model. Unlike PoseNet, the features learned by our VLocNet++$_{\mathrm{MTL}}$ model directly correspond to the red groundtruth trajectory shown in Figure 7.13 (b), whereas PoseNet fails to capture the pose distribution in several areas of the trajectory. This demonstrates that our proposed multitask VLocNet++$_{\mathrm{MTL}}$ architecture trained using our Geometric Consistency loss function learns a global localization model that is geometrically consistent with respect to the motion of the robot equipped with the camera.

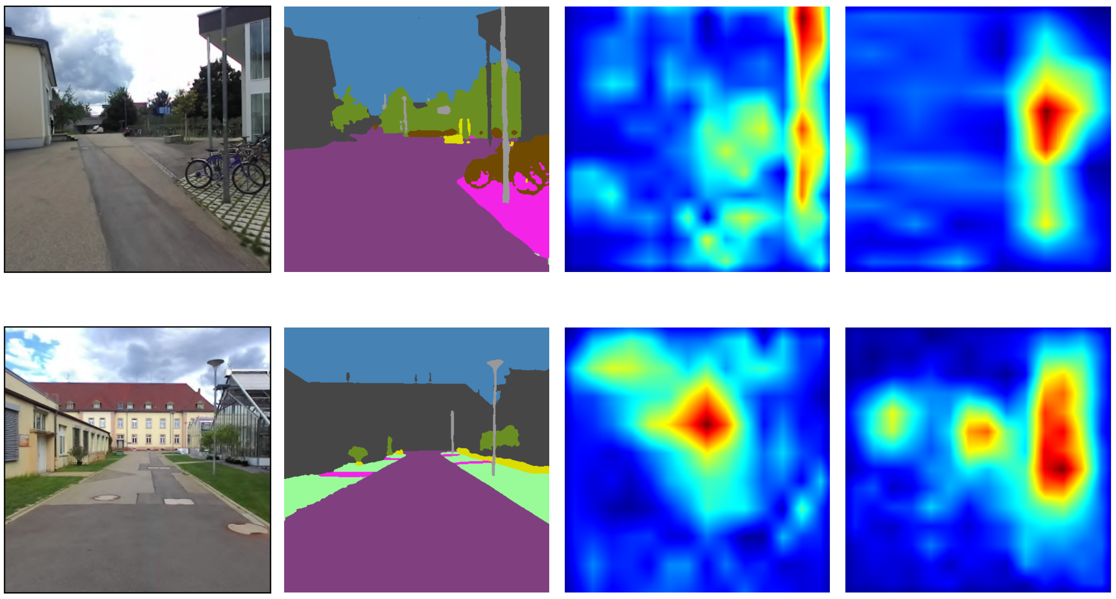


**Figure 7.11:** Comparison of the regression activation maps of our single-task VLocNet++$_{\mathrm{STL}}$ and multitask VLocNet++$_{\mathrm{MTL}}$ [34]. The visualization of the activation maps are generated using Grad-CAM++ [204] and the results are shown on the models trained on the DeepLoc dataset along with the corresponding semantic segmentation output. The color legend for the segmentation labels correspond to those shown in the benchmarking results in Table 7.11.

### 7.4.7 Visualization of the Regression Activation Maps

In an effort to investigate the effect of incorporating semantic information in the features learned by the global pose regression stream, we visualize the regression activation maps of the network for both the single-task VLocNet++$_{\mathrm{STL}}$ and multitask VLocNet++$_{\mathrm{MTL}}$ using Grad-CAM++ [204]. The Grad-CAM++ technique generates an activation map using a weighted combination of the positive partial derivatives of feature maps from the last convolutional layer of the network. The resulting activation map acts as a visual explanation to the output produced by the network. Figure 7.11 shows the input image, the corresponding semantic segmentation output from VLocNet++$_{\mathrm{MTL}}$ and the activation maps of both the single-task as well as the multitask VLocNet++ model. We present results on two example scenes from the DeepLoc dataset that contain glass facades and optical glare. Despite the challenging nature of both scenes, our model yields an accurate semantic segmentation output with high granularity. Investigating the activation maps of our single-task and multitask models, we observe that the activation maps generated from VLocNet++$_{\mathrm{MTL}}$ contains less noisy activations. Moreover, we observe that the activation maps of the multitask model places more attention on multiple stable structures that are unique to the scene such as the pole in the first example and the glass building in the bottom example. This demonstrates that fusing semantic features into the global pose regression stream enables the network to focus its attention on semantically more informative regions in the image in order to yield an accurate pose estimate.

## 7.4.8 Qualitative Evaluations

The experimental results that we have presented thus far demonstrate the capability of our proposed architectures in terms of the standard benchmarking metrics. In this section, we present exhaustive qualitative results on both the Microsoft 7-Scenes and DeepLoc datasets to analyze the performance of our models in each of the tasks.

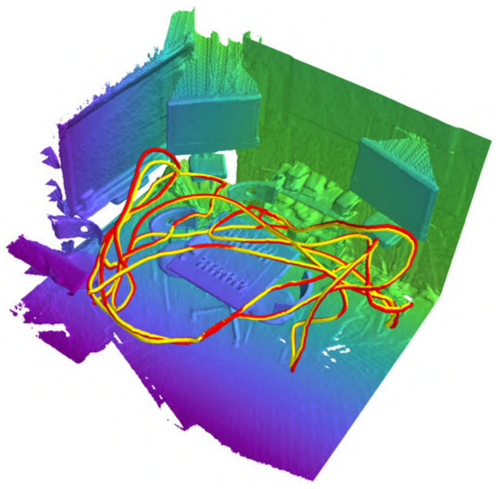
### 7.4.8.1 Qualitative Localization Results

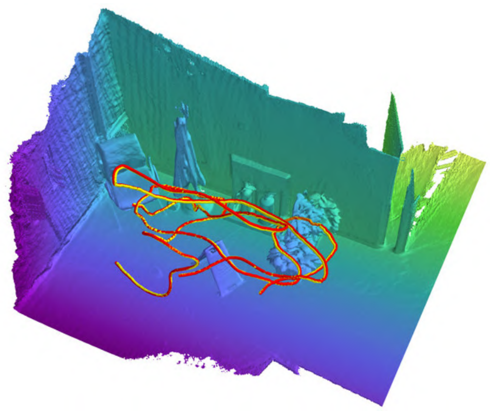
In order to qualitatively evaluate the localization performance of the VLocNet++$_{\text{MTL}}$ model in the various scenes, we present visualizations depicting the predicted poses in comparison to the groundtruth poses on the indoor Microsoft 7-Scenes dataset as well as the outdoor DeepLoc dataset in Figures 7.12 and 7.13 respectively. The predicted poses from our network are represented by the yellow trajectory and the groundtruth poses are shown in red. Note that we only show the trajectory plotted with respect to the 3D scene model for visualization purposes and our approach does not rely on the 3D model for localization rather it operates on monocular images.

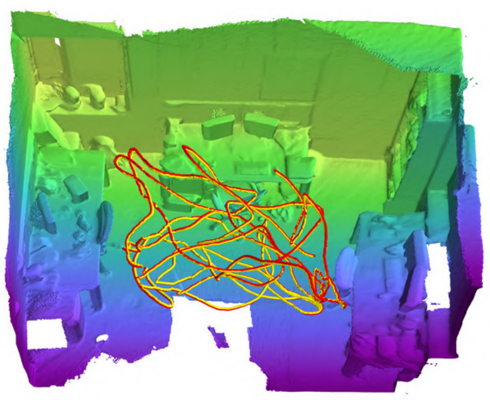
Our proposed VLocNet++$_{\text{MTL}}$ model accurately estimates the global pose in both indoor (Figures 7.12 (a), (b), (c), (d), (e), (f)), and 7.13 (a)) and outdoor (Figure 7.12 (b)) environments, while being robust to textureless regions (Figures 7.12 (b) and (c)), repetitive structures (Figure 7.12 (f)), scenes with reflective surfaces (Figures 7.12 (e) and 7.13 (a)) and motion blur (Figures 7.12 (a) and (d)). Despite these challenges, the poses predicted by VLocNet++$_{\text{MTL}}$ are well aligned with their groundtruth counterparts. Furthermore, by adopting a methodological approach to exploiting information from the multiple task-specific network streams as well as by effectively incorporating motion-specific features and encoding semantic information about the scene, VLocNet++ accurately estimate the 6-DoF global pose without requiring a 3D model of the environment. Additionally, we provide interactive visualizations of the 3D scene models along with the predicted and groundtruth poses at `http://deeploc.cs.uni-freiburg.de`.

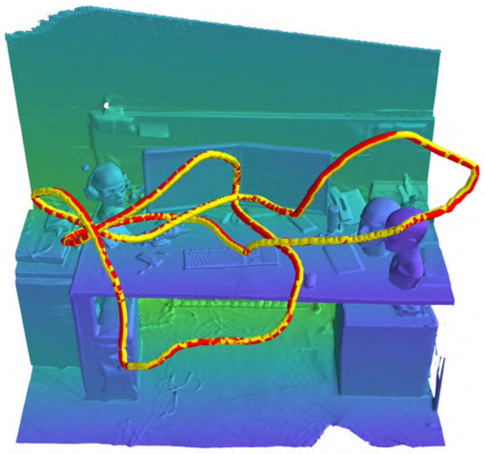
### 7.4.8.2 Qualitative Semantic Segmentation Results

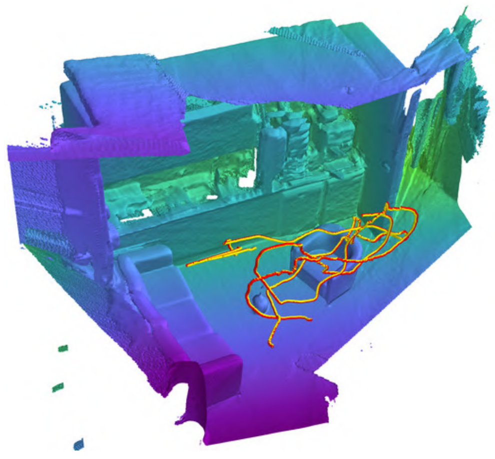
In this section, we present qualitative results of semantic segmentation using our proposed VLocNet++$_{\text{MTL}}$ architecture in comparison with the AdapNet++ model that we build upon. The primary goal of this experiment is to analyze the improvement in semantic segmentation due to the incorporation of the proposed self-supervised warping technique and the fusion of location-specific information from the global pose regression stream.


(a) Chess


(b) Fire


(c) Office


(d) Heads


(e) Pumpkin

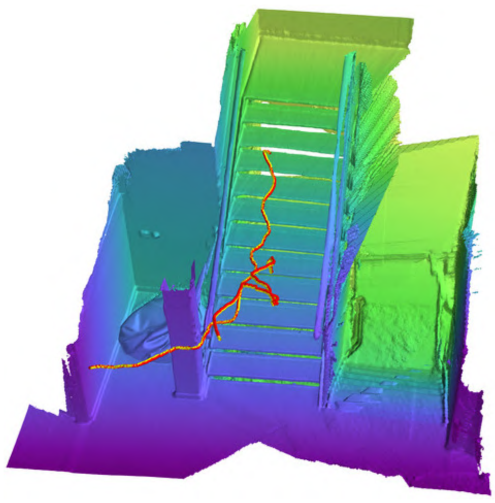
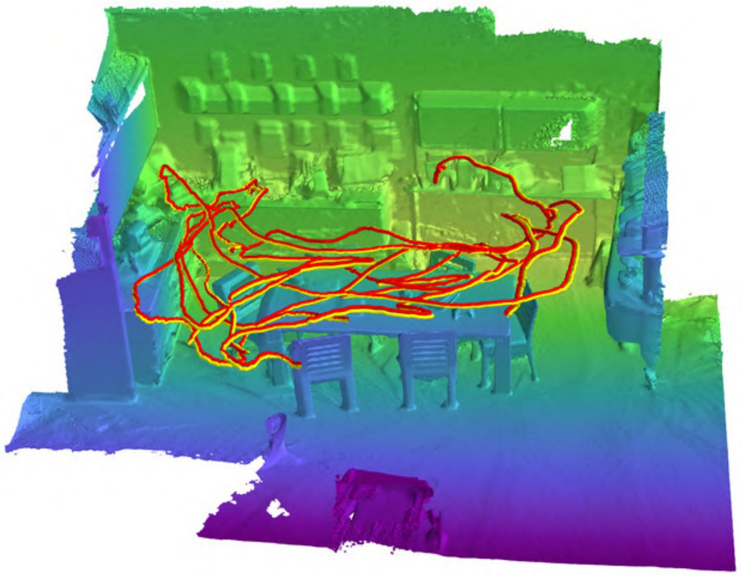

(f) Stairs

**Figure 7.12:** Qualitative localization results of the predicted global pose (yellow) versus the groundtruth pose (red) plotted with respect to the 3D scene model for visualization [34].


(a) Redkitchen

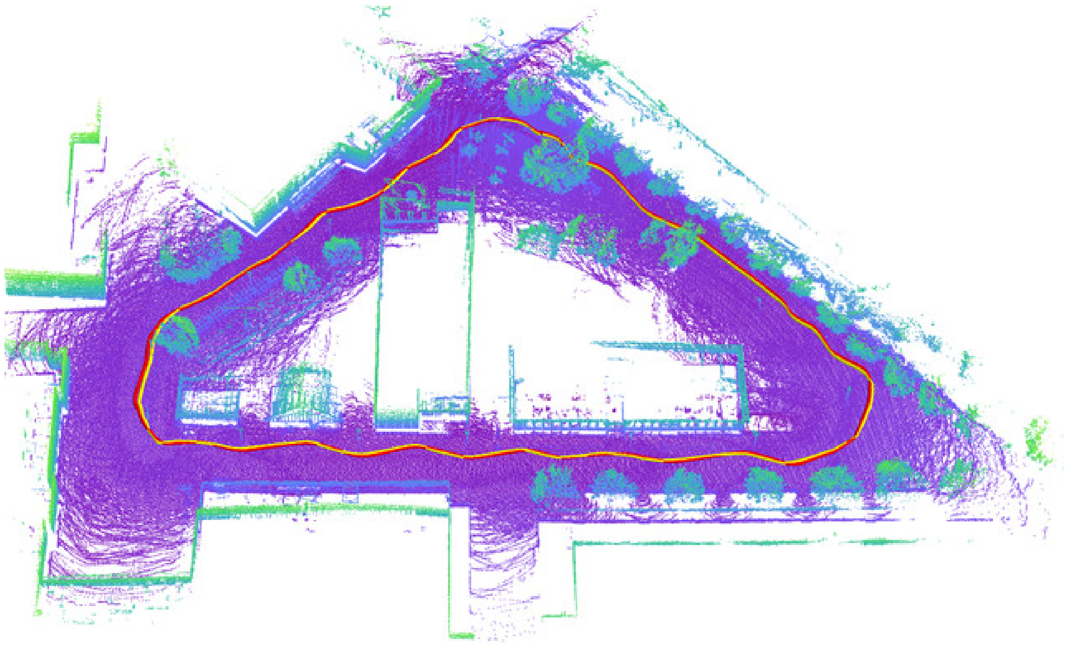

(b) DeepLoc

**Figure 7.13:** Qualitative localization results depicting the predicted global pose (yellow trajectory) versus the groundtruth pose (red trajectory) plotted with respect to the 3D scene model for visualization [34]. VLocNet++$_{\mathrm{MTL}}$ accurately estimates the global pose in both indoor and outdoor environments while being robust to textureless regions, repetitive as well as reflective structures in the environment where local feature-based pipelines perform poorly. Note that we only show the trajectory plotted with respect to the 3D scene model for visualization, our approach does not rely on a 3D model for localization. We show the second testing loop for the DeepLoc dataset, as visualizing all the testing loops in one scene creates a intertwined output that is hard to analyze.

