Argoverse: 3D Tracking and Forecasting with Rich Maps

Ming-Fang Chang* [1,2], John Lambert*[1,3], Patsorn Sangkloy*[1,3], Jagjeet Singh*[1], Sławomir Bąk[1], Andrew Hartnett[1], De Wang[1], Peter Carr[1], Simon Lucey[1,2], Deva Ramanan[1,2], and James Hays[1,3]

[1]Argo AI, [2]Carnegie Mellon University, [3]Georgia Institute of Technology

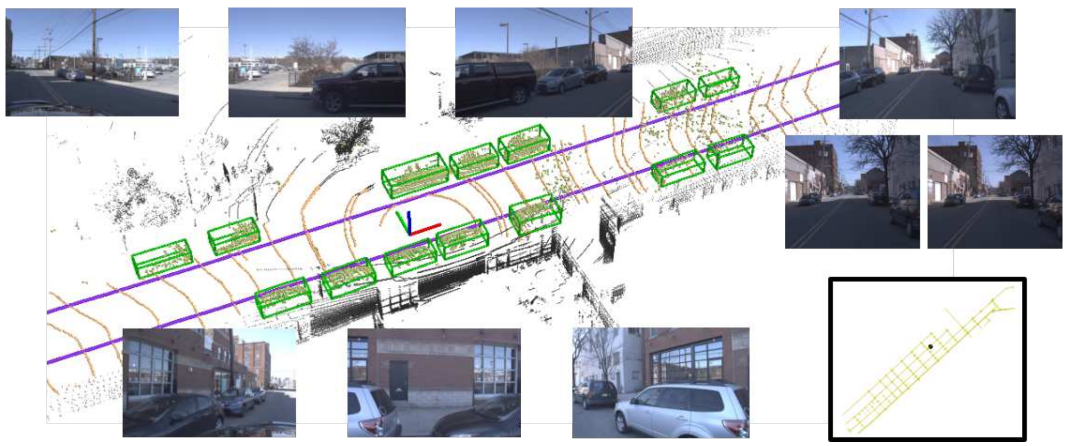

Figure 1: We introduce a dataset for 3D tracking and forecasting with *rich maps* for autonomous driving. Our dataset contains sequences of LiDAR measurements, 360° RGB video, front-facing stereo (middle-right), and 6-dof localization. All sequences are aligned with maps containing lane center lines (magenta), driveable region (orange), and ground height. Sequences are annotated with 3D cuboid tracks (green). A wider map view is shown in the bottom-right.

Abstract

We present Argoverse, *a dataset designed to support autonomous vehicle perception tasks including 3D tracking and motion forecasting. Argoverse includes sensor data collected by a fleet of autonomous vehicles in Pittsburgh and Miami as well as 3D tracking annotations, 300k extracted* interesting *vehicle trajectories, and rich semantic maps. The sensor data consists of 360° images from 7 cameras with overlapping fields of view, forward-facing stereo imagery, 3D point clouds from long range LiDAR, and 6-dof pose. Our 290km of mapped lanes contain rich geometric and semantic metadata which are not currently available in any public dataset. All data is released under a Creative Commons license at Argoverse.org. In baseline experiments, we use map information such as lane direction, driveable area, and ground height to improve the accuracy of 3D object tracking. We use 3D object tracking to "mine" for more than 300k interesting vehicle trajectories to create a trajectory forecasting benchmark. Motion forecasting experiments ranging in complexity from classical methods (k-NN) to LSTMs demonstrate that using detailed "vector maps" with lane-level information substantially reduces prediction error. Our tracking and forecasting experiments represent only a superficial exploration of the potential of rich maps in robotic perception. We hope that Argoverse will enable the research community to explore these problems in greater depth.*

*Equal contribution

1. Introduction

Datasets and benchmarks for a variety of perception tasks in autonomous driving have been hugely influential to the computer vision community over the last few years. We are particularly inspired by the impact KITTI [10] has had in opening new research directions. However, publicly available datasets for autonomous driving rarely include *map* data, even though detailed maps are critical to the development real world autonomous systems. Publicly available maps, e.g. OpenStreetMap, can be useful, but have limited detail and accuracy.