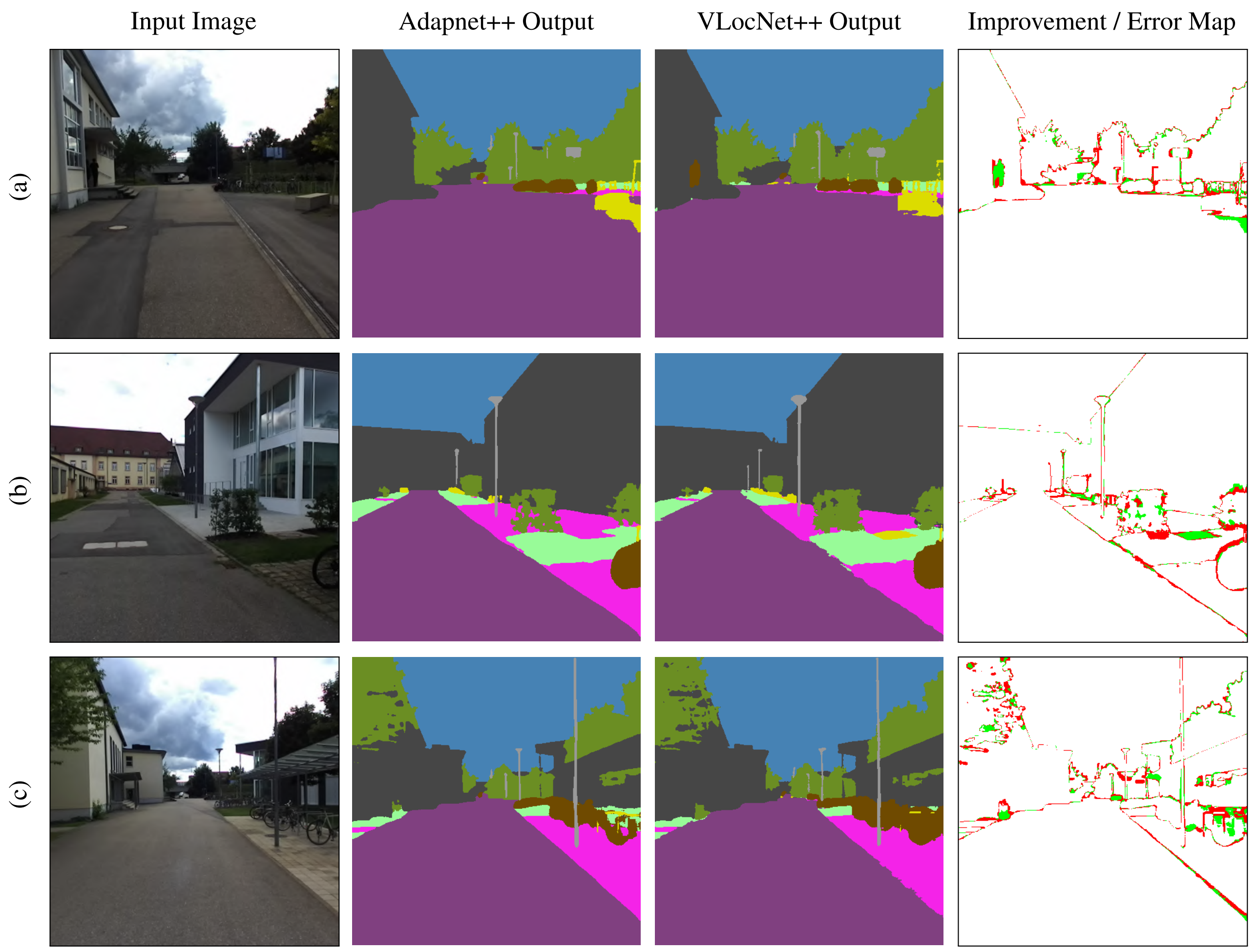


**Figure 7.14:** Qualitative semantic segmentation results of VLocNet++$_{\text{MTL}}$ in comparison to AdapNet++ on the DeepLoc dataset. VLocNet++$_{\text{MTL}}$ accurately captures the various elements of the scene that are in bad lighting or partly occluded by leveraging intermediate network representations from previous observations. The color legend for the segmentation labels correspond to those shown in the benchmarking results in Table 7.11.

We present results on the DeepLoc dataset and show the improvement / error map which indicates the improvement over the segmentation output of AdapNet++ in green and the misclassifications by the VLocNet++$_{\text{MTL}}$ model with respect to the groundtruth in red.

Figures 7.14 and 7.15 show the qualitative results from this experiment. Figure 7.14 (a) depicts a scene where the improvement in semantic segmentation output of VLocNet++$_{\text{MTL}}$ can primarily be seen in the detection of the person standing next to the building on the left as belonging to the *dynamic* object class, identification of the *vegetation* behind the underpassage in the center of the image, detection of the entire *sidewalk* that the bicyles are parked on in the center of the image and parts of the *road* on the right being corrected predicted. While, in Figure 7.14 (b), the manhole cover near the grass on the right of the image is accurately classified as the *other* object class, whereas AdapNet++ entirely misclassifies the manhole over as *sidewalk*. In addition, the third *pole* in the end of the street is entirely captured in the segmentation of VLocNet++$_{\text{MTL}}$, although it is missing in the segmentation output of the AdapNet++ model. Subsequently, in the output of VLocNet++$_{\text{MTL}}$ shown in Figure 7.14 (c), we observe that the corner of the shed on the right which is translucent, is precisely predicted as the *building* class and the bicycles on the right as well as in the center of the image are more accurately predicted in comparison to the segmentation output of the AdapNet++ model. Moreover, in all the examples shown in Figure 7.14, the boundaries between the vegetation and its surroundings are more precisely predicted in the output of the VLocNet++$_{\text{MTL}}$ model. Most of these improvements can be attributed to our proposed self-supervised warping scheme that fuses the intermediate representations from the previous timestep into the current view to improve the segmentation performance by aggregating more scene-level context. Therefore in the aforementioned scenarios where bad lighting and occlusion cause misclassifications, leveraging and incorporating features of the same location from another view improves the overall prediction.

Figure 7.15 shows the second set of examples on the DeepLoc dataset where in Figure 7.15 (a), our proposed VLocNet++$_{\text{MTL}}$ accurately classifies the entire shed in the center of the image as the *building* class and captures the whole stretch of the *sidewalk* near the trashcan in the center of the image. Whereas, AdapNet++ misclassifies these regions. Moreover, in the output of AdapNet++, we observe that in the center of the image, the network predicts parts of the building behind the trees as a tree trunk (*vegetation*) and the discolored road near the edge of the building on the left as the *building* class, where our VLocNet++$_{\text{MTL}}$ accurately predicts the object category of these pixels. In Figure 7.15 (b), we observe that VLocNet++$_{\text{MTL}}$ more precisely captures the boundaries between *grass* and *vegetation*, accurately segments the entire *sidewalk* path near the trashcan in the right and captures more of the structure of the rails on the left of the image. In the last example shown in Figure 7.15 (c), we can see that VLocNet++$_{\text{MTL}}$ accurately captures the trash can in the center-right of the road, while the structure is entirely missing the segmentation output of the AdapNet++ model. Another challenging aspect is to identify the boundary

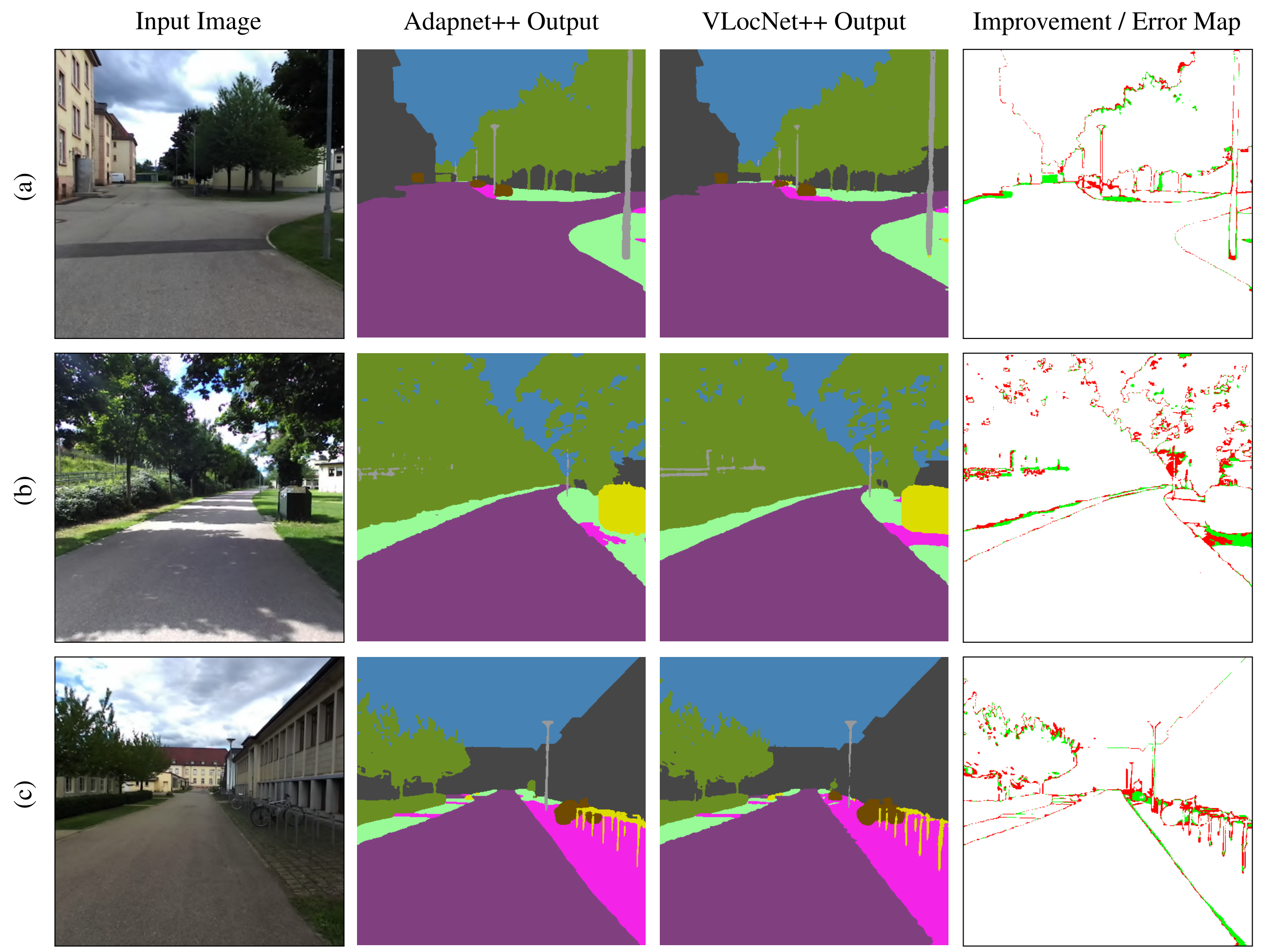


**Figure 7.15:** Qualitative semantic segmentation results of VLocNet++$_{\text{MTL}}$ in comparison to AdapNet++ in challenging scenarios in the DeepLoc dataset. VLocNet++ accurately predicts the boundaries of objects and captures entire thin pole-like structures. The color legend for the segmentation labels correspond to those shown in the benchmarking results in Table 7.11.

between *road* and *sidewalk* and well as *sidewalk* and *grass*, which can be often challenging even for humans when viewing from large distances. Inspecting the examples, we see that our VLocNet++$_{\mathrm{MTL}}$ model precisely detects these boundaries.

In the examples where an object is entirely missing in the segmentation output of the AdapNet++ model, but it is accurately captured in the output of the VLocNet++$_{\mathrm{MTL}}$ model, the improvement can be attributed to both the fusion of location specific information and the fusion of warped intermediate representations from the previous timesteps. Most of the scenes have thin pole-like structures such as lamp posts, sign posts, and fences are the hardest to fully segment for any semantic segmentation network. Although the VLocNet++$_{\mathrm{MTL}}$ architecture utilizes a considerably small input image, it is still able to detect the entire structure of pole-like objects more accurately than AdapNet++ due to the aggregation of information from previous observations. One of the challenging aspects of the DeepLoc dataset is the presence of glass buildings which are reflective or translucent that makes classifying them in images extremely challenging. Despite the presence of several glass-constructs (Figures 7.14 (b) and (c)), our network classifies the corresponding pixels as their correct semantic object categories. Finally, we observe several challenging perceptual conditions in these examples such as varying lighting conditions that cause shadows (Figure 7.15 (b)), glare (Figures 7.14 (a) and (c)) and over/under exposure due to sunlight (Figures 7.14 (c), 7.15 (a) and 7.15(b)). In all these scenarios, our models yield an accurate representation of the scene while being robust to these disturbances.

## 7.5 Related Work

In this chapter, we presented novel convolutional neural network architectures that simultaneously learn to regress the global pose, predict the semantics of the scene and estimate the visual odometry in a joint multitask learning framework. Our framework is structured to enable inductive transfer by leveraging domain specific information from each of the tasks for their mutual benefit. More importantly, we introduced a principled approach to embed geometric and semantic knowledge of the world into the pose regression network to learn a model that is geometrically consistent with respect to the motion. Over the past decade, there has been a gradual shift from employing traditional handcrafted pipelines to learning-based approaches. Although significant strides have been made, most of these approaches have been particularly proposed to address perception related tasks. However, the applicability of these learning-based techniques have also been explored for tasks in other domains. In Chapter 4, we presented a thorough review of the literature on semantic scene segmentation. In this section, we discuss some of the recent developments in multitask learning, visual localization and odometry estimation.

**Multitask Learning** can be defined as an inductive transfer mechanism that improves generalization by leveraging domain specific information from related tasks [296]. It has been applied to a wide range of tasks from different domains including image understanding [297, 298], sentiment prediction [299], language modelling and translation [300], semantic segmentation [301, 302] and even recently on learning from demonstration [303]. Yu *et al.* propose a model based on convolutional neural networks and denoising autoencoders for the task of face detection and object recognition using RGB-D images [297]. Their joint model consists of individual streams for each task, while sharing the input layer and the final hidden layer. Their approach demonstrates substantial increase in performance by employing their joint model in comparison to individually trained models. Bilen *et al.* [298] propose the use of a instance normalization layer to normalize the information across a single network that is trained to recognize across multiple visual domains including objects, digits, signs and faces. X-ResNets [299] are residual models with cross connections that have less than 40% of the parameters than their single-task counterparts and achieve competitive performance for visual sentiment concept detection. Shazeer *et al.* [300] introduce a model with a sparsely-gated mixture of experts layer with thousands of feed-forward sub-networks for the task of language modelling and machine translation.

Teichmann *et al.* propose the Multinet [301] architecture consisting of a shared encoder and task-specific decoders for classification, detection and semantic segmentation. Lu *et al.*[304] introduce an automatic approach to learn compact multitask networks for person attribute classification in which the method first starts with a thin network model and expands it during training using a multi-round branching mechanism which determines with whom each task shares features in each layer. Kuga *et al.* [305] introduce a multitask architecture that consists of multimodal encoder-decoder streams that are connected via shared latent representation as well as shared skip connections. They demonstrate that sharing representations across different modalities improves the multitask learning performance. Misra *et al.* propose Cross-stitch networks [306] for multitask learning in which they introduce the cross-stitch unit that combines activations from multiple networks by modeling shared representations as linear combinations. They demonstrate that cross-stitched networks acheive better performance than networks found by brute-force search. For combining different loss functions in a multitask model, Kendall *et al.* [302] propose a loss function based on maximizing the Gaussian likelihood using homoscedastic task uncertainty. Rouhollah *et al.* propose an architecture that takes raw images as input and generates arm trajectories for the robot to perform different manipulation actions [303]. Their work demonstrates that sharing weights across different tasks and using VAE-GAN-based reconstruction improves the generalization capability. Most of these approaches have shared sections of the network (encoders in some cases) that learn low-level representations, followed by individual task-specific branches that learn specialized features for a particular task. In this work, we proposed a novel multitask learning framework that incorporates our weighted fusion layer at multiple network stages between different task-specific streams to learn the optimal weightings for the fusion of feature maps based on region activations for the mutual benefit of the tasks.

**Visual Localization** approaches can be broadly classified into metric and appearance-based techniques. Metric approaches [307, 308] estimate the 6-DoF pose of an image by computing local image features to establish correspondences and use the matches to estimate the geometric relationship with a 3D model built using SfM or SLAM. Whereas, appearance-based localization techniques [309, 310] provide a coarse location estimate by employing image retrieval techniques to find the closest match against a database of images. The approach that we present in this chapter tackles the problem of metric localization. Sparse feature-based localization approaches learn a set of feature descriptors from training images which are then employed to learn a codebook of 3D descriptors against which a query image can be matched [274, 311]. Some approaches terminate the matching procedure as soon as a fixed threshold has been reached [312, 313], while other approaches reduce the number of full codebook searches using bidirectional matching [310, 314, 315] or employ a two-stage matching technique [316]. Relying on bidirectional information or multiple matching stages also helps to improve the accuracy by eliminating incorrect matches before the pose estimation stage. Recent methods [275, 284] that efficiently find feature correspondences within the codebook train regression forests on 3D scene data and use RANSAC to infer the final location of the query image. Donoser *et al.* propose a discriminative classification approach using random ferns which demonstrates improved pose accuracy while allowing for faster run-time [317]. Despite the accurate pose estimates provided by these methods, they still require a cumbersome feature selection step for building a codebook and the overall run-time depends on the size of the 3D model as well as the number of feature correspondences that are found. This in turn results in suboptimal performance when employed in large environments and scenes that contain textureless or repetitive structures.

Recently, pre-trained CNNs designed for classification tasks have been successfully adapted for pose regression. 6-DoF pose regression models can also be considered as multitask models as they regress for both translation and rotation components individually. Kendall *et al.* proposed the first end-to-end approach for directly regressing the 6-DoF camera pose from a monocular image using a CNN called PoseNet [276]. Since then several improvements have been proposed in terms of estimating the uncertainty of the poses using Bayesian CNNs [287], incorporating Long-Short Term Memory (LSTM) units for dimensionality reduction [277], symmetric encoder-decoder architecture for regression [291] and an improved loss function based on scene geometry [282]. Laskar *et al.* [286] propose a hybrid approach in which first a convolutional neural network trained on relative camera pose estimation is employed for feature extraction. Subsequently, the extracted features are then used to identify the nearest neighbors of the query image from a set of database images.

Deep learning approaches require large amounts of training data, however most existing visual localization datasets have sparse pose information. In order to alleviate this problem, Naseer *et al.* [285] and Wu *et al.* [283] propose augmentation strategies to increase the 3D pose coverage of images. Additionally, Wu *et al.* propose to use the Euler6 representation to counteract ambiguities of quaternions in representing angles. Recently, Brachmann *et al.* proposed a differentiable version of RANSAC termed DSAC [292] for camera localization. DSAC introduces two methods to make RANSAC differentiable; by replacing the deterministic hypothesis section using soft argmax selection or with a probabilistic selection. Subsequently, Brachmann *et al.*[293] propose DSAC2 in which they introduce an entropy controlled soft inlier count to score the hypotheses produced by the neural network. Most of the aforementioned end-to-end approaches append branches with inner-product layers to a pre-trained classification network and utilize naive loss functions for pose regression that yield a substantially lower performance than state-of-the-art local feature-based approaches. In this chapter, we introduced our Geometric Consistency loss function that constricts the search space with the relative motion information during training to obtain pose estimates that are globally consistent. Furthermore, we proposed a weighted fusion layer that enables aggregation of motion-specific temporal features while simultaneously encoding semantic features into the pose regression stream.

**Visual Odometry:** Another closely related problem in robotics is estimating the incremental motion of the robot using only sequential camera images. In one of the initially proposed methods, Konda *et al.* [318] adopt a classification approach to this problem, where a convolutional neural network with a softmax layer is used to infer the relative transformation between two images using a prior set of discretized velocities and directions. Nicholai *et al.* [288] introduce an end-to-end approach in which they combine both image and LiDAR information to estimate the ego-motion from a consecutive frames. They project the point cloud onto the 2D image and feed this information to a simple Siamese architecture with alternating convolution and pooling layers to estimate the visual odometry. Similarly, Mohanty *et al.* [289] propose a Siamese-type architecture for odometry estimation called DeepVO in which the translational and rotational components of the pose are regressed through a Euclidean loss layer with equal weight values. The architecture was based on the AlexNet model [22] and they also experiment with appending FAST features along with the images as input to the network. Melekhov *et al.* [290] add a constant weighting term to balance both the translational and rotational components of the loss, thereby yielding an improved ego-motion estimate. Furthermore, they incorporate a spatial pyramid pooling layer in their architecture which renders their model robust to varying input image resolutions. Yin *et al.* [319] addresses the critical problem of recovering the scale in monocular visual odometry by employing a convolutional neural fields to simultaneously estimate depth along with the ego-motion.

More recently, there are a class of methods that formulate structure from motion as a learning problem. Ummenhofer *et al.* [320] propose the DeMoN architecture that learns to estimate depth and camera motion from unconstrained image pairs using an novel loss function based on spatial relative differences. Their architecture consisting of multiple stacked encoder-decoder networks and an iterative network that improves its own predictions. Zhou *et al.* [321] introduce a unsupervised learning framework that uses the task of view synthesis for supervision of monocular depth and camera motion estimation from video sequences. Vijayanarasimhan *et al.* propose the SfM-Net [322] architecture that predicts pixel-wise depth, camera motion, object motion and object masks while learning in a self-supervised fashion from video sequences. Self-supervision is achieved using gradients from pixel matching across consecutive frames, constrained by forward-backward consistency on the computed motion and 3D structure. While the aforementioned techniques focus on learning supervised and unsupervised specialized models for ego-motion estimation, our goal in this work is to learn a joint multitask model for three diverse tasks and achieve superior performance compared to its single-task counterparts. Inspired by the success of residual networks in various visual recognition tasks, we proposed a Siamese-type dual stream architecture built upon the full pre-activation ResNet-50 [27] model for visual odometry estimation.

Contrary to the aforementioned task-specific methods that train individual models for a specific task, we proposed a joint end-to-end trainable multitask architecture for 6-DoF visual localization, odometry estimation and semantic segmentation from consecutive monocular images. In order to exploit the inherent interdependencies between the three tasks and to enable inductive transfer of information, we structure our multitask learning framework to be interdependent on the outputs as well as the intermediate representations of the task-specific network streams. Our approach enables a competitive and collaborative action that regularizes the network and substantially improves the performance of the joint model in comparison to individual specialized models as well as state-of-the-art techniques.

## 7.6 Conclusions

In this chapter, we introduced two novel end-to-end trainable multitask convolutional neural network architectures for 6-DoF visual localization, semantic segmentation and odometry estimation from consecutive monocular images. The goal of our architecture is to exploit the complex interdependencies within these tasks for their mutual benefit. At the lowest level, our architecture incorporates a hard parameter sharing scheme and employs a joint optimization strategy with multitask learnable loss weightings for learning inter-task correlations. We presented a new strategy for simultaneously encoding geometric and structural constraints into the the pose regression network by temporally aggregating learned motion-specific information and effectively fusing semantically meaningful representations. To this end, we proposed a weighted fusion layer that learns the most optimal weightings for fusion based on region activations. Furthermore, we presented a self-supervised warping technique that fuses feature maps from multiple views and resolutions using the representational warping concept from multi-view geometry, for scene-level context aggregation. This enables our model to be robust to camera angle deviations, object scale and frame-level distortions, while implicitly introducing feature augmentation which facilitates faster convergence. More importantly, we introduced the novel Geometric Consistency loss function that constricts the search space of the global pose regression stream while training by leveraging the relative motion information from the shared auxiliary odometry stream and by exploiting the aggregated motion-specific information to learn a model that is geometrically consistent.

In order to facilitate learning of joint multitask models for visual localization, odometry estimation and semantic segmentation, we introduced a first-of-a-kind large-scale urban outdoor localization dataset with multiple loops captured at different times of the day. We made our dataset consisting of 6-DoF camera poses and manually annotated pixel-level semantic groundtruth labels, publicly available to encourage future progress. The dataset can be utilized for benchmarking a number of perception and localization tasks as it contains challenging scenarios such as varying lighting conditions, reflective glare from the sun, shadows, motion-blur, buildings with similar facades, repetitive structures and translucent as well as reflective buildings made of glass. We presented comprehensive benchmarking results on the indoor Microsoft 7-Scenes and outdoor DeepLoc datasets. Extensive experimental evaluations demonstrate that both our single-task and multitask models achieve state-of-the-art performance on each of the tasks, compared to existing deep learning-based approaches as well as their single-task counterparts. Our proposed VLocNet++ architecture outperforms the state-of-the-art end-to-end learning model for visual localization on the Microsoft 7-Scenes benchmark by 93.81% and 91.72% in the translational and rotational components of the pose respectively. More importantly, our VLocNet++ architecture exceeds the overall state-of-the-art on the benchmark by 67.5% in the translational and 25.9% in the rotational components of the pose, while being 60.5% faster and simultaneously performing multiple tasks. The approach presented in this chapter is the first to close the performance gap between local feature-based and deep learning-based methods for visual localization. Additionally, VLocNet++ outperforms the previous state-of-the-art semantic segmentation architecture by 6.94% in the mIoU score, and demonstrates an improvement of 40.80% in the orientation for visual odometry estimation. We also presented detailed ablation studies, intuitive visual explanations and comprehensive qualitative results that demonstrate the capabilities our models. Our results indicate the feasibility of learning deep multitask models for critical robotic tasks beyond the visual perception domain. Overall, our findings are an encouraging sign that learning multitask models for various robotic tasks is a promising research direction.

# Chapter 8

# Conclusions and Discussion

This thesis addresses a number of problems related to robot perception and localization by learning to leverage the inherent structure from various modalities and across different tasks. We presented multiple innovative end-to-end convolutional neural network architectures for classifying terrains using proprioceptive sensor data, learning to semantically classify various objects present in scenes at the pixel-level, learning to dynamically fuse information from multiple modalities in a self-supervised manner, learning to jointly estimate the semantics as well the motion of objects in scenes at the pixel-level, and finally learning to visually localize, semantically segment scenes and estimate the visual odometry in a multitask framework. Using extensive experiments on standard benchmark datasets and in different real-world environments, we demonstrated that our proposed architectures substantially exceed the state-of-the-art while enabling efficient deployment on real robot systems. Additionally, we also addressed several important fundamental scientific questions in deep learning while tackling these complex problems. We believe that these techniques play a crucial role in enabling our contributions to be practical solutions for real robot systems.

We first tackled the challenge of learning to classify terrains from vehicle-terrain interaction sounds in an end-to-end manner. Typically, handcrafted audio features were used along with specialized preprocessing steps that do not generalize due to the unstructured nature of the vehicle-terrain interaction sounds. We proposed two novel convolutional neural network architectures that incorporate our new global statistical pooling strategy to achieve the time-series representation learning. Furthermore, we also proposed a recurrent architecture that exploits the temporal dynamics of the signal to further improve upon the performance. In order to enable our model to be robust to different ambient environmental noises, we proposed a noise-aware training scheme that randomly injects ambient noise samples during training to regularize the network and to learn more generalizable models. Our contribution is the first approach to address the problem using an end-to-end learning technique. We extensively evaluated our networks on over six hours of vehicle-terrain interaction data that contains nine different indoor and outdoor terrains. The results demonstrate that both our networks significantly outperform existing techniques thereby achieving state-of-the-art performance with a substantially faster inference time. We also presented experiments with an inexpensive low-quality microphone in a new environment that demonstrates the significant noise robustness and the hardware independence of our approach.

As accurate scene understanding is precursor for autonomous navigation, our second contribution enables a robot to efficiently understand the semantics of the environment from visual images. We proposed two fully-convolutional neural network architectures that are based on the encoder-decoder design topology. These architectures incorporate our multiscale residual units that are effectively better at learning multiscale features than the commonly employed multigrid method. We also proposed the efficient atrous spatial pyramid pooling module that has a large effective receptive field to capture long-range context and to aggregate multiscale representations. This module substantially improves the performance over the standard atrous spatial pyramid pooling while consuming less than ten-times the amount of parameters. In order to obtain a high resolution segmentation output, we proposed a new decoder with skip refinement stages, complemented with a multi-resolution supervision scheme to deeply supervise the training. Our proposed decoder substantially improves the segmentation along object boundaries and effectively recovers the structure of thin pole-like objects. In order to enable efficient deployment on embedded GPUs, we proposed a network-wide holistic pruning approach that compresses our models without leading to a drop in the performance. Comprehensive benchmarking results on Cityscapes, Synthia, SUN RGB-D, ScanNet and Freiburg Forest datasets demonstrate that our architectures achieve state-of-the-art performance while consuming lesser number of parameters and having a faster inference time than existing methods. We also presented real-world experiments using our AIS perception car that demonstrates the generalization ability of our models.

Subsequently, we addressed the problem of multimodal semantic segmentation in an effort to adaptively exploit complementary features to improve the robustness of the model. Existing techniques for multimodal perception, naively concatenate features from multiple individual modality streams, however this does not enable the model to leverage the features according to critical factors that influence the fusion. We tackled this problem from two perspectives. Our first architecture probabilistically fuses semantically mature complementary features according to the object classes present in the scene. While our second architecture dynamically fuses multimodal features at different intermediate network stages according to the semantic objects, their spatial location in the world and the scene context. In order to enable the network to optimally exploit complementary features, we adopt a self-supervised learning approach to train the fusion model. Our proposed fusion schemes are independent of the base architecture and can be easily employed with any semantic segmentation framework. We evaluated our fusion approaches using multiple modalities including visual images, depth and near-infrared on the Cityscapes, Synthia, SUN RGB-D, ScanNet and Freiburg Forest datasets. We demonstrated that both our fusion models exceed the performance of unimodal segmentation as well as existing multimodal fusion techniques and achieve state-of-the-art performance on each of the aforementioned datasets. More importantly, we presented results in adverse perceptual conditions including snow, rain, fog, night-time, motion blur and optical glare that shows the exceptional robustness of our models. Furthermore, we presented real-world navigation experiments using our Viona robot that employs only our multimodal fusion model as a perception module while autonomously navigating several kilometers of an unstructured forested environment.

In an effort to learn a more informative scene understanding model, we further tackled the problem of joint semantic motion segmentation with the additional goal of exploiting representations from both tasks for their mutual benefit. Most existing semantic segmentation networks do not exploit motion information and existing motion segmentation networks do not exploit semantics of the scene. However, both motion and semantics can provide complementary information that we exploit in our proposed architectures. Our first contribution for this problem is a two-stream fully-convolutional architecture that simultaneously learns semantic features using an encoder-decoder network and motion features from self-generated optical flow maps. The network then fuses semantic features into the motion segmentation stream to yield the pixel-wise segmentation output in which each pixel is assigned to a semantic object class as well as a static or moving motion label. As the ego-motion of the robot itself creates spurious optical flow magnitudes that do not represent the motion of moving objects in the scene, we proposed an ego-flow suppression technique that we incorporated in our architectures to compensate for this effect. Our second architecture that we introduced simultaneously exploits motion cues to improve leaning of semantics and adaptively fuses semantic features into the motion segmentation stream to further improve motion segmentation. Our temporal warping scheme first transforms the ego-flow suppressed optical flow maps into an edge-enhanced flow representation which is then used to warp and dynamically fuse intermediate network representations in the semantic stream across the temporal dimension. Our warping scheme substantially improves the semantic segmentation performance along the boundaries of the objects. Using extensive experiments on the Cityscapes-Motion, KITTI-Motion and ApolloScape-Motion datasets, we demonstrated that our architectures outperform existing semantic motion segmentation techniques, as well as specialized task-specific networks, thereby achieving state-of-the-art performance. Our experiments show that fusing semantic features into the motion segmentation network makes the model more robust to segmenting multiple moving objects in extremely cluttered scenes and the adaptive fusion prevents over-segmentation of the moving objects. Additionally, we presented real-world experiments using our AIS perception car that demonstrates the effectiveness of our model in challenging scenarios and the generalization ability to previously unseen environments.

Furthermore, a key contribution of this thesis is the multitask learning architectures that we proposed for geometrically consistent semantic visual localization. These architectures enable robots to simultaneously localize, understand the semantics of the scene and estimate their ego-motion using a single coherent framework. Thus far, most existing networks that learn to regress the 6-DoF global pose employ the standard Euclidean loss function that does enable the network to exploit geometric information about the environment. In order to address this problem, we proposed two novel architectures complemented with a new Geometric Consistency loss function. Our first architecture consists of a single stream for learning the global pose and a Siamese-type double stream architecture for odometry estimation. Our proposed Geometric consistency loss function exploits the relative motion information from the auxiliary odometry stream to constrict the search space of the global pose regression stream to learn a model that is globally consistent. In order to conserve the amount of parameters consumed by the network and enable inter-task learning, our network streams employ hard parameter sharing that enables exploiting the complex interdependencies between the tasks. We additionally proposed an improved architecture that fuses semantically meaningful representations into the global pose regression stream to encode structural constraints and simultaneously aggregates motion-specific information using our weighted fusion layer to enable our Geometric Consistency loss to effectively leverage this information. Our proposed weighted fusion layer enables learning of optimal weightings for the fusion based on region activations. Furthermore, we proposed a self-supervised warping technique that exploits the estimated relative motion from the odometry stream to warp semantic features from the previous frame with the representations of the current frame to implicitly introduce feature augmentation that accelerates the training and improves the semantic segmentation performance. We presented comprehensive experimental evaluations on the indoor Microsoft 7-Scenes dataset and the outdoor DeepLoc dataset that demonstrate that our networks outperform their single-task counterparts as well as existing multitask methods, thereby setting the new state-of-the-art on both these benchmarks while simultaneously performing multiple tasks. The results highlight that our network is the first deep learning approach to outperform local-feature based techniques in visual localization, in addition to being more robust in situations that contain textureless regions and glass constructs. Overall, our results demonstrate the benefit in learning these three diverse tasks in a multitask learning framework to obtain a more accurate and compact model.

In summary, we proposed several contributions in this thesis that enable robots to reliably perceive and understand our complex dynamic world using multiple modalities, and robustly localize themselves in the environment using only visual images. Our proposed models generalize effectively to different challenging scenarios and adverse perceptual conditions by dynamically adapting their weights as well as by learning in a self-supervised manner. Moreover, our architectures outperform the state-of-the-art in each of the addressed tasks with a faster inference time while consuming substantially lesser number of parameters. These factors are critical enablers for efficient real world deployment. We believe that our proposed methods have brought us closer to deploying intelligent robots in increasingly complex environments and we hope that this thesis inspires future work that will lead to revolutionary new applications.

# Future Work

There are several avenues for future research as well as straightforward extension of the work that we presented in this thesis. We presented highly accurate approaches for terrain classification using proprioceptive sensors that enables a robot to classify the terrain underneath its wheels. However, robots also need to estimate the traversability of distant terrains that are in front of it, in order to plan trajectories that are more efficient for traversal. We can leverage our proprioceptive terrain classifier to provide labels for training an exteroceptive classifier in a self-supervised manner. This technique will alleviate the need of requiring annotated groundtruth labels. For example, consider a patch of terrain in front of the robot that an exteroceptive sensor such as a camera is used to capture and assume that the robot is equipped with a microphone as well as our audio-based vehicle-terrain interaction classifier. The robot can acquire terrain labels for the previously observed visual patch captured by the camera by traversing to that location and assigning the output of the audio classifier as the label for the visual patch. A similar technique [323] was previously explored for planetary rovers where an SVM was trained to classify the terrain using handcrafted features. However, in the previous work, the exteroceptive classifier was trained in a supervised manner once enough data was gathered. For future work, we propose to utilize our CNN-based proprioceptive classifier to train an unsupervised visual classifier using a deep clustering approach [324] in a self-supervised manner. Furthermore, an exploration planner with a suitable cost function can be employed in conjunction with this system to enable the robot to traverse to locations consisting of terrains that the exteroceptive classifier is more uncertain about. This would enable the robot to traverse the environment and learn in a self-supervised manner from experience leading to a life-long learning framework.