Intuitively, 3D scene understanding would be easier if maps directly told us which 3D points belong to the road, which belong to static buildings, what lane a tracked object is in, what the speed limit for that lane is, how far it is to the next intersection, etc. But since publicly available datasets don't contain such rich mapped attributes it is an open research question of how to represent and utilize these features. Argoverse is the first large autonomous driving dataset with such detailed maps. We examine the potential utility of these new map features on two tasks – 3D tracking and motion forecasting, and we offer a significant amount of real-world, annotated data to enable new benchmarks for these problems.

Our contributions in this paper include:

- We release a large scale dataset with synchronized data from LiDAR, 360° and stereo cameras sampled across two cities and varied conditions.
- We provide ground truth tracking annotation of objects in 3D, with ten times more tracks than the KITTI [10] tracking benchmark.
- We create a large scale forecasting benchmark of trajectories capturing scenarios like turns at intersections, driving with many vehicles nearby, and lane changes.
- We release map data and an API which can be used to develop map-based perception algorithms. To our knowledge, there is no publicly available equivalent to our semantic vector map of road infrastructure and traffic rules.
- We examine the influence of map context in 3D tracking and trajectory forecasting.
- We release the first large-scale dataset suitable for training and benchmarking automatic map creation, often known as *map automation*.
- We release the first fully panoramic, high-frame rate large-scale dataset collected outdoors on a vehicle, opening new possibilities for city-scale reconstruction with photometric-based direct methods.

2. Related Work

Autonomous Driving Datasets with Map Information. Until recently, it was rare to find datasets that provide detailed map information associated with annotated data. Works such as TorontoCity [36] and ApolloScape [18] focus on map construction tasks but without 3D annotation for dynamic objects. The nuScenes dataset [5] contains maps in the form of binary, rasterized, top-down indicators of region of interest (where region of interest is the union of driveable area and sidewalk). This map information is provided for 1000 annotated vehicle log segments (or "scenes") in Singapore and Boston. Like nuScenes, Argoverse includes maps of driveable area, but we also include ground height and a "vector map" of lane centerlines and their connectivity.

Autonomous Driving Datasets with 3D Tracking Annotations. Many existing datasets for object tracking focus on pedestrian tracking from image/video sequences [32, 28, 2]. Several datasets provide raw data from self driving car sensors, but without any object annotations [27, 30, 33]. The ApolloCar3D dataset [34] is oriented toward 3D semantic object keypoint detection instead of tracking. KITTI [10] and H3D [31] offers 3D bounding box and track annotations but does not provide a map and the camera field of view is frontal, rather than 360°. nuScenes [5] currently provides 360° data and a benchmark for 3D object detection, with tracking annotation also available. The Argoverse-Tracking dataset contains 360° track annotations in 3D space aligned with detailed map information. See Table 1 for a comparison between 3D autonomous vehicle datasets.

Autonomous Driving Datasets with Mined Trajectory Data. TrafficPredict [26] also uses sensor-equipped vehicles to observe driving trajectories in the wild and build a forecasting benchmark. The TrafficPredict dataset consists of 155 minutes of observations compared to 320 hours of observations in Argoverse.

Using Maps for Self-driving Tasks. While high definition (HD) maps are widely used by motion planning systems, few works explore the use of this strong prior in perception systems [38] despite the fact that the three winning entries of the 2007 DARPA Urban Challenge relied on a DARPA-supplied map – the *Route Network Definition File* (RNDF) [29, 35, 3]. Hecker *et al.* [13] show that end-to-end route planning can be improved by processing rasterized maps from OpenStreetMap and TomTom. Liang *et al.* [22] demonstrate that using road centerlines and intersection polygons from OpenStreetMap can help infer crosswalk location and direction. Yang *et al.* [38] show that incorporating ground height and road segment into LiDAR points can improve 3D object detection. Suraj *et al.* [25] use dashboard-mounted monocular cameras on a fleet of vehicles to build a 3D map via city-scale structure-from-motion for localization of ego-vehicles and trajectory extraction.

3D Object Tracking. In traditional approaches for point cloud tracking, segments of points can be accumulated using clustering algorithms such as DBSCAN [9, 20] or connected components of an occupancy grid [21, 17], and then associated based on some distance function using the Hungarian algorithm. Held *et al.* utilize probabilistic approaches to point cloud segmentation and tracking [14, 16, 15]. Recent work demonstrates how 3D instance segmentation and 3D motion (in the form of 3D scene flow, or per-point velocity vectors) can be estimated directly on point cloud input with deep networks [37, 23]. Our dataset enables 3D tracking with sensor fusion in a 360° frame.