Semantic scene understanding is one of the fundamental problems in robotics and computer vision for which there are numerous challenges to overcome. Recently, more light-weight classification networks have been proposed than the residual network architecture that we build upon for the encoder of our semantic segmentation models. Architectures such as ShuffleNet v2 [325] and MobileNet v2 [326] utilize group convolutions and depth-wise separable convolutions to keep the number of FLOPS low, thereby demonstrating faster inference time. This also enables building deeper architectures with the number of parameters still substantially lower than residual networks. An interesting research direction would be to employ either ShuffleNet v2 or MobileNet v2 as the base encoder architecture and build upon it to effectively learn multiscale features. In this thesis, we proposed the eASPP module to aggregate multiscale features with a large receptive fields. However, we use the same dilation rate in the consecutive atrous convolutions in a branch of the eASPP. Using a larger dilation rate in the second atrous convolution is one straightforward solution to learn larger effective receptive fields and therefore capture larger context. Solutions to other challenges such as developing an efficient decoder that learns to upsample as opposed to simple bilinear upsampling and more aggressive automatic pruning strategies have to be explored.

In this thesis, we presented robust techniques for multimodal perception that adapts to the observed scene in order to optimally leverage complementary features. For future work in this domain, we propose to first extend our SSMA fusion mechanism for multimodal fusion of more than two modalities. This extension is straightforward as the topology of our SSMA unit can take multiple tensors as input and it would yield output weights with the dimensions of the concatenated input tensors correspondingly for suppressing or enhancing the modality-specific feature maps. Subsequently, we propose to develop a network to first estimate the usefulness of a particular modality and then act as a switch that activates or deactivates the modality-specific streams based on this factor. This technique will also make the perception system more robust to sensor failures as the network would detect that there is no information from a specific modality and disconnect the corresponding stream. No such known approach exists to the best of our knowledge.

We addressed the problem of semantic motion segmentation in this thesis where our network simultaneously predicts the pixel-wise semantic object labels as well as the the motion status of each pixel in the image. One future direction would be to exploit the motion cues to infer the instances of the objects which can be utilized by robots for reasoning about the environment more effectively. Currently, there are several methods that have been proposed for instance segmentation [327, 328, 329], motion segmentation [54, 269, 330] and semantic segmentation [59, 133, 135] individually, however to the best of our knowledge there are no approaches that exploit motion information to implicitly infer instances for semantic segmentation. Instance separation boundaries can be learned from the motion of moving objects assuming that two objects do not move with the same velocity adjacent to each other. This problem is significantly more challenging without making this assumption and therefore poses an interesting research problem. Another potential research direction relates to how the ego-flow suppression is introduced in the network and the supervised learning of optical flow. Currently the IMU measurements are leveraged to compute the relative motion with a predicted depth map of scene from a CNN and the optical flow maps are learned using the FlowNet3 architecture that we embed in our network. Recently, several techniques have been proposed for unsupervised learning of optical flow and ego-motion [331, 332] that demonstrate impressive results without needing any labeled training data. An interesting research direction would be to reformulate our semantic motion segmentation problem in the context of multitask learning of semantics, motion segmentation, instance segmentation, optical flow and ego-motion in a single coherent framework. In this framework, only the semantic segmentation network has to be supervised, the other tasks can be learned in an unsupervised manner. This would also pose an interesting challenge for the optimization of the model as it involves simultaneously training five different diverse classification and regression tasks.

The VLocNet++ architecture that we presented is the first deep learning approach to achieve state-of-the-art performance for visual localization. The effectiveness of our architecture is primarily due to the efficient aggregation of motion-specific information and the Geometric Consistency loss function that constricts the search space with respect to the relative motion. Our architecture only uses two consecutive images as input to learn a globally consistent model. The performance of the model can be substantially improved by considering a temporal window of images. This involves incorporating recurrent units such as Long Short-Term Memory Units (LSTMs) [96] or Gated Recurrent Units (GRUs) [333] in the end of the global pose regression stream as well as in the odometry stream. Secondly, our Geometric Consistency loss function has to be adapted to minimize the loss with respect to a sequence of relative poses. Furthermore, the robustness of the network in dynamic environments can be substantially improved by additionally feeding the predicted pixel-wise ephemerality mask [334] as inputs to the odometry network stream. This would definitely be a fruitful research avenue.

In conclusion, the aforementioned research directions are only a small subset of the interesting problems that can be solved as an extension of this thesis. There are several new challenges that are arising every day due to the fast paced development of this field. This is the most exciting time to be working in this domain of robotics and we believe the best is yet to come.

# Appendices

# Appendix A

# Detailed Multimodal Semantic Segmentation Results

In this appendix, we present detailed multimodal semantic segmentation results on the Cityscapes [143], Synthia [144], SUN RGB-D [145], ScanNet [146] and Freiburg Forest [50] datasets. We compare the individual object class IoU scores of our proposed CMoDE and SSMA fusion models that were introduced in Chapter 5 with each of the unimodal semantic segmentation models (RGB, Depth, HHA, and EVI), baseline fusion techniques (Average, Maximum, Stacking, and Late Fusion) as well as the state-of-the-art fusion approaches (LFC [50] and FuseNet [162]). The mIoU scores for these models were presented in the benchmarking results shown in Section 5.3.2 and the topology of the baseline architectures are described in Section 5.2.3.

## A.1 Evaluation on the Cityscapes Dataset

Table A.1 shows the results on the Cityscapes dataset. Our SSMA_msf model that performs RGB-HHA fusion outperforms the other techniques in 8 out of the 11 semantic object classes thereby achieving the state-of-the-art performance. The unimodal models are outperformed by the multimodal fusion approaches in each of the semantic object categories, demonstrating the utility of employing multimodal fusion not only for increasing the robustness in challenging perceptual conditions but also improving the overall performance in terms of the IoU scores. Comparing the multimodal fusion performance of our proposed SSMA_msf model with the previous state-of-the-art LFC model, we observe that our model outperforms LFC substantially in each of the semantic object categories for both RGB-D fusion and RGB-HHA fusion. Analyzing the performance of RGB-HHA fusion in comparison to RGB-D fusion, we see that the *sky* and *person* classes are more accurately classified using RGB-D data, where all the other classes are more accurately classified with RGB-HHA fusion. The better performance of the RGB-D fusion for the *sky* class can be attributed to the fact that the depth maps have no information for the sky class and it appears as a homogeneous region as seen in Figure 4.11 (d), whereas the HHA

**Table A.1:** Detailed performance comparison of multimodal semantic segmentation on the Cityscapes dataset. Results are shown in terms of the class-wise IoU scores and the corresponding mIoU scores are shown in Table 5.1.

| Network | Approach | Sky | Building | Road | Sidewalk | Fence | Veg | Pole | Car | Sign | Person | Cyclist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 🟦 | ⬛ | 🟪 | 🟪 | 🟫 | 🟩 | ⬜ | 🟦 | 🟨 | 🟥 | ⬛ |
| RGB | Unimodal | 94.18 | 91.49 | 97.93 | 84.40 | 54.98 | 92.09 | 58.85 | 93.86 | 72.61 | 75.52 | 72.90 |
| Depth | Unimodal | 80.62 | 80.68 | 96.77 | 76.28 | 38.17 | 76.78 | 43.21 | 87.26 | 40.15 | 60.29 | 49.69 |
| HHA | Unimodal | 80.44 | 81.59 | 96.69 | 76.59 | 43.94 | 78.77 | 44.54 | 87.44 | 42.31 | 60.97 | 51.17 |
| RGB-D | Average | 91.39 | 90.47 | 97.90 | 84.29 | 54.39 | 90.45 | 55.95 | 93.05 | 66.48 | 74.04 | 68.83 |
| | Maximum | 91.34 | 90.47 | 97.91 | 84.30 | 54.12 | 90.48 | 55.80 | 93.02 | 66.92 | 73.78 | 68.80 |
| | Stacking | 93.70 | 91.17 | 97.83 | 83.91 | 55.11 | 92.02 | 57.07 | 93.33 | 71.79 | 74.50 | 71.82 |
| | Late Fusion | 91.95 | 90.62 | 97.88 | 83.89 | 54.95 | 90.34 | 57.74 | 92.89 | 64.04 | 74.46 | 67.19 |
| | LFC [50] | 93.98 | 91.69 | 98.00 | 85.04 | 55.58 | 92.08 | 59.68 | 93.84 | 72.34 | 76.44 | 72.56 |
| | FuseNet [162] | 90.93 | 90.08 | 97.01 | 83.84 | 53.35 | 90.02 | 55.03 | 92.76 | 66.02 | 73.07 | 68.05 |
| | CMoDE (Ours) | 93.91 | 92.11 | 97.99 | 85.25 | 63.92 | 91.94 | 57.21 | 93.40 | 71.42 | 75.32 | 72.20 |
| | SSMA (Ours) | 94.48 | 92.51 | 98.02 | 85.17 | 59.98 | 92.65 | 62.61 | 94.21 | 74.34 | 77.64 | 73.54 |
| | SSMA_msf (Ours) | **94.65** | 92.92 | 98.12 | 85.71 | 62.27 | 93.06 | 63.78 | 94.62 | 76.50 | **80.03** | 76.24 |
| RGB-HHA | Average | 91.56 | 90.49 | 97.78 | 83.47 | 58.04 | 90.36 | 58.01 | 93.06 | 68.23 | 73.48 | 69.28 |
| | Maximum | 91.5 | 90.47 | 97.78 | 83.44 | 57.65 | 90.34 | 57.95 | 93.03 | 68.57 | 73.40 | 69.29 |
| | Stacking | 94.00 | 91.67 | 97.85 | 84.15 | 53.84 | 92.15 | 59.31 | 93.51 | 72.88 | 75.31 | 72.13 |
| | Late Fusion | 91.82 | 90.42 | 97.73 | 83.17 | 59.34 | 90.16 | 57.67 | 92.96 | 66.34 | 72.63 | 66.84 |
| | LFC [50] | 93.87 | 91.74 | 98.00 | 85.09 | 56.32 | 92.11 | 60.07 | 93.97 | 72.29 | 76.20 | 72.80 |
| | CMoDE (Ours) | 93.96 | 92.10 | 97.97 | 85.19 | **65.52** | 91.94 | 57.36 | 93.28 | 71.29 | 74.87 | 72.21 |
| | SSMA (Ours) | 94.37 | 92.55 | 98.11 | 85.55 | 61.04 | 92.67 | 63.64 | 94.37 | 74.93 | 77.44 | 74.35 |
| | SSMA_msf (Ours) | 94.57 | **93.00** | **98.22** | **86.45** | 63.85 | **93.16** | **65.91** | **94.77** | **76.63** | 80.01 | **76.86** |

image appears noisy in the *sky* region due to the angle between the gravity and the local surface normal that is computed in the HHA transformation. The performance difference between the RGB-D model and the RGB-HHA model for the *person* class is only 0.02%, therefore it is within the bounds of being model noise. Another interesting result that can be seen is that the *fence* class performs better with the CMoDE approach in comparison to the SSMA fusion technique by 1.67% in the IoU score. It appears that the probabilistic class-wise fusion in CMoDE is more beneficial in this case than the dynamic weighting that is performed by the SSMA module. Furthermore, comparing the performance of the multimodal SSMA_msf model with the unimodal RGB model for RGB-HHA fusion, we observe that the largest improvement of 8.87% and 7.06% in the IoU score is achieved by the *fence* and *pole* classes respectively, which are the hardest semantic object classes to segment due to their thin structure. Subsequently, semantic classes such as *sign* and *person* also demonstrate a notable improvement of 4.02% and 4.49% in the IoU score respectively.

## A.2 Evaluation on the Synthia Dataset

Table A.2 shows the detailed multimodal semantic segmentation results on the Synthia dataset. As the Synthia dataset does not provide camera calibration parameters, it is infeasible to compute the HHA encoding, therefore we only report results for the RGB-D fusion. Unlike the performance on the Cityscapes dataset, it can be seen that our SSMA_msf model outperforms both the unimodal models, all the baseline fusion techniques and the state-of-the-art fusion approaches in each of the individual object class IoU scores. Comparing the individual class IoU scores of our SSMA_msf model with the unimodal RGB model, once again we observe that the largest improvement in object classes that have thin structures and irregular geometric shapes. A significant improvement of 10.28%, 9.83% and 7.36% is observed for the *sign, pole* and *fence* classes respectively. Object classes that have irregular geometric shapes such as *vegetation, person* and *cyclist* also show a substantial improvement due to the multimodal fusion. Moreover, as this dataset is synthetic, the unimodal model trained on the depth modality outperforms the visual RGB model in each of the object classes. This demonstrates the benefit of employing modalities that capture the geometry of the environment. Our proposed SSMA_msf model effectively fuses geometric and appearance features thereby substantially outperforming the existing fusion techniques.

## A.3 Evaluation on the SUN RGB-D Dataset

The SUN RGB-D dataset has 37 semantic object classes making it one of the hardest datasets to benchmark on. Moreover, object classes such as *blinds, desk, shelves, floor mat* and *shower curtain*, only have a few examples making it even more challenging to learn

**Table A.2:** Detailed performance comparison of multimodal semantic segmentation on the Synthia dataset. Results are shown in terms of the class-wise IoU scores and the corresponding mIoU scores are shown in Table 5.2.

| Network | Approach | Sky | Building | Road | Sidewalk | Fence | Veg | Pole | Car | Sign | Person | Cyclist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ⬛ | 🟥 | 🟪 | 🟦 | 🟦 | 🟩 | 🟨 | 🟪 | 🟥 | 🟩 | 🟦 |
| RGB | Unimodal | 97.77 | 96.98 | 96.60 | 95.70 | 79.87 | 89.63 | 74.94 | 94.22 | 71.99 | 83.64 | 72.31 |
| Depth | Unimodal | 97.78 | 97.97 | 97.34 | 96.04 | 83.48 | 94.63 | 74.56 | 95.34 | 70.86 | 82.65 | 75.90 |
| RGB-D | Average | 98.41 | 98.07 | 97.60 | 96.72 | 83.34 | 94.04 | 77.54 | 95.98 | 76.38 | 86.61 | 76.69 |
| | Maximum | 98.40 | 98.06 | 97.58 | 96.70 | 83.21 | 93.99 | 77.38 | 95.92 | 76.03 | 86.55 | 76.56 |
| | Stacking | 98.48 | 98.14 | 97.49 | 96.65 | 80.64 | 94.24 | 79.77 | 96.10 | 75.06 | 86.53 | 75.32 |
| | Late Fusion | 98.48 | 98.15 | 97.63 | 96.78 | 84.24 | 94.56 | 78.04 | 96.12 | 76.45 | 86.43 | 77.37 |
| | LFC [50] | 98.49 | 98.14 | 97.55 | 96.57 | 84.16 | 94.59 | 76.58 | 95.88 | 75.75 | 85.33 | 77.37 |
| | FuseNet [162] | 97.27 | 96.26 | 95.93 | 95.34 | 79.02 | 89.14 | 74.59 | 93.35 | 71.76 | 82.82 | 71.66 |
| | CMoDE (Ours) | 98.32 | 98.16 | 97.65 | 96.94 | 81.63 | 94.41 | 80.02 | 96.55 | 77.21 | 87.35 | 77.07 |
| | SSMA (Ours) | 98.80 | 98.52 | 98.12 | 97.45 | 85.17 | 95.44 | 83.34 | 97.21 | 80.54 | 89.44 | 79.65 |
| | SSMA_msf (Ours) | **98.91** | **98.66** | **98.34** | **97.73** | **87.33** | **95.92** | **84.77** | **97.49** | **82.27** | **90.53** | **81.18** |

**Table A.3:** Detailed performance comparison of multimodal RGB-D semantic segmentation on the SUN RGB-D dataset.

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Blinds | Desk | Shlves | Crtain | Dresr | Pillow | Mirr |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟦 | 🟧 | 🟧 | 🟩 | 🟩 | 🟥 | 🟥 | 🟪 | 🟪 | 🟫 | 🟦 | 🟪 | 🟪 | ⬛ | ⬜ | 🟨 | 🟨 | 🟦 |
| RGB | 73.81 | 84.79 | 47.45 | 64.31 | 65.76 | 52.15 | 49.97 | 41.66 | 46.99 | 32.83 | 45.19 | 32.71 | 34.72 | 21.16 | 5.03 | 50.32 | 37.45 | 32.35 | 25.89 |
| Depth | 69.54 | 84.51 | 40.19 | 70.81 | 63.21 | 52.13 | 48.20 | 24.57 | 30.46 | 28.02 | 28.59 | 32.21 | 10.92 | 17.51 | 3.92 | 50.76 | 37.02 | 40.15 | 37.83 |
| Average | 74.79 | 86.70 | 48.32 | 72.23 | 69.53 | 56.84 | 52.69 | 38.09 | 46.37 | 34.12 | 42.62 | 35.51 | 23.79 | 22.58 | 3.88 | 56.23 | 43.30 | 40.84 | 38.69 |
| Maximum | 74.91 | 86.69 | 48.11 | 72.05 | 69.39 | 56.63 | 52.50 | 38.00 | 45.99 | 33.55 | 42.85 | 35.17 | 24.03 | 22.45 | 4.00 | 55.98 | 43.13 | 40.80 | 38.36 |
| Stacking | 75.46 | 86.90 | 43.39 | 62.02 | 67.49 | 49.89 | 50.11 | 37.57 | 43.91 | 27.66 | 45.58 | 29.15 | 28.49 | 12.24 | 3.65 | 51.46 | 36.44 | 31.81 | 28.38 |
| Late Fusion | 76.08 | 87.53 | 47.81 | 71.18 | 69.86 | 56.06 | 53.82 | 43.07 | 49.54 | 35.11 | 48.90 | 36.44 | 34.85 | 23.83 | 7.22 | 57.69 | 43.11 | 37.67 | 37.57 |
| LFC [50] | 74.73 | 87.68 | 47.24 | 71.79 | 70.86 | 57.54 | 55.06 | 41.19 | 49.62 | 34.34 | 44.42 | 34.88 | 25.56 | 19.72 | 6.33 | 57.39 | 42.22 | 40.21 | 42.16 |
| FuseNet [162] | 72.83 | 83.37 | 46.16 | 63.64 | 65.41 | 50.01 | 47.03 | 40.93 | 45.80 | 29.81 | 45.97 | 31.08 | 33.37 | 21.43 | 4.28 | 47.73 | 36.41 | 32.62 | 26.11 |
| CMoDE (Ours) | 76.93 | 88.64 | 48.51 | 71.35 | 71.55 | **60.05** | 54.39 | 45.17 | 50.84 | 40.52 | 48.56 | **61.08** | **46.81** | **27.04** | 4.88 | 61.09 | 46.80 | 41.21 | 39.32 |
| SSMA (Ours) | 79.77 | 90.77 | 46.91 | 71.05 | 71.71 | 56.90 | 53.96 | 45.57 | 50.89 | 28.95 | 53.42 | 35.34 | 7.61 | 3.64 | 6.50 | 62.02 | 41.68 | 45.18 | 45.59 |
| SSMA_msf (Ours) | **80.12** | **91.01** | 48.01 | 72.36 | 72.54 | 58.19 | 54.35 | 46.58 | 51.75 | 29.88 | 54.03 | 35.99 | 6.01 | 2.76 | 5.84 | **63.19** | 42.93 | **46.18** | 46.52 |

| Network | FMat | Clths | Ceil | Books | Fridge | Tv | Paper | Towel | ShwrC | Box | Wbrd | Person | NStnd | Toilet | Sink | Lamp | Bthtub | Bag |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟩 | 🟨 | 🟪 | 🟥 | 🟦 | 🟧 | 🟩 | 🟪 | ⬜ | 🟪 | 🟩 | 🟨 | 🟥 | 🟦 | 🟩 | 🟪 | 🟧 |
| RGB | **0.12** | 21.68 | 60.65 | 32.41 | 46.96 | 18.81 | 20.79 | 27.95 | 1.32 | 22.70 | 51.08 | 33.77 | 20.59 | 71.28 | 55.31 | 36.74 | 38.60 | 15.34 |
| Depth | 0.00 | 18.49 | 57.89 | 22.82 | 29.55 | 23.18 | 7.89 | 20.33 | 9.71 | 17.52 | 22.86 | 31.18 | 17.52 | 67.26 | 50.95 | 34.97 | 54.42 | 11.12 |
| Average | 0.00 | 21.92 | 67.31 | 31.99 | 45.98 | 25.27 | 15.72 | 27.57 | 5.62 | 25.77 | 43.24 | 40.92 | 20.35 | 75.97 | 57.75 | 41.41 | 55.33 | 16.48 |
| Maximum | 0.00 | 21.83 | 67.16 | 31.81 | 45.85 | 25.43 | 15.38 | 27.20 | 5.83 | 25.50 | 43.64 | 40.73 | 20.21 | 75.84 | 57.64 | 41.24 | 55.26 | 16.34 |
| Stacking | 0.00 | 14.78 | 58.84 | 33.61 | 38.10 | 31.88 | 18.44 | 21.98 | 1.23 | 21.09 | 47.79 | 26.10 | 20.94 | 66.34 | 55.67 | 36.75 | 29.02 | 15.70 |
| Late Fusion | 0.00 | 22.77 | 68.22 | 36.20 | 46.81 | 27.11 | 19.22 | 29.08 | 1.25 | 25.72 | 49.02 | 35.76 | 21.69 | 77.30 | 59.70 | 41.55 | 51.34 | 17.09 |
| LFC [50] | 0.00 | 23.63 | 70.81 | 35.09 | 48.46 | 27.58 | 15.89 | 27.92 | 3.21 | 24.70 | 48.69 | 40.76 | 20.30 | 77.10 | 59.07 | 42.49 | 55.78 | 17.30 |
| FuseNet [162] | 0.00 | 20.40 | 62.01 | 30.09 | 45.10 | 13.26 | 20.62 | 26.38 | 1.10 | 21.25 | 48.27 | 36.18 | 21.40 | 71.69 | 53.06 | 35.82 | 37.83 | 16.11 |
| CMoDE (Ours) | 0.00 | 23.27 | 68.95 | 28.04 | 47.15 | 21.91 | 19.81 | 22.61 | 9.62 | 27.43 | 52.74 | 33.92 | 22.03 | 68.89 | **60.97** | 36.90 | 45.95 | 14.11 |
| SSMA (Ours) | 0.00 | 27.11 | 72.31 | 35.88 | 50.98 | 54.21 | 19.19 | 37.28 | **32.47** | 28.33 | 59.76 | 42.16 | 20.61 | 76.77 | 61.00 | 41.68 | 51.10 | 15.96 |
| SSMA_msf (Ours) | 0.00 | 27.77 | 72.80 | 36.80 | 52.27 | **58.72** | 19.46 | **38.21** | 31.00 | 29.70 | **60.26** | 44.62 | 19.42 | 78.46 | 61.87 | 42.37 | 48.67 | 16.57 |

**Table A.4:** Detailed performance comparison of multimodal RGB-HAA semantic segmentation on the SUN RGB-D dataset. Results are shown in terms of the class-wise IoU scores and the corresponding mIoU scores are shown in Table 5.3.

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Blinds | Desk | Shlves | Crtain | Dresr | Pillow | Mirror |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟦 | 🟧 | 🟧 | 🟩 | 🟩 | 🟥 | 🟥 | 🟪 | 🟪 | 🟫 | 🟦 | 🟪 | 🟪 | ⬛ | ⬜ | 🟨 | 🟨 | 🟦 |
| RGB | 73.81 | 84.79 | 47.45 | 64.31 | 65.76 | 52.15 | 49.97 | 41.66 | 46.99 | 32.83 | 45.19 | 32.71 | 34.72 | 21.16 | 5.03 | 50.32 | 37.45 | 32.35 | 25.89 |
| HHA | 67.39 | 85.58 | 42.08 | 68.56 | 65.56 | 50.56 | 49.89 | 25.92 | 33.54 | 25.83 | 29.06 | 29.31 | 8.28 | 15.90 | 5.57 | 48.98 | 38.24 | 42.33 | 39.06 |
| Average | 73.08 | 87.30 | 47.44 | 70.83 | 70.61 | 56.47 | 53.04 | 38.49 | 47.70 | 32.29 | 42.49 | 34.64 | 24.73 | 21.05 | 4.25 | 54.55 | 44.10 | 42.09 | 37.82 |
| Maximum | 73.23 | 87.28 | 47.37 | 70.69 | 70.47 | 56.33 | 52.86 | 38.51 | 47.47 | 32.09 | 42.57 | 34.36 | 24.92 | 21.05 | 44.68 | 54.43 | 43.90 | 42.08 | 37.54 |
| Stacking | 73.58 | 86.04 | 45.43 | 62.61 | 66.02 | 48.31 | 50.71 | 37.97 | 43.38 | 31.38 | 46.75 | 30.01 | 33.60 | 18.25 | 7.38 | 38.84 | 32.78 | 31.77 | 31.81 |
| Late Fusion | 75.87 | 87.43 | 47.64 | 71.03 | 70.54 | 56.59 | 54.75 | 43.55 | 49.91 | 35.99 | 48.53 | 36.49 | 34.40 | 20.97 | 6.46 | 57.50 | 44.12 | 38.68 | 40.39 |
| LFC [50] | 76.54 | 87.74 | 47.05 | **72.85** | 69.76 | 57.09 | 54.37 | 40.36 | 49.09 | 37.53 | 43.58 | 36.21 | 32.90 | 22.11 | 8.39 | 59.38 | 45.46 | 38.87 | 42.19 |
| CMoDE (Ours) | 76.18 | 88.73 | **49.55** | 72.24 | 71.43 | 56.09 | **55.46** | 46.26 | 50.36 | **42.19** | 47.58 | 39.21 | 25.31 | 24.87 | **48.84** | 60.61 | **48.67** | 42.98 | 44.06 |
| SSMA (Ours) | 78.94 | 90.70 | 48.03 | 70.04 | 72.15 | 55.53 | 54.29 | 45.52 | 50.22 | 38.47 | 52.68 | 38.17 | 16.97 | 4.05 | 4.14 | 58.44 | 43.63 | 43.71 | 45.40 |
| SSMA_msf (Ours) | 79.52 | **91.01** | 50.50 | 71.40 | **73.26** | 56.96 | 55.17 | **47.19** | **52.29** | 40.49 | **54.89** | 37.85 | 16.18 | 3.02 | 3.77 | 60.38 | 43.94 | 45.82 | **47.53** |

| Network | FMat | Clths | Ceil | Books | Fridge | Tv | Paper | Towel | ShwrC | Box | Wbrd | Person | NStnd | Toilet | Sink | Lamp | Bthtub | Bag |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟦 | 🟩 | 🟨 | 🟪 | 🟥 | 🟦 | 🟧 | 🟩 | 🟪 | ⬜ | 🟪 | 🟩 | 🟨 | 🟥 | 🟦 | 🟩 | 🟪 | 🟧 |
| RGB | **0.12** | 21.68 | 60.65 | 32.41 | 46.96 | 18.81 | 20.79 | 27.95 | 1.32 | 22.70 | 51.08 | 33.77 | 20.59 | 71.28 | 55.31 | 36.74 | 38.60 | 15.34 |
| HHA | 0.00 | 18.00 | 58.43 | 24.77 | 36.35 | 25.26 | 8.54 | 19.75 | 13.80 | 15.68 | 27.87 | 29.16 | 15.36 | 66.26 | 48.72 | 35.98 | 51.88 | 12.22 |
| Average | 0.00 | 22.24 | 68.54 | 34.12 | 49.24 | 28.41 | 16.67 | 26.21 | 9.71 | 24.37 | 46.23 | 44.19 | 19.54 | 75.72 | 57.35 | 41.19 | 52.42 | 18.17 |
| Maximum | 0.00 | 22.19 | 68.27 | 34.03 | 49.13 | 28.56 | 16.28 | 26.08 | 9.97 | 24.13 | 46.28 | 43.46 | 19.47 | 75.62 | 57.12 | 41.09 | 52.56 | 17.86 |
| Stacking | 0.11 | 16.03 | 61.38 | 33.31 | 44.59 | 30.36 | 21.15 | 23.32 | 0.00 | 20.71 | 48.64 | 32.62 | 20.17 | 73.70 | 54.94 | 35.17 | 39.74 | 14.57 |
| Late Fusion | 0.00 | 25.60 | 69.74 | 36.00 | 48.27 | 28.67 | 18.86 | 27.06 | 5.44 | 25.33 | 51.77 | 38.19 | 21.02 | 76.20 | 60.23 | 41.90 | 58.70 | 15.73 |
| LFC [50] | 0.00 | 23.09 | 70.20 | 34.24 | 47.60 | 28.63 | 14.11 | 28.32 | 2.72 | 25.41 | 46.03 | 36.97 | 21.09 | 77.71 | 59.57 | 42.88 | 56.88 | 13.86 |
| CMoDE (Ours) | 0.00 | 24.12 | 68.51 | 28.75 | 50.23 | 24.29 | 19.03 | 28.05 | 6.06 | 25.89 | 57.22 | 41.41 | 19.34 | 71.22 | 58.76 | 36.28 | 52.11 | 16.48 |
| SSMA (Ours) | 0.00 | 27.25 | 74.63 | 36.47 | 49.06 | 47.30 | 21.66 | 32.20 | 29.24 | 27.36 | 54.02 | 47.30 | 23.31 | 77.07 | 59.52 | 41.94 | 57.91 | 17.86 |
| SSMA_msf (Ours) | 0.00 | **28.33** | **75.14** | **38.19** | **52.34** | 57.86 | **22.68** | 35.34 | 29.90 | **30.23** | 56.24 | **48.98** | **23.44** | **78.61** | 60.08 | **43.83** | **60.12** | **19.57** |

representations of these classes. We present the experimental comparisons on this dataset in Table A.3 and A.4, for the RGB-D fusion and RGB-HHA fusion respectively. Note that we compare the performance of the individual semantic object classes across both RGB-D and RGB-HHA fusion, therefore we only highlight the score of the best performing model for a specific object class across both tables in bold.

Our SSMA_msf model trained on RGB-HHA images achieves the highest overall performance by outperforming the other fusion techniques in 25 out of the total 37 semantic object classes. Our CMoDE fusion approach achieves the highest performance in the other 10 object classes. While our standard SSMA model outperforms the other techniques for the *shower curtain* class and the unimodal visual RGB model achieves the highest performance for the *floor mat* class. Interestingly, among all the datasets, the *floor mat* object class from the SUN RGB-D dataset is the only category for which the unimodal visual RGB model achieves the highest performance. This is primarily due to two factors. The first being that the depth images do not have a high enough resolution to capture the structure of this object class, rather it appears flat as the floor, which makes learning geometric features of this object from depth or its transformed HHA variant infeasible. Secondly, there are only a handful of training examples of this class. Even with image augmentations, there are less than 40 examples in the entire training dataset. It can be seen that that the unimodal depth model and all the multimodal fusion techniques do not detect this object, while the unimodal visual RGB model achieves an IoU score of 0.12%. The results also demonstrate that the previous state-of-the-art LFC model outperforms our SSMA_msf model for RGB-HHA fusion by 1.45% only for the *bed* class, while it is outperformed by the SSMA_msf model for all the other object classes.

Comparing the performance of our multimodal SSMA_msf models with the unimodal RGB model, we observe the highest improvement of 39.91% in the IoU score for the *tv* class. This class often has a substantial amount of reflections on it which causes the unimodal visual RGB model to perform poorly. However, our SSMA_msf model exploits the complementary geometric features to more accurately segment this object. A significant improvement of 21.64%, 21.52% and 15.21% is observed for the *mirror, bathtub* and *person* categories. Both the mirror and bathtub classes also have reflective surfaces and the *person* class is often severely occluded due to clutter in this dataset which contributes to the poor performance of the unimodal visual RGB model. Our CMoDE model significantly improves the performance of semantic object classes such as *shelves, counter* and *blinds* by 43.81%, 28.37% and 12.09% respectively. Finally, our standard SSMA model achieves an improvement of 31.15% for the *shower curtain* class. Note that we only described the substantially large improvements, the other object classes also show notable improvements. It can be observed that on this dataset, the improvement due to multimodal fusion is significantly larger than on the other datasets due to the indoor nature of the scenes which enables the depth modality to capture rich information of the entire scene, unlike in outdoor environments where the depth modality does not provide any

**Table A.5:** Detailed performance comparison of multimodal semantic segmentation on the ScanNet dataset. Results are shown in terms of the class-wise IoU scores and the corresponding mIoU scores are shown in Table 5.4.

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Desk | Curtain | Fridge | ShwrC | Toilet | Sink | Bthtub | OthrF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟫 | 🟦 | 🟨 | 🟪 | 🟩 | 🟫 | 🟪 | 🟩 | 🟦 | 🟦 | ⬛ | 🟨 | 🟦 | 🟪 | 🟪 | 🟩 | 🟩 | 🟥 | 🟧 | 🟦 |
| RGB | 74.24 | 71.95 | 48.78 | 76.06 | 62.72 | 63.55 | 56.71 | 52.52 | 53.41 | 52.29 | 38.43 | 45.19 | 41.43 | 64.30 | 60.91 | 40.64 | 56.86 | 30.62 | 38.60 | 24.34 |
| Depth | 73.16 | 86.00 | 47.36 | 80.74 | 67.82 | 66.33 | 57.80 | 38.85 | 39.33 | 56.31 | 32.08 | 44.39 | 35.10 | 55.97 | 55.28 | 37.08 | 63.62 | 46.92 | 64.61 | 31.14 |
| Average | 75.36 | 83.41 | 50.54 | 82.98 | 67.89 | 71.63 | 58.74 | 48.74 | 50.34 | 69.21 | 35.53 | 45.06 | 35.69 | 59.22 | 64.78 | 45.51 | 62.30 | 40.38 | 63.62 | 32.20 |
| Maximum | 75.34 | 83.31 | 50.40 | 82.64 | 67.66 | 71.25 | 58.54 | 48.30 | 49.97 | 69.15 | 35.72 | 44.51 | 35.26 | 59.08 | 64.38 | 45.02 | 62.31 | 40.11 | 62.81 | 32.22 |
| Stacking | 73.08 | 80.82 | 43.92 | 75.38 | 60.82 | 61.83 | 52.32 | 47.58 | 49.12 | 65.11 | 29.92 | 40.06 | 38.23 | 52.57 | 63.66 | 42.03 | 57.38 | 39.89 | 49.12 | 32.61 |
| Late Fusion | 75.12 | 81.86 | 53.14 | 84.52 | 69.47 | 71.99 | 60.10 | 50.61 | 55.31 | 69.35 | 41.90 | 48.21 | 41.69 | 58.87 | 62.08 | 45.59 | 64.96 | 39.61 | 55.39 | 33.82 |
| LFC [50] | 77.59 | 84.55 | 55.07 | 85.07 | 70.24 | 73.38 | 60.20 | 52.62 | 56.29 | 70.23 | 42.37 | 46.76 | 42.89 | 61.16 | 69.26 | 45.38 | 67.07 | 47.40 | 66.61 | 35.94 |
| FuseNet [162] | 69.19 | 77.39 | 47.30 | 72.97 | 58.14 | 59.74 | 50.49 | 43.36 | 50.60 | 67.35 | 27.88 | 39.99 | 32.06 | 50.43 | 61.40 | 30.52 | 47.42 | 27.57 | 39.99 | 27.86 |
| CMoDE (Ours) | 77.39 | 86.69 | 54.84 | 82.60 | 72.35 | 71.88 | 67.71 | 52.32 | 55.46 | 72.24 | 41.78 | 45.63 | 50.97 | 62.46 | 73.71 | 47.58 | 61.98 | 53.50 | 62.64 | 41.20 |
| SSMA (Ours) | 83.37 | 89.19 | 62.99 | 83.91 | **76.51** | 81.57 | 54.22 | 61.29 | 43.29 | 65.37 | 54.84 | 46.35 | 46.47 | 60.04 | **75.38** | 55.57 | 67.80 | 58.88 | 73.07 | 43.77 |
| SSMA_msf (Ours) | **84.96** | 88.85 | **64.57** | **87.87** | 75.57 | 82.34 | 57.44 | 61.86 | 43.12 | 68.68 | 53.53 | **49.24** | 47.82 | 64.66 | 75.20 | **58.18** | 66.66 | **62.52** | **73.23** | **47.17** |