Trajectory Forecasting: Spatial context and social interactions can influence the future path of pedestrians and cars. Social-LSTM[1] proposes a novel pooling layer to capture social interaction of pedestrians. Social-GAN [11] attempts to model the multimodal nature of the predictions. However, both have only been tested on pedestrian trajectories, with no use of static context (e.g. a map). Deo et al. [8] propose a convolutional social pooling approach wherein

DATASET NAME	MAP TYPE	EXTENT OF ANNOTATED LANES	DRIVEABLE AREA COVERAGE	CAMERA FRAME RATE	360° CAMERAS	INCLUDE STEREO	# TRACKED OBJECTS
KITTI [10]	None	0 km	0 m^2	10 Hz	no	✓	917
Oxford RobotCar [27]	None	0 km	0 m^2	11/16Hz	no	no	0
H3D [31]	None	0 km	0 m^2	30 Hz	no	no	13,763
nuScenes v1.0 [5]	Raster	0 km	1,115,844 m^2	12 Hz	✓	no	64,386
Argoverse-Tracking-Beta (human annotated)	Vector +Raster	204 km (MIA) +86 km (PIT)	1,074,614 m^2	30 Hz	✓	✓	10,572
Argoverse-Forecasting (mined trajectories)	Vector +Raster	204 km (MIA) +86 km (PIT)	1,074,614 m^2	-	no	no	16.4M

Table 1: **Public self-driving datasets**. We compare recent, publicly available self-driving datasets with 3D object annotations for tracking. Coverage area for nuScenes is based on its *road and sidewalk* raster map. Argoverse coverage area is based on our *driveable area* raster map.

they first predict the maneuver and then the trajectory conditioned on that maneuver. In the self-driving domain, the use of spatial context is of utmost importance and it can be efficiently leveraged from the maps. Chen et al. [7] use a feature-driven approach for social and spatial context by mapping the input image to a small number affordances of a road/traffic state. However, they limit their experiments to a simulation environment. IntentNet [6] extends the joint detection and prediction approach of Luo et al. [24] by discretizing the prediction space and attempting to predict one of eight common driving maneuvers. DESIRE [19] demonstrates a forecasting model capturing both social interaction and spatial context. The authors note that the benefits from these two additional components are small on the KITTI dataset, attributing this to the minimal inter-vehicle interactions in the data.

3. The Argoverse Dataset

Our sensor data, maps, and annotations are the *primary contribution* of this work. We also develop an API which helps connect the map data with sensor information e.g. ground point removal, nearest centerline queries, and lane graph connectivity; see Supplemental Material for more details. Our data, annotations, and API are available under a Creative Commons license at *Argoverse.org*.

We collect raw data from a fleet of autonomous vehicles in Pittsburgh, Pennsylvania, USA and Miami, Florida, USA. These cities have distinct climate, architecture, infrastructure, and behavior patterns. The captured data spans different seasons, weather conditions, and times of day. The data used for our dataset traverses nearly 300km of mapped road lanes and comes from a subset of our fleet operating area.

Sensors. Our cars are equipped with two roof-mounted VLP-32 LiDAR sensors with an overlapping 40° vertical field of view and a range of 200m, roughly twice that as the sensors used in nuScenes and KITTI. On average, our LiDAR sensors produce a point cloud at each sweep with three times the density of the LiDAR sweeps in the nuScenes [5] dataset (ours $\sim 107,000$ points vs. nuScenes' $\sim 35,000$ points). The vehicles have 7 high-resolution ring cameras (1920×1200) recording at 30 Hz with overlapped field of view providing 360° coverage. In addition there are 2 front-facing stereo cameras (2056×2464) sampled at 5 Hz. Faces and license plates are procedurally blurred in camera data to maintain privacy. Finally, 6-DOF localization for each timestamp comes from a combination of GPS-based and sensor-based localization. Vehicle localization and maps use a city-specific coordinate system described in more detail in the Supplemental Material. Sensor measurements for particular driving sessions are stored in "logs", and we provide intrinsic and extrinsic calibration data for the LiDAR sensors and all 9 cameras for each log. Figure 2 visualizes our sensor data in 3D. Similar to [33], we place the origin of the vehicle coordinate system at the center of the rear axle. All sensors are roof-mounted, with a LiDAR sensor surrounded by 7 "ring" cameras (clockwise: facing front center, front right, side right, rear right, rear left, side left, and front left) and 2 stereo cameras. Figure 3 visualizes the geometric arrangement of our sensors.