| Network | Wall | Floor | Cabnt | Bed | Chair | Sofa | Table | Door | Wndw | Bshelf | Picture | Cnter | Desk | Curtain | Fridge | ShwrC | Toilet | Sink | Bthtub | OthrF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 🟫 | 🟦 | 🟨 | 🟪 | 🟩 | 🟫 | 🟪 | 🟩 | 🟦 | 🟦 | ⬛ | 🟨 | 🟦 | 🟪 | 🟪 | 🟩 | 🟩 | 🟥 | 🟧 | 🟦 |
| RGB | 74.24 | 71.95 | 48.78 | 76.06 | 62.72 | 63.55 | 56.71 | 52.52 | 53.41 | 52.29 | 38.43 | 45.19 | 41.43 | 64.30 | 60.91 | 40.64 | 56.86 | 30.62 | 38.60 | 24.34 |
| HHA | 72.01 | 84.23 | 46.93 | 81.33 | 68.21 | 64.89 | 56.41 | 36.72 | 38.24 | 43.75 | 35.64 | 42.49 | 30.29 | 58.17 | 61.74 | 43.43 | 60.62 | 43.25 | 60.25 | 32.73 |
| Average | 74.11 | 83.04 | 51.51 | 82.37 | 67.91 | 68.48 | 57.30 | 44.36 | 47.79 | 66.02 | 36.12 | 44.83 | 32.12 | 59.64 | 67.92 | 47.42 | 64.57 | 38.71 | 59.31 | 33.60 |
| Maximum | 74.16 | 83.14 | 51.38 | 82.25 | 67.85 | 68.32 | 57.04 | 44.20 | 47.54 | 65.74 | 36.30 | 44.41 | 31.74 | 59.62 | 67.72 | 47.37 | 64.28 | 38.70 | 58.86 | 33.67 |
| Stacking | 71.67 | 83.71 | 46.81 | 73.11 | 63.02 | 60.45 | 52.28 | 47.45 | 47.67 | 67.98 | 31.28 | 42.17 | 35.10 | 56.28 | 65.52 | 42.91 | 59.21 | 38.80 | 53.51 | 31.22 |
| Late Fusion | 74.84 | 83.45 | 53.08 | 83.86 | 70.41 | 70.61 | 59.27 | 49.44 | 54.03 | 65.57 | 41.00 | 47.97 | 40.23 | 60.62 | 68.92 | 45.90 | 67.45 | 39.63 | 53.92 | 34.74 |
| LFC [50] | 77.20 | 84.96 | 55.72 | 84.75 | 71.39 | 71.45 | 59.45 | 50.09 | 53.64 | 68.55 | 42.75 | 46.20 | 40.54 | 61.14 | 72.77 | 48.02 | **67.47** | 46.38 | 63.40 | 36.35 |
| CMoDE (Ours) | 76.36 | 86.07 | 54.56 | 80.05 | 71.64 | 78.53 | 68.40 | 50.22 | 55.78 | **78.65** | 49.73 | 42.14 | **54.20** | 64.51 | 72.01 | 49.19 | 62.11 | 49.99 | 65.12 | 36.27 |
| SSMA (Ours) | 82.37 | **89.25** | 55.56 | 87.06 | 74.15 | 80.45 | 68.82 | 59.94 | **57.05** | 68.06 | 55.84 | 28.07 | 52.05 | 70.40 | 74.02 | 53.53 | 64.10 | 59.21 | 70.65 | 40.23 |
| SSMA_msf (Ours) | 81.63 | 88.85 | 57.08 | 85.89 | 74.52 | **84.69** | **71.69** | **64.70** | 55.57 | 72.18 | **59.89** | 29.64 | 52.64 | **72.04** | 73.41 | 56.68 | 62.92 | 60.95 | 72.61 | 40.46 |

information for distant objects.

## A.4 Evaluation on the ScanNet Dataset

Table A.5 presents the multimodal fusion results on the recently introduced ScanNet dataset. Our proposed SSMA_msf model outperforms the unimodal models as well as the multimodal fusion approaches in 13 out of the total of 20 semantic object classes in the dataset. While our standard SSMA model achieves the highest performance for 4 of the other classes and our CMoDE model achieves the highest performance for the *bookshelf* and *desk* classes. Interestingly the previous state-of-the-art LFC fusion technique outperforms our SSMA_msf model by 4.55% for the *toilet* class, while our model achieves a higher performance for all the other semantic object classes. Nevertheless, comparing the performance of our SSMA_msf model with the unimodal visual RGB model, we observe a significant improvement of 34.63% and 31.90% for the *bathtub* and *sink* classes respectively. In addition, we observe a substantial improvement of over 20% for several semantic object classes including *sofa, picture* and *other furniture*.

Comparing the performance of our proposed SSMA_msf model with the FuseNet model for RGB-D fusion, we observe that our model outperforms FuseNet significantly in 19 out of the total of 20 semantic object categories. Our model is outperformed by FuseNet by 7.46% for the *window* class. However, our model outperforms FuseNet in the *window* class using RGB-HHA fusion. Similar to the performance of our models on the SUN RGB-D dataset, the large improvement due to multimodal fusion on this dataset can be attributed to the indoor scenes for which the depth modality contains rich information. The depth maps in the ScanNet dataset are considerably less noisy than those in the SUN RGB-D dataset. This correlates to the larger improvement that we obtain due to multimodal fusion on this dataset. Moreover, this is the first real world dataset in which the unimodal model trained on depth images outperforms the corresponding visual RGB model. This shows that geometry information can be more informative for learning semantics of the scene than visual appearance information as long at it accurately captures the geometry of the entire scene. Our results demonstrate that the performance of multimodal fusion is largely limited by the quality of complementary information captured in the modalities as well as the noise in the sensor data.

## A.5 Evaluation on the Freiburg Forest Dataset

Finally, Table A.6 shows the multimodal fusion results on the Freiburg Forest dataset. This dataset consists of three inherently different modalities that provide appearance, geometry and reflectance information. It contains scenarios in challenging perceptual conditions including low lighting, shadows, glare on the optics, motion blur, snow, over exposure and

**Table A.6:** Detailed performance comparison of multimodal semantic segmentation on the Freiburg Forest dataset. Results are shown in terms of the class-wise IoU scores and the corresponding mIoU scores are shown in Table 5.5.

| Network | Approach | Trail ⬜ | Grass 🟩 | Veg. 🟫 | Sky 🟦 | Obstacle 🟨 |
|---|---|---|---|---|---|---|
| RGB | Unimodal | 89.27 | 89.41 | 91.43 | 93.01 | 52.32 |
| Depth | Unimodal | 73.22 | 79.22 | 87.01 | 90.32 | 39.84 |
| EVI | Unimodal | 87.66 | 86.81 | 89.40 | 92.11 | 48.79 |
| RGB-D | Average | 85.90 | 84.57 | 86.59 | 89.93 | 50.54 |
| | Maximum | 88.23 | 87.01 | 88.45 | 91.12 | 53.28 |
| | Stacking | 89.51 | 89.31 | 91.22 | 92.85 | 52.77 |
| | Late Fusion | 88.59 | 88.53 | 90.86 | 92.75 | 49.77 |
| | LFC [50] | 87.65 | 88.18 | 90.45 | 92.13 | 49.73 |
| | FuseNet [162] | 88.36 | 88.29 | 90.68 | 92.44 | 49.39 |
| | CMoDE (Ours) | 89.28 | 89.32 | 91.31 | 92.84 | 53.31 |
| | SSMA (Ours) | 90.23 | 90.08 | 92.01 | 93.62 | 53.08 |
| | SSMA_msf (Ours) | **90.42** | **90.28** | **92.16** | **93.77** | 53.33 |
| RGB-EVI | Average | 89.49 | 88.97 | 91.02 | 92.96 | 52.60 |
| | Maximum | 89.49 | 88.97 | 91.02 | 92.96 | 52.60 |
| | Stacking | 89.43 | 89.42 | 91.27 | 92.82 | 52.97 |
| | Late Fusion | 89.04 | 88.63 | 90.90 | 93.02 | 52.42 |
| | LFC [50] | 89.60 | 88.98 | 91.07 | 93.04 | 52.26 |
| | CMoDE (Ours) | 89.23 | 89.31 | 91.38 | 92.97 | 53.62 |
| | SSMA (Ours) | 90.17 | 89.93 | 91.93 | 93.57 | 53.88 |
| | SSMA_msf (Ours) | 90.41 | 90.03 | 92.05 | 93.74 | **54.68** |

under exposure. However the improvement due to multimodal fusion is not significantly observed in terms of the performance metrics but rather more in the qualitative results as shown in Section 5.3.5. The results demonstrate that each of our proposed models: CMoDE, SSMA and the SSMA_msf variant, outperforms the previous state-of-the-art LFC model in each of the semantic object categories for both RGB-D and RGB-EVI fusion. Moreover, it can be seen the performance of our SSMA_msf model for RGB-D fusion and RGB-EVI fusion, both show similar improvement in comparison to the unimodal visual RGB model. The results also show that our SSMA_msf model outperforms the unimodal models as well the other multimodal fusion techniques in each of the semantic object classes in the dataset. For most semantic object categories, a marginally higher improvement can be seen using RGB-D fusion. However only the *obstacle* class achieves a higher performance using RGB-EVI fusion. Overall, we observe that simple fusion techniques such as direct concatenation of feature maps from individual modality-specific network streams at different intermediate network stages are significantly outperformed by more intelligent dynamic approaches that adapt the fusion of the modality-specific feature maps based on the observed scene. Nevertheless, the results presented in this section demonstrate the complexity of multimodal perception.

# List of Figures

# List of Tables

# Bibliography

[1] J. McCarthy, M. L. Minsky, N. Rochester, and C. E. Shannon, "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955," *AI magazine*, vol. 27, no. 4, p. 12, 2006.

[2] H. Simon, "The shape of automation for men and management," *Harper and Row*, vol. 40, 1965.

[3] D. Crevier, *AI: the tumultuous history of the search for artificial intelligence*. Basic Books, 1993.

[4] "Ai expert newsletter: W is for winter archived 9 november 2013 at the wayback machine." https://web.archive.org/web/20131109201636/http://www.ainewsletter.com/newsletters/aix_0501.htm, 2005, accessed: 10-11-2018.

[5] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[6] D. A. Forsyth and J. Ponce, "A modern approach," *Computer vision: a modern approach*, pp. 88–101, 2003.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[8] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[9] I. F. of Robotics, "Executive summary world robotics 2018 industrial robots," *Available online on http://www. ifr. org*, pp. 13–22, 2018.

[10] D. Wettergreen, G. Foil, P. M. Furlong, and D. R. Thompson, "Science autonomy for rover subsurface exploration of the atacama desert," *AI Magazine*, vol. 35, no. 4, October 2014.

[11] A. R. Lanfranco, A. E. Castellanos, J. P. Desai, and W. C. Meyers, "Robotic surgery: a current perspective," *Annals of surgery*, vol. 239, no. 1, p. 14, 2004.

[12] A. D. Bowen, D. R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J. C. Kinsey, M. Heintz, G. McDonald, D. B. Peters *et al.*, "The nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth," in *OCEANS 2008*, 2008, pp. 1–10.

[13] "irobot corp. irobot: Our history," http://www.irobot.com/us/Company/About/Our_History.aspx, 2018, accessed: 10-11-2018.

[14] I. F. of Robotics, "Executive summary world robotics 2018 service robots," *Available online on http://www. ifr. org*, pp. 13–16, 2018.

[15] "Fetch robotics," http://fetchrobotics.com, 2018, accessed: 10-11-2018.

[16] "Beam robot, suitable technologies," https://www.suitabletech.com/, 2018, accessed: 10-11-2018.

[17] A. K. Singh and G. Nandi, "Nao humanoid robot," *Robotics Autonomous Systems*, vol. 79, no. C, pp. 108–121, 2016.

[18] T. Shibata, "Development and spread of therapeutic medical robot, paro: Innovation of non-pharmacological therapy for dementia and mental health," *Journal of Information Processing and Management*, vol. 60, no. 4, pp. 217–228, 2017.

[19] "Mars science laboratory curiosity rover," https://mars.jpl.nasa.gov/msl, 2018, accessed: 10-11-2018.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[21] M. Minsky, "Neural nets and the brain-model problem," *Doctoral dissertation, Princeton University, NJ*, 1954.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[24] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.

[26] A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks,” in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, T. Jebara and E. P. Xing, Eds., 2014, pp. 1764–1772.

[27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[29] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” *arXiv preprint arXiv:1606.05250*, 2016.

[30] “Waymo,” https://waymo.com, 2018, accessed: 10-11-2018.

[31] “Uber,” https://uber.com, 2018, accessed: 10-11-2018.

[32] “Navya,” https://navya.tech, 2018, accessed: 10-11-2018.

[33] P. Gao, H.-W. Kaas, D. Mohr, and D. Wee, “Automotive revolution–perspective towards 2030 how the convergence of disruptive technology-driven trends could transform the auto industry,” *Advanced Industries, McKinsey & Company*, 2016.

[34] N. Radwan, A. Valada, and W. Burgard, “Vlocnet++: Deep multitask learning for semantic visual localization and odometry,” *IEEE Robotics And Automation Letters (RA-L)*, vol. 3, no. 4, pp. 4407–4414, 2018.

[35] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.-A. Fernandez-Madrigal, and J. González, “Multi-hierarchical semantic maps for mobile robotics,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 2278–2283.

[36] R. Drouilly, P. Rives, and B. Morisset, “Semantic representation for navigation in large-scale environments,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1106–1111.

[37] M. Johnson-Roberson, J. Bohg, G. Skantze, J. Gustafson, R. Carlson, B. Rasolzadeh, and D. Kragic, “Enhanced visual scene understanding through human-robot dialog,”

in *International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 3342–3348.

[38] "Marble - urban last-mile robots," https://angel.co/marble-11, 2018, accessed: 10-11-2018.

[39] M. Joerss, J. Schröder, F. Neuhaus, C. Klink, and F. Mann, "Parcel delivery: The future of last mile," *McKinsey & Company*, 2016.

[40] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "Herb: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, p. 5, 2010.

[41] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, 2014, p. 9.

[42] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic *et al.*, "Benchmarking 6dof outdoor visual localization in changing conditions," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2018.

[43] W. Winterhalter, F. Fleckenstein, B. Steder, L. Spinello, and W. Burgard, "Accurate indoor localization for RGB-D smartphones and tablets given 2D floor plans," in *International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[44] T. Naseer, G. Oliveira, T. Brox, and W. Burgard, "Semantics-aware visual localization under challenging perceptual conditions," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[45] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous robot navigation in highly populated pedestrian zones," *Journal of Field Robotics*, vol. 32, no. 4, pp. 565–589, 2015.

[46] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 473–480.

[47] A. Valada, L. Spinello, and W. Burgard, "Deep feature learning for acoustics-based terrain classification," in *Proceedings of the International Symposium of Robotics Research*, 2015.

[48] G. Oliveira, A. Valada, C. Bollen, W. Burgard, and T. Brox, "Deep learning for human part discovery in images," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[49] A. Valada, G. L. Olivera, T. Brox, and W. Burgard, "Towards robust semantic segmentation using deep fusion," in *In Proceedings of the Workshop on Limits and Potentials of Deep Learning in Robotics at Robotics: Science and Systems (RSS)*, 2016.

[50] A. Valada, G. Oliveira, T. Brox, and W. Burgard, "Deep multispectral semantic scene understanding of forested environments using multimodal fusion," in *Proceedings of the International Symposium for Experimental Robotics*, 2016.

[51] A. Valada, A. Dhall, and W. Burgard, "Convoluted mixture of deep experts for robust semantic segmentation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop, State Estimation and Terrain Perception for All Terrain Mobile Robots*, 2016.

[52] A. Valada and W. Burgard, "Deep spatiotemporal models for robust proprioceptive terrain classification," *International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1521i–1539, 2017.

[53] A. Valada, J. Vertens, A. Dhall, and W. Burgard, "Adapnet: Adaptive semantic segmentation in adverse environmental conditions," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[54] J. Vertens, A. Valada, and W. Burgard, "Smsnet: Semantic motion segmentation using deep convolutional neural networks," in *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[55] W. Burgard, A. Valada, N. Radwan, T. Naseer, J. Zhang, J. Vertens, O. Mees, A. Eitel, and G. Oliveira, "Perspectives on deep multimodel robot learning," in *Proceedings of the International Symposium of Robotics Research*, 2017.

[56] A. Valada, N. Radwan, and W. Burgard, "Deep auxiliary learning for visual localization and odometry," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[57] A. Valada and W. Burgard, "Learning reliable and scalable representations using multimodal multitask deep learning," in *In Proceedings of RSS Pioneers at Robotics: Science and Systems (RSS)*, 2018.

[58] A. Valada, N. Radwan, and W. Burgard, "Incorporating semantic and geometric priors in deep pose regression," in *In Proceedings of the Workshop on Learning and Inference in Robotics: Integrating Structure, Priors and Models at Robotics: Science and Systems (RSS)*, 2018.

[59] A. Valada, R. Mohan, and W. Burgard, "Self-supervised model adaptation for multimodal semantic segmentation," *arXiv preprint arXiv:1808.03833*, 2018.

[60] F. Boniardi, A. Valada, W. Burgard, and G. D. Tipaldi, "Autonomous indoor robot navigation using sketched maps and routes," in *RSS Workshop on Model Learning for Human-Robot Communication*, 2015.

[61] ——, "Autonomous indoor robot navigation using a sketch interface for drawning maps and routes," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[62] N. Radwan, A. Valada, and W. Burgard, "Multimodal interaction-aware motion prediction for autonomous street crossing," *arXiv preprint arXiv:1808.06887*, 2018.

[63] M. Mittal, A. Valada, and W. Burgard, "Vision-based autonomous landing in catastrophe-struck environments," in *In Proceedings of the Workshop on Vision-based Drones: What's Next? at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[64] B. Blaus, "Medical gallery of blausen medical 2014," *WikiJournal of Medicine*, vol. 1, no. 2, 2014.

[65] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the International Conference in Machine Learning (ICML)*, 2010, pp. 807–814.

[66] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[67] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations (ICLR)*, 2016.

[68] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the International Conference in Machine Learning (ICML)*, 2016, pp. 1050–1059.

[69] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[70] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *arXiv preprint arXiv:1409.4842*, 2014.

[71] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.

[72] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *The European Conference on Computer Vision (ECCV)*, 2016, pp. 630–645.

[73] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, “Stanley: The robot that won the darpa grand challenge,” *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[74] B. Suger, B. Steder, and W. Burgard, “Traversability analysis for mobile robots in outdoor environments: A semi-supervised learning approach based on 3d-lidar data,” in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.

[75] C. A. Brooks and K. Iagnemma, “Vibration-based terrain classification for planetary exploration rovers,” *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1185–1191, Dec 2005.

[76] J. Libby and A. T. Stentz, “Using sound to classify vehicle-terrain interactions in outdoor environments,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2012.

[77] J. Christe and N. Kottege, “Acoustics based terrain classification for legged robots,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[78] M. C. Ozkul, A. Saranli, and Y. Yazicioglu, “Acoustic surface perception from naturally occurring step sounds of a dexterous hexapod robot,” *Mechanical Systems and Signal Processing*, vol. 40, no. 1, pp. 178–193, 2013.

[79] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, “The pothole patrol: Using a mobile sensor network for road surface monitoring,” in *The Sixth Annual International conference on Mobile Systems, Applications and Services (MobiSys 2008)*, Breckenridge, U.S.A., June 2008.

[80] C. Weiss, H. Frohlich, and A. Zell, “Vibration-based terrain classification using support vector machines,” in *International Conference on Intelligent Robots and Systems (IROS)*, Oct 2006, pp. 4429–4434.

[81] G.-Y. Sung, D.-M. Kwak, and J. Lyou, “Neural network based terrain classification using wavelet features,” *Journal of Intelligent & Robotic Systems*, vol. 59, no. 3, pp. 269–281, 2010.

[82] X. Tan and B. Triggs, “Enhanced local texture feature sets for face recognition under difficult lighting conditions,” *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1635–1650, 2010.

[83] S. Otte, S. Laible, R. Hanten, M. Liwicki, and A. Zell, “Robust visual terrain classification with recurrent neural networks,” in *In Proc. of European Symposium on Artificial Neural Networks*, 2015.

[84] A. Santamaria-Navarro, E. H. Teniente, M. Morta, and J. Andrade-Cetto, “Terrain classification in complex three-dimensional outdoor environments,” *Journal of Field Robotics*, vol. 32, no. 1, pp. 42–60, 2015.

[85] S. T. Namin, M. Najafi, and L. Petersson, “Multi-view terrain classification using panoramic imagery and lidar,” in *International Conference on Intelligent Robots and Systems (IROS)*, Sept 2014, pp. 4936–4943.

[86] I. Posner, M. Cummins, and P. Newman, “Fast probabilistic labeling of city maps,” *Proceedings of Robotics: Science and Systems IV, Zurich, Switzerland*, 2008.

[87] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, “Learning long-range vision for autonomous off-road driving,” *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, Feb. 2009.

[88] U. A. Muller, L. D. Jackel, Y. LeCun, and B. Flepp, “Real-time adaptive off-road vehicle navigation and terrain classification,” *Proc. SPIE*, vol. 8741, pp. 87 410A–87 410A–19, 2013.

[89] P. A. Plonski, P. Tokekar, and V. Isler, “Energy-efficient path planning for solar-powered mobile robots,” *Journal of Field Robotics*, vol. 30, no. 4, pp. 583–601, 2013.

[90] J. Thiemann, N. Ito, and E. Vincent, “The diverse environments multi-channel acoustic noise database (demand): A database of multichannel environmental noise recordings,” in *21st International Congress on Acoustics*, 2013.

[91] P. Khunarsal, C. Lursinsap, and T. Raicharoen, “Very short time environmental sound classification based on spectrogram pattern matching,” *Information Sciences*, vol. 243, pp. 57 – 74, 2013.

[92] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv preprint arXiv:1207.0580*, 2012.

[93] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[94] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[95] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[96] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[97] B. E. Kingsbury, N. Morgan, and S. Greenberg, "Robust speech recognition using the modulation spectrogram," *Speech communication*, vol. 25, no. 1-3, pp. 117–132, 1998.

[98] P. Boersma and D. Weenink, "Praat: doing phonetics by computer [computer program]," in *Version 5.3.51, retrieved 2 June 2013 from http://www.praat.org/*, 2013.

[99] T. Giannakopoulos, D. Kosmopoulos, A. Aristidou, and S. Theodoridis, *Advances in Artificial Intelligence: 4th Helenic Conference on AI, SETN 2006, Heraklion, Crete, Greece, May 18-20, 2006. Proceedings*. Springer Berlin Heidelberg, 2006, ch. Violence Content Classification Using Audio Features, pp. 502–507.

[100] M. C. Wellman, N. Srour, and D. B. Hillis, "Feature extraction and fusion of acoustic and seismic sensors for target identification," *Proc. SPIE*, vol. 3081, pp. 139–145, 1997.

[101] D. P. W. Ellis, "Classifying music audio with timbral and chroma features," in *8th International Conference on Music Information Retrieval*, 2007.

[102] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, Jul 2002.

[103] B. Verma, *Pattern Recognition Technologies and Applications: Recent Advances: Recent Advances*. IGI Global, 2008.

[104] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[105] M. A. Bartsch and G. H. Wakefield, “Audio thumbnailing of popular music using chroma-based representations,” *IEEE Transactions on multimedia*, vol. 7, no. 1, pp. 96–104, 2005.

[106] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.

[107] W. Zaremba and I. Sutskever, “Learning to execute,” *arxiv preprint arxiv: 1410.4615*, 2014.

[108] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS’10). Society for Artificial Intelligence and Statistics*, 2010.

[109] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[110] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[111] C.-C. Chang and C.-J. Lin, “Libsvm: a library for support vector machines,” *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[112] P. C. Loizou, *Speech Enhancement: Theory and Practice*, ser. Signal processing and communications. CRC press, 2007.

[113] L. Ojeda, J. Borenstein, G. Witus, and R. Karlsen, “Terrain characterization and classification with a mobile robot,” *Journal of Field Robotics*, vol. 23, no. 2, pp. 103–122, 2006.

[114] E. Trautmann and L. Ray, “Mobility characterization for autonomous mobile robots using machine learning,” *Autonomous Robots*, vol. 30, no. 4, pp. 369–383, 2011.

[115] M. A. Hoepflinger, C. D. Remy, M. Hutter, L. Spinello, and R. Siegwart, “Haptic terrain classification for legged robots,” in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 2828–2833.

[116] G. Best, P. Moghadam, N. Kottege, and L. Kleeman, “Terrain classification using a hexapod robot,” in *Australasian Conference on Robotics and Automation*, 2013.

[117] C. A. Brooks and K. D. Iagnemma, “Self-supervised classification for planetary rover terrain sensing,” in *Aerospace Conference, 2007 IEEE*, March 2007, pp. 1–9.

[118] R. S. Durst and E. P. Krotkov, “Object classification from analysis of impact acoustics,” in *Intelligent Robots and Systems 95.’Human Robot Interaction and Cooperative Robots’, Proceedings. 1995 IEEE/RSJ International Conference on*, vol. 1, 1995, pp. 90–95.

[119] L. Fei-Fei, C. Koch, A. Iyer, and P. Perona, “What do we see when we glance at a scene?” *Journal of Vision*, vol. 4, no. 8, pp. 863–863, 2004.

[120] Y. Xiang and D. Fox, “Da-rnn: Semantic mapping with data associated recurrent neural networks,” *arXiv preprint arXiv:1703.03098*, 2017.

[121] N. Audebert, B. Le Saux, and S. Lefèvre, “Beyond rgb: Very high resolution urban remote sensing with multimodal deep networks,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 140, pp. 20–32, 2018.

[122] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234–241.

[123] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, “Largescale image retrieval with attentive deep local features,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 3456–3465.

[124] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from rgbd images,” in *The European Conference on Computer Vision (ECCV)*, 2012, pp. 746–760.

[125] Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler, “segdeepm: Exploiting segmentation and context in deep neural networks for object detection,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4703–4711.

[126] A. Ošep, A. Hermans, F. Engelmann, D. Klostermann, M. Mathias, and B. Leibe, “Multi-scale object candidates for generic object tracking in street scenes,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3180–3187.

[127] H. Zhang, A. Geiger, and R. Urtasun, “Understanding high-level semantics by modeling traffic patterns,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 3056–3063.

[128] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, "Joint semantic segmentation and 3d reconstruction from monocular video," in *The European Conference on Computer Vision (ECCV)*, 2014, pp. 703–718.

[129] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 2–23, 2009.

[130] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical crfs for object class image segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 739–746.

[131] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *The IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 1–8.

[132] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[133] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3640–3649.

[134] G. Papandreou, I. Kokkinos, and P.-A. Savalle, "Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 390–399.

[135] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[136] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv: 1511.00561*, 2015.

[137] G. L. Oliveira, C. Bollen, W. Burgard, and T. Brox, "Efficient and robust deep networks for semantic segmentation," *International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 472–491, 2018.

[138] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *arxiv preprint arXiv: 1606.00915*, 2016.

[139] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Hypercolumns for object segmentation and fine-grained localization,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 447–456.

[140] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, “Pruning convolutional neural networks for resource efficient inference,” *International Conference on Learning Representations (ICLR)*, 2017.

[141] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, “Learning efficient convolutional networks through network slimming,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[142] S. Anwar, K. Hwang, and W. Sung, “Structured pruning of deep convolutional neural networks,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 3, p. 32, 2017.

[143] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[144] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[145] S. Song, S. P. Lichtenberg, and J. Xiao, “Sun rgb-d: A rgb-d scene understanding benchmark suite.” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 5, 2015, p. 6.

[146] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[147] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[148] C. Liang-Chieh, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *International Conference on Learning Representations (ICLR)*, 2015.

[149] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial Intelligence and Statistics*, 2015, pp. 562–570.

[150] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems*, 1990, pp. 598–605.

[151] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2074–2082.

[152] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *International Conference on Learning Representations (ICLR)*, 2017.

[153] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[154] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[155] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[156] A. Huete, C. Justice, and W. Van Leeuwen, "Modis vegetation index (mod13)," *Algorithm theoretical basis document*, vol. 3, p. 213, 1999.

[157] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2005, pp. 807–814.

[158] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5162–5170.

[159] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden,

M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.

[160] J. Ku, A. Harakeh, and S. L. Waslander, "In defense of classical image processing: Fast depth completion on the cpu," *arXiv preprint arXiv:1802.00036*, 2018.

[161] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *The European Conference on Computer Vision (ECCV)*, 2014.

[162] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: incorporating depth into semantic segmentation via fusion-based cnn architecture," in *Proceedings of the Asian Conference on Computer Vision*, 2016.

[163] A. Eitel, J. T. Springenberg, L. Spinello, M. A. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in *International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[164] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *The European Conference on Computer Vision (ECCV)*, 2012.

[165] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, "A category-level 3d object dataset: Putting the kinect to work," in *Proceedings of the IEEE International Conference on Consumer Depth Cameras for Computer Vision*, 2013, pp. 141–165.

[166] J. Xiao, A. Owens, and A. Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *The IEEE International Conference on Computer Vision (ICCV)*, 2013.

[167] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[168] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," *arXiv preprint arXiv: 1506.04579*, 2015.

[169] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1520–1528.

[170] S. R. Bulò, L. Porzi, and P. Kontschieder, "In-place activated batchnorm for memory-optimized training of dnns," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[171] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," *arXiv preprint arXiv:1802.02611*, 2018.

[172] L.-C. Chen, M. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, and J. Shlens, "Searching for efficient multi-scale architectures for dense image prediction," in *Advances in Neural Information Processing Systems*, 2018, pp. 8713–8724.

[173] Y. Zhuang, F. Yang, L. Tao, C. Ma, Z. Zhang, Y. Li, H. Jia, X. Xie, and W. Gao, "Dense relation network: Learning consistent and context-aware representation for semantic image segmentation," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 3698–3702.

[174] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.

[175] A. Dai and M. Nießner, "3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation," *arXiv preprint arXiv:1803.10409*, 2018.

[176] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," *arXiv preprint arXiv:1412.6856*, 2014.

[177] P. Kohli, L. Ladicky, and P. H. Torr, "Robust higher order potentials for enforcing label consistency," *International Journal of Computer Vision*, vol. 82, no. 3, pp. 302–324, 2009.

[178] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5987–5995.

[179] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *arXiv preprint arXiv:1709.01507*, 2017.

[180] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *arXiv preprint arXiv:1610.02357*, 2016.

[181] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *The IEEE International Conference on Computer Vision (ICCV)*, 2009.

[182] P. Sturgess, K. Alahari, L. Ladicky, and P. H. S. Torr, "Combining Appearance and Structure from Motion Features for Road Scene Understanding," in *Proceedings of the British Machine Vision Conference*, 2009.

[183] J. Shotton, M. Johnson, and R. Cipolla, “Semantic texton forests for image categorization and segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[184] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, “Segmentation and recognition using structure from motion point clouds,” in *The European Conference on Computer Vision (ECCV)*, D. Forsyth, P. Torr, and A. Zisserman, Eds., 2008.

[185] C. Zhang, L. Wang, and R. Yang, “Semantic segmentation of urban scenes using dense depth maps,” in *The European Conference on Computer Vision (ECCV)*, K. Daniilidis, P. Maragos, and N. Paragios, Eds., 2010.

[186] N. Plath, M. Toussaint, and S. Nakajima, “Multi-class image segmentation using conditional random fields and global classification,” in *Proceedings of the International Conference in Machine Learning (ICML)*, 2009.

[187] D. Grangier, L. Bottou, and R. Collobert, “Deep convolutional networks for scene parsing,” in *ICML Workshop on Deep Learning*, 2009.

[188] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Scene parsing with multiscale feature learning, purity trees, and optimal covers,” in *Proceedings of the International Conference in Machine Learning (ICML)*, 2012.

[189] P. O. Pinheiro and R. Collobert, “Recurrent convolutional neural networks for scene labeling,” in *Proceedings of the International Conference in Machine Learning (ICML)*, 2014.

[190] G. Lin, A. Milan, C. Shen, and I. D. Reid, “RefineNet: Multi-path refinement networks for high-resolution semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[191] P. Krähenbühl and V. Koltun, “Efficient inference in fully connected crfs with gaussian edge potentials,” in *Advances in Neural Information Processing Systems*, 2011, pp. 109–117.

[192] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid, “Efficient piecewise training of deep structured models for semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3194–3203.

[193] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, “Conditional random fields as recurrent neural networks,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1529–1537.

[194] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, “Semantic image segmentation via deep parsing network,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1377–1385.

[195] R. Vemulapalli, O. Tuzel, M.-Y. Liu, and R. Chellapa, “Gaussian conditional random field network for semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3224–3233.

[196] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, “Feedforward semantic segmentation with zoom-out features,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3376–3385.

[197] D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 2650–2658.

[198] Z. Wu, C. Shen, and A. v. d. Hengel, “Bridging category-level and instance-level semantic image segmentation,” *arXiv preprint arXiv:1605.06885*, 2016.

[199] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, “Understanding convolution for semantic segmentation,” *arXiv preprint arXiv:1702.08502*, 2017.

[200] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” *arXiv preprint arXiv:1703.06211*, 2017.

[201] L. Schneider, M. Jasch, B. Fröhlich, T. Weber, U. Franke, M. Pollefeys, and M. Rätsch, “Multimodal neural networks: Rgb-d for semantic segmentation and object detection,” in *Image Analysis*, P. Sharma and F. M. Bianchi, Eds., Cham, 2017, pp. 98–109.

[202] R. M. Cichy, D. Pantazis, and A. Oliva, “Similarity-based fusion of meg and fmri reveals spatio-temporal dynamics in human cortex during visual object recognition,” *Cerebral Cortex*, vol. 26, no. 8, pp. 3563–3579, 2016.