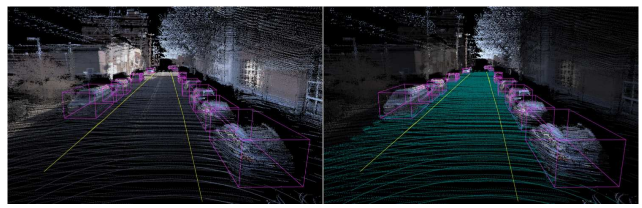

Figure 2: **3D visualization of an Argoverse scene.** Left: we accumulate LiDAR points and project them to a virtual image plane. Right: using our map, LiDAR points beyond driveable area are dimmed and points near the ground are highlighted in cyan. Cuboid object annotations and road center lines are shown in pink and yellow.

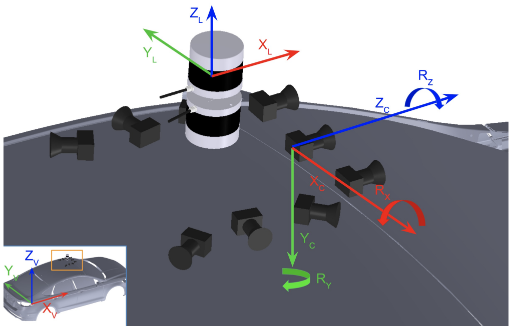

Figure 3: **Car sensor schematic.** Three reference coordinate systems are displayed: (1) the *vehicle frame*, with X_v forward, Y_v left, and Z_v up, (2) the *camera frame*, with X_c across imager, Y_c down imager, and Z_c along optical axis. (3) the *LiDAR* frame, with X_L forward, Y_L left, and Z_L up. Positive rotations R_X, R_Y, R_Z are defined for each coordinate system as rotation about the respective axis following the right-hand rule.

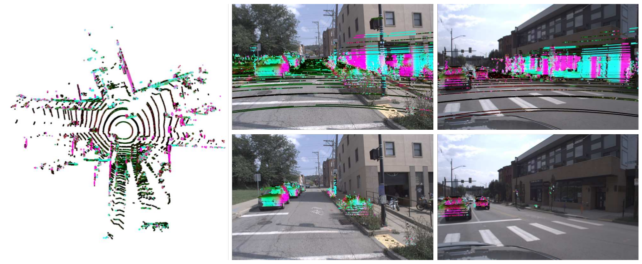

Figure 4: **Map-based ground removal example.** Some Argoverse scenes contain uneven ground, which is challenging to remove with simple heuristics (e.g. assuming ground is planar). Above, the projected LiDAR points are colored by surface normal. The ground surface normal color is non-uniform in the birds-eye-view projection (left). The green color on the slope (middle column) differs from other parts of ground (right column). The lower row uses our map tools to remove ground points and points beyond driveable area.

3.1. Maps

Argoverse contains three distinct maps – (1) a vector map of lane centerlines and their attributes, (2) a rasterized map of ground height, and (3) a rasterized map of driveable area and region of interest (ROI).

Vector Map of Lane Geometry. Our *vector map* consists of semantic road data represented as a localized graph rather than rasterized into discrete samples. The vector map we release is a simplification of the map used in fleet operations. In our vector map, we offer lane centerlines, split into lane segments. We observe that vehicle trajectories generally follow the center of a lane so this is a useful prior for tracking and forecasting.

A lane segment is a segment of road where cars drive in single-file fashion in a single direction. Multiple lane segments may occupy the same physical space (e.g. in an intersection). Turning lanes which allow traffic to flow in either direction would be represented by two different lanes that occupy the same physical space.

For each lane centerline, we provide a number of semantic attributes. These lane attributes describe whether a lane is located within an intersection or has an associated traffic control measure (Boolean values that are not mutually inclusive). Other semantic attributes include the lane's turn direction (*left, right, or none*) and the unique identifiers for the lane's predecessors (lane segments that come before) and successors (lane segments that come after) of which there can be multiple (for merges and splits, respectively). Centerlines are provided as "polylines", i.e. a sequence of straight segments. Each straight segment is defined by 2 vertices: (x, y, z) start and (x, y, z) end. Thus, curved lanes are approximated with a set of straight lines.

We observe that in Miami, lane segments that could be used for route planning are on average 3.84m $\pm$ 0.89 wide. In Pittsburgh, the average width is 3.97m $\pm$ 1.04 in width. Other types of lane segments that would not be suitable for self-driving, e.g. bike lanes, can be as narrow as 0.97m in Miami and as narrow as 1.06m in Pittsburgh.

Rasterized Driveable Area Map. Our maps include binary driveable area labels at 1 meter grid resolution. A driveable area is an area where it is possible for a vehicle to drive (though not necessarily legal). Driveable areas can encompass a road's shoulder in addition to the normal driveable area that is represented by a lane segment. Our track annotations (Section 3.2) extend to 5 meters beyond the driveable area. We call this larger area our *region of interest* (ROI).

Rasterized Ground Height Map. Finally, our maps include real-valued ground height at 1 meter resolution. Knowledge of ground height can be used to remove LiDAR returns on static ground surfaces and thus makes the 3D detection of dynamic objects easier. Figure 4 demonstrates the use of our ground height map to remove LiDAR points on the road.

3.2. 3D Track Annotations

Argoverse-Tracking-Beta[1] contains 100 vehicle log segments with human-annotated data 3D tracks. These 100 segments vary in length from 15 to 60 seconds and collectively contain 10,572 tracked objects. We compare this to other datasets in Table 1. For each log segment, we annotate all objects of interest (both dynamic and static) with bounding cuboids which follow the 3D LiDAR returns associated with each object over time. We only annotate objects within 5 meters of the *driveable area* as defined by our map. For objects that are not visible for the entire segment duration, tracks are instantiated as soon as the object becomes visible in the LiDAR point cloud and tracks are terminated

[1]We refer to our tracking data as *beta* in anticipation of minor refinements or expansion to this dataset before final benchmark release.

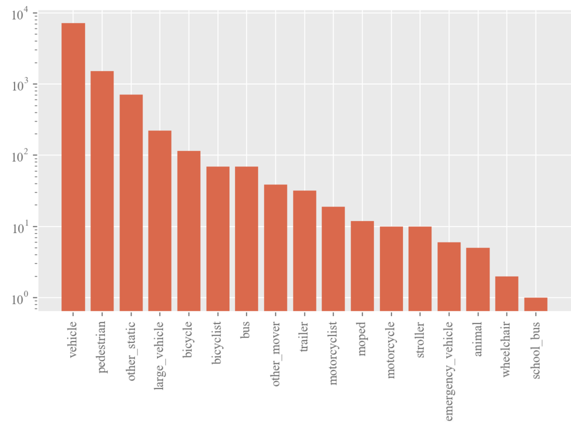

Figure 5: **Distribution of object classes.** This plot shows, in log scale, the number of objects annotated for each class in the 100 log segments in *Argoverse-Tracking-Beta*.

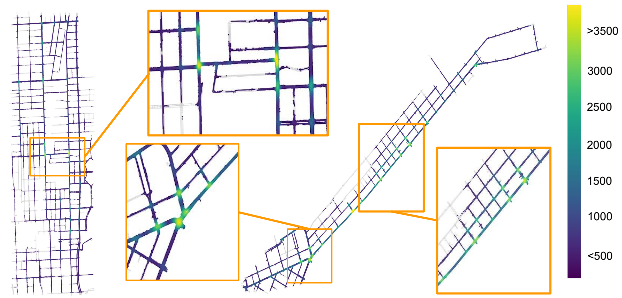

Figure 6: **Distribution of mined trajectories.** The colors indicate the number of mined trajectories across the maps of Miami (left) and Pittsburgh (right). The heuristics to find interesting vehicle behavior lead to higher concentrations in intersections and on busy roads such as Liberty and Penn Ave (southeast roads in bottom right inset).

when the object ceases to be visible. We mark objects as "occluded" whenever they become invisible within the sequence. Each object is labeled with one of 17 categories, including OTHER_STATIC and OTHER_MOVER for static and dynamic objects that do not fit into other predefined categories. More than 70% of tracked objects are vehicles, but we also observe pedestrians, bicycles, mopeds, and more. Figure 5 show the distribution of classes for annotated objects. All track labels pass through a manual quality assurance review process. Figures 1 and 2 show qualitative examples of our human annotated labels. We divide our annotated tracking data into 60 training, 20 validation, and 20 testing sequences.

3.3. Mined Trajectories for Motion Forecasting

We are also interested in studying the task of *motion forecasting* in which we predict the location of a tracked object some time in the future. Motion forecasts can be critical to safe autonomous vehicle motion planning. While our human-annotated 3D tracks are suitable training and test data for motion forecasting, the motion of many of vehicles is relatively uninteresting – in a given frame, most cars are either parked or traveling at nearly constant velocity. Such tracks are hardly a representation of real forecasting challenges. We would like a benchmark with more diverse scenarios e.g. managing an intersection, slowing for a merging vehicle, accelerating after a turn, stopping for a pedestrian on the road, etc. To sample enough of these *interesting* scenarios we track objects from 1006 driving hours across both Miami and Pittsburgh and find vehicles with interesting behavior in 320 of those hours. In particular, we look for vehicles that are either (1) at intersections (2) taking left or right turns (3) changing to adjacent lanes or (4) in dense traffic. In total, we collect 333,441 five second sequences and use them in the forecasting benchmark. Each sequence contains the 2D, birds-eye-view centroid of each tracked object sampled at 10hz. Figure 6 shows the geographic distribution of these sequences. In Section 5, we do not evaluate motion forecasts for pedestrians and stationary vehicles, but still retain their trajectories for context in "social" forecasting models. The 333,441 sequences are split into 211,691 train, 41,146 validation, and 80,604 test sequences. Each sequence has one challenging trajectory which is the focus of our forecasting benchmark. The train, val, and test sequences are taken from disjoint parts of our cities, i.e. roughly one eighth and one quarter of each city is set aside as validation and test data, respectively. This dataset is far larger than what could be mined from publicly available autonomous driving datasets and we have the advantage of using our maps to make it easier to track objects. While data of this scale is appealing because it allows us to see rare behaviors and train complex models, it is too large to exhaustively verify the accuracy of the mined trajectories and thus there is some noise and error inherent in the data.

4. 3D Object Tracking

In this section, we examine how various baseline tracking methods perform on the Argoverse 3D tracking benchmark. Our baseline methods are LiDAR-centric and operate directly in 3D. In addition to measuring the baseline difficulty of our benchmark, we measure how some simple map-based heuristics can influence tracking accuracy. For these baselines, our tracking and evaluation is limited to *vehicles* only.

Given a sequence of F frames, each frame contains set of 3D points from LiDAR $\{P_i \mid i = 1, ..., N\}$, where $P_i \in R^3$ of x, y, z coordinates, we want to determine a set of track hypothesis $\{T_j \mid j = 1, ..., n\}$ where n is the number of unique objects in the whole sequence, and T_j contains the set of object center locations at frames f for $f = \{f_{start}, ..., f_{end}\}$, the range of frames where the object is visible. We usually have a dynamic observer as our car is in motion more often than not. The tracked vehicles in the scene around us can be static or moving.

Our baseline tracking pipeline clusters LiDAR returns to detect potential objects, uses Mask R-CNN [12] to prune non-vehicle LiDAR returns, associates clusters over time using the Hungarian algorithm, estimating transformations between clusters with ICP, and estimates vehicle pose with a Kalman Filter. More details are provided in the Supplemental Material.

The tracker uses the following map attributes:

Driveable area. Since our baseline is focused on vehicle tracking, we constrain our tracker to the driveable area as specified by the map. This covers any region where it is possible for vehicle to drive (see Section 3.1). This reduces the opportunities for false positives.

Ground removal. We use map information to perform ground-removal. In contrast to local ground-plane estimation methods, the map-based approach is effective in sloping and uneven environments.

Lane Direction. Determining the vehicle orientation from LiDAR alone is a challenging task even for humans due to LiDAR sparsity and partial views. We observe that vehicle orientation rarely violates lane direction, especially so outside of intersections. Fortunately, such information is available in our dataset, so we adjust vehicle orientation based on lane direction whenever the vehicle is not at the intersection and contains too few LiDAR points.

4.1. Evaluation

We use standard evaluation metrics commonly used for multiple object trackers (MOT) [28, 4]. The MOT metric relies on a distance/similarity function between ground truth and predicted objects to determine an optimal assignment. Instead of IoU (Intersection