[203] S. W. Running, R. Nemani, J. M. Glassy, and P. E. Thornton, “Modis daily photosynthesis (psn) and annual net primary production (npp) product (mod17) algorithm theoretical basis document,” *University of Montana, SCF At-Launch Algorithm ATBD Documents*, 1999.

[204] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," *arXiv preprint arXiv:1710.11063*, 2017.

[205] X. Ren, L. Bo, and D. Fox, "Rgb-(d) scene labeling: Features and algorithms," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[206] L. Bo, X. Ren, and D. Fox, "Unsupervised feature learning for rgb-d based object recognition," in *Experimental Robotics*, 2013, pp. 387–402.

[207] ——, "Hierarchical matching pursuit for image classification: Architecture and fast algorithms," in *Advances in Neural Information Processing Systems*, 2011, pp. 2115–2123.

[208] D. Munoz, J. A. Bagnell, and M. Hebert, "Co-inference for multi-modal scene analysis," in *The European Conference on Computer Vision (ECCV)*, 2012.

[209] A. Hermans, G. Floros, and B. Leibe, "Dense 3d semantic mapping of indoor scenes from rgb-d images," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[210] D.-K. Kim, D. Maturana, M. Uenoyama, and S. Scherer, "Season-invariant semantic segmentation with a deep multimodal network," in *Field and Service Robotics*, 2017.

[211] Z. Li, Y. Gan, X. Liang, Y. Yu, H. Cheng, and L. Lin, "Lstm-cf: Unifying context modeling and fusion with lstms for rgb-d scene labeling," in *The European Conference on Computer Vision (ECCV)*, 2016.

[212] W. Wang and U. Neumann, "Depth-aware cnn for rgb-d segmentation," *arXiv preprint arXiv:1803.06791*, 2018.

[213] C. Couprie, C. Farabet, L. Najman, and Y. LeCun, "Indoor semantic segmentation using depth information," *arXiv preprint arXiv:1301.3572*, 2013.

[214] J. Jiang, L. Zheng, F. Luo, and Z. Zhang, "Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation," *arXiv preprint arXiv:1806.01054*, 2018.

[215] D. Lin, G. Chen, D. Cohen-Or, P.-A. Heng, and H. Huang, "Cascaded feature network for semantic segmentation of rgb-d images," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1320–1328.

[216] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, “3d graph neural networks for rgbd semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5199–5208.

[217] D. Lin, S. Fidler, and R. Urtasun, “Holistic scene understanding for 3d object detection with rgbd cameras,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 1417–1424.

[218] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng, “Convolutional-recursive deep learning for 3d object classification,” in *Advances in Neural Information Processing Systems*, 2012, pp. 656–664.

[219] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, “Adaptive mixtures of local experts,” *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[220] D. Eigen, M. Ranzato, and I. Sutskever, “Learning factored representations in a deep mixture of experts,” *arXiv preprint arXiv:1312.4314*, 2013.

[221] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010.

[222] Y. Cheng, R. Cai, Z. Li, X. Zhao, and K. Huang, “Locality-sensitive deconvolution networks with gated fusion for rgb-d indoor semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 3, 2017.

[223] O. Mees, A. Eitel, and W. Burgard, “Choosing smartly: Adaptive multimodal fusion for object detection in changing environments,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 151–156.

[224] B. K. Horn and B. G. Schunck, “Determining optical flow,” *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[225] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “Flownet 2.0: Evolution of optical flow estimation with deep networks,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017, p. 6.

[226] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, “Three-dimensional scene flow,” in *The IEEE International Conference on Computer Vision (ICCV)*, vol. 2, 1999, pp. 722–729.

[227] E. Ilg, T. Saikia, M. Keuper, and T. Brox, “Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation,” *arXiv preprint arXiv:1808.01838*, 2018.

[228] P. Bideau and E. Learned-Miller, "It's moving! a probabilistic model for causal motion segmentation in moving camera videos," in *European Conference on Computer Vision*, 2016, pp. 433–449.

[229] P. H. Torr, "Geometric motion segmentation and model selection," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 356, no. 1740, pp. 1321–1340, 1998.

[230] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 6, pp. 1187–1200, 2014.

[231] A. Papazoglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *The IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 1777–1784.

[232] D. Reddy, P. Singhal, and M. Krishna, "Semantic motion segmentation using dense crf formulation," in *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, 2014, p. 56.

[233] Q. Fan, Y. Yi, L. Hao, F. Mengyin, and W. Shunting, "Semantic motion segmentation for urban dynamic scene understanding," in *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*, 2016, pp. 497–502.

[234] T.-H. Lin and C.-C. Wang, "Deep learning of spatio-temporal features with geometric-based moving point detection for motion segmentation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3058–3065.

[235] P. Tokmakov, K. Alahari, and C. Schmid, "Learning motion patterns in videos," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 531–539.

[236] K. Fragkiadaki, P. Arbelaez, P. Felsen, and J. Malik, "Learning to segment moving objects in videos," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4083–4090.

[237] N. Haque, D. Reddy, and M. Krishna, "Joint semantic and motion segmentation for dynamic scenes using deep convolutional networks," *arXiv preprint arXiv:1704.08331*, 2017.

[238] R. Gadde, V. Jampani, and P. V. Gehler, "Semantic video cnns through representation warping," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017.

[239] N. Haque, D. Reddy, and M. Krishna, "Kitti semantic ground truth," https://github.com/native93/KITTI-Semantic-Ground-Truth/, 2016.

[240] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," *arXiv preprint arXiv: 1803.06184*, 2018.

[241] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell, "Clockwork convnets for video semantic segmentation," in *The European Conference on Computer Vision (ECCV)*, 2016, pp. 852–868.

[242] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *The European Conference on Computer Vision (ECCV)*, 2016, pp. 597–613.

[243] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4040–4048.

[244] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[245] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *The European Conference on Computer Vision (ECCV)*, Oct. 2012, pp. 611–625.

[246] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *The European Conference on Computer Vision (ECCV)*, 2010.

[247] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[248] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, and A. M. Lopez, "Vision-based offline-online perception paradigm for autonomous driving," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015, pp. 231–238.

[249] P. Xu, F. Davoine, J.-B. Bordes, H. Zhao, and T. Denœux, "Multimodal information fusion for urban scene understanding," *Machine Vision and Applications*, vol. 27, no. 3, pp. 331–349, 2016.

[250] M. Siam, H. Mahgoub, M. Zahran, S. Yogamani, M. Jagersand, and A. El-Sallab, "Modnet: Moving object detection network with motion and appearance for autonomous driving," *arXiv preprint arXiv:1709.04821*, 2017.

[251] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 4306–4312.

[252] R. Vidal and S. Sastry, "Optimal segmentation of dynamic scenes from two perspective views," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2003, pp. II–II.

[253] P. Spagnolo, T. Orazio, M. Leo, and A. Distante, "Moving object segmentation by background subtraction and temporal analysis," *Image and Vision Computing*, vol. 24, no. 5, pp. 411–423, 2006.

[254] P. Gao, X. Sun, and W. Wang, "Moving object detection based on kirsch operator combined with optical flow," in *International Conference on Image Analysis and Signal Processing (IASP)*, 2010, pp. 620–624.

[255] M. P. Patel and S. K. Parmar, "Moving object detection with moving background using optic flow," in *Recent Advances and Innovations in Engineering (ICRAIE)*, 2014, pp. 1–6.

[256] C. S. Royden and K. D. Moore, "Use of speed cues in the detection of moving objects by moving observers," *Vision research*, vol. 59, pp. 17–24, 2012.

[257] R. K. Namdev, A. Kundu, K. M. Krishna, and C. Jawahar, "Motion segmentation of multiple objects from a freely moving monocular camera," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 4092–4099.

[258] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 926–932.

[259] A. Wedel, A. Meißner, C. Rabe, U. Franke, and D. Cremers, "Detection and segmentation of independently moving objects from dense scene flow," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2009, pp. 14–27.

[260] J.-Y. Kao, D. Tian, H. Mansour, A. Vetro, and A. Ortega, "Moving object segmentation using depth and optical flow in car driving sequences," in *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 11–15.

[261] W. Choi, C. Pantofaru, and S. Savarese, “A general framework for tracking multiple people from a moving camera,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 7, pp. 1577–1591, 2013.

[262] B. Drayer and T. Brox, “Object detection, tracking, and motion segmentation for object-level video segmentation,” *arXiv preprint arxiv:1608.03066*, 2016.

[263] V. Romero-Cano and J. I. Nieto, “Stereo-based motion detection and tracking from a moving platform,” in *IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 499–504.

[264] T.-H. Lin and C.-C. Wang, “Deep learning of spatio-temporal features with geometric-based moving point detection for motion segmentation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3058–3065.

[265] S. Tourani and K. M. Krishna, “Using in-frame shear constraints for monocular motion segmentation of rigid bodies,” *Journal of Intelligent & Robotic Systems*, vol. 82, no. 2, pp. 237–255, 2016.

[266] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, “Learning video object segmentation from static images,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[267] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, “One-shot video object segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[268] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang, “Segflow: Joint learning for video object segmentation and optical flow,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 686–695.

[269] S. D. Jain, B. Xiong, and K. Grauman, “Fusionseg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[270] T. Chen and S. Lu, “Object-level motion detection from moving cameras,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 11, pp. 2333–2343, 2017.

[271] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, “Netvlad: Cnn architecture for weakly supervised place recognition,” in *Proceedings of the IEEE*

*International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[272] M. Cummins and P. Newman, “Fab-map: Probabilistic localization and mapping in the space of appearance,” *International Journal of Robotics Research*, vol. 27, no. 6, 2008.

[273] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, “On the performance of convnet features for place recognition,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[274] T. Sattler, B. Leibe, and L. Kobbelt, “Efficient & effective prioritized matching for large-scale image-based localization,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pp. 1744–1756, 2017.

[275] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. Torr, “Exploiting uncertainty in regression forests for accurate camera relocalization,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[276] A. Kendall, M. Grimes, and R. Cipolla, “Posenet: A convolutional network for real-time 6-dof camera relocalization,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[277] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, “Image-based localization using lstms for structured feature correlation,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[278] R. Clark, S. Wang, A. Markham, A. Trigoni, and H. Wen, “Vidloc: 6-dof video-clip relocalization,” *arXiv preprint arXiv:1702.06521*, 2017.

[279] N. Kobyshev, H. Riemenschneider, and L. Van Gool, “Matching features correctly through semantic understanding,” in *2nd International Conference on 3D Vision (3DV)*, vol. 1, 2014, pp. 472–479.

[280] G. Singh and J. Košecká, “Semantically guided geo-location and modeling in urban environments,” *Large-Scale Visual Geo-Localization*, 2016.

[281] N. Rader, M. Bausano, and J. E. Richards, “On the nature of the visual-cliff-avoidance response in human infants,” *Child Development*, pp. 61–68, 1980.

[282] A. Kendall and R. Cipolla, “Geometric loss functions for camera pose regression with deep learning,” *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[283] J. Wu, L. Ma, and X. Hu, "Delving deeper into convolutional neural networks for camera relocalization," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[284] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2930–2937.

[285] T. Naseer and W. Burgard, "Deep regression for monocular camera-based 6-dof global localization in outdoor environments," in *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[286] Z. Laskar, I. Melekhov, S. Kalia, and J. Kannala, "Camera relocalization by computing pairwise relative poses," *arXiv preprint arXiv:1707.09733*, 2017.

[287] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[288] A. Nicolai, R. Skeele, C. Eriksen, and G. A. Hollinger, "Deep learning for laser based odometry estimation," in *RSS workshop Limits and Potentials of Deep Learning in Robotics*, 2016.

[289] V. Mohanty, S. Agrawal, S. Datta, A. Ghosh, V. D. Sharma, and D. Chakravarty, "Deepvo: A deep learning approach for monocular visual odometry," *arXiv preprint arXiv:1611.06069*, 2016.

[290] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, "Relative camera pose estimation using convolutional neural networks," *arXiv preprint arXiv:1702.01381*, 2017.

[291] ——, "Image-based localization using hourglass networks," *arXiv preprint arXiv:1703.07971*, 2017.

[292] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "DSAC - differentiable RANSAC for camera localization," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[293] E. Brachmann and C. Rother, "Learning less is more - 6d camera localization via 3d surface regression," *arXiv preprint arXiv:1711.10228*, 2017.

[294] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia, "Multi-task cnn model for attribute prediction," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1949–1959, 2015.

[295] I. Borg and P. Groenen, “Modern multidimensional scaling: theory and applications,” *Journal of Educational Measurement*, vol. 40, no. 3, pp. 277–280, 2003.

[296] R. Caruana, “Multitask learning,” *Machine Learning*, 1997.

[297] B. Yu and I. Lane, “Multi-task deep learning for image understanding,” in *2014 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2014, pp. 37–42.

[298] H. Bilen and A. Vedaldi, “Universal representations: The missing link between faces, text, planktons, and cat breeds,” *arXiv preprint arXiv:1701.07275*, 2017.

[299] B. Jou and S.-F. Chang, “Deep cross residual learning for multitask visual recognition,” in *Proceedings of the 2016 ACM on Multimedia Conference*, 2016, pp. 998–1007.

[300] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, “Outrageously large neural networks: The sparsely-gated mixture-of-experts,” *arXiv preprint arXiv:1701.06538*, 2017.

[301] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, “Multinet: Real-time joint semantic reasoning for autonomous driving,” *arXiv preprint arXiv:1612.07695*, 2016.

[302] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” *arXiv preprint arXiv:1705.07115*, 2017.

[303] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, “Vision-based multitask manipulation for inexpensive robots using end-to-end learning from demonstration,” *arXiv preprint arXiv:1707.02920*, 2017.

[304] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. S. Feris, “Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification.” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[305] R. Kuga, A. Kanezaki, M. Samejima, Y. Sugano, and Y. Matsushita, “Multi-task learning using multi-modal encoderdecoder networks with shared skip connections,” in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2017.

[306] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, “Cross-stitch networks for multitask learning,” *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[307] C. McManus, B. Upcroft, , and P. Newman, “Scene signatures: Localised and point-less features for localisation,” in *Robotics: Science and Systems*, 2014.

[308] T. Sattler, B. Leibe, and L. Kobbelt, “Fast image-based localization using direct 2d-to-3d matching,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2011.

[309] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla, “24/7 place recognition by view synthesis,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[310] T. Sattler, M. Havlena, K. Schindler, and M. Pollefeys, “Large-scale location recognition and the geometric burstiness problem,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[311] Q. Hao, R. Cai, Z. Li, L. Zhang, Y. Pang, and F. Wu, “3d visual phrases for landmark recognition,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[312] S. Choudhary and P. Narayanan, “Visibility probability structure from sfm datasets and applications,” in *The European Conference on Computer Vision (ECCV)*, 2012, pp. 130–143.

[313] Y. Li, N. Snavely, and D. P. Huttenlocher, “Location recognition using prioritized feature matching,” in *The European Conference on Computer Vision (ECCV)*, 2010, pp. 791–804.

[314] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, “Worldwide pose estimation using 3D point clouds,” in *The European Conference on Computer Vision (ECCV)*, 2012.

[315] T. Sattler, M. Havlena, F. Radenovic, K. Schindler, and M. Pollefeys, “Hyperpoints and fine vocabularies for large-scale location recognition,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2102–2110.

[316] J. Wang, H. Zha, and R. Cipolla, “Coarse-to-fine vision-based localization by indexing scale-invariant features,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 2, pp. 413–422, 2006.

[317] M. Donoser and D. Schmalstieg, “Discriminative feature-to-point matching in image-based localization,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[318] K. Konda and R. Memisevic, “Learning visual odometry with a convolutional network,” in *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2015.

[319] X. Yin, X. Wang, X. Du, and Q. Chen, "Scale recovery for monocular visual odometry using depth estimated with deep convolutional neural fields," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5870–5878.

[320] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "Demon: Depth and motion network for learning monocular stereo," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[321] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[322] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki, "Sfm-net: Learning of structure and motion from video," *arXiv preprint arXiv:1704.07804*, 2017.

[323] C. A. Brooks and K. Iagnemma, "Self-supervised terrain classification for planetary surface exploration rovers," *Journal of Field Robotics*, vol. 29, no. 3, pp. 445–468, 2012.

[324] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," *arXiv preprint arXiv:1807.05520*, 2018.

[325] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," *arXiv preprint arXiv:1807.11164*, 2018.

[326] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[327] L.-C. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, and H. Adam, "Masklab: Instance segmentation by refining object detection with semantic and direction features," *arXiv preprint arXiv:1712.04837*, 2017.

[328] J. Uhrig, E. Rehder, B. Fröhlich, U. Franke, and T. Brox, "Box2pix: Single-shot instance segmentation by assigning pixels to object boxes," in *IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[329] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8759–8768.

[330] L. Shao, P. Shah, V. Dwaracherla, and J. Bohg, “Motion-based object segmentation based on dense rgb-d scene flow,” *arXiv preprint arXiv:1804.05195*, 2018.

[331] Z. Yin and J. Shi, “Geonet: Unsupervised learning of dense depth, optical flow and camera pose,” in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2018.

[332] A. Ranjan, V. Jampani, K. Kim, D. Sun, J. Wulff, and M. J. Black, “Adversarial collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation,” *arXiv preprint arXiv:1805.09806*, 2018.

[333] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[334] D. Barnes, W. Maddern, G. Pascoe, and I. Posner, “Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1894–1900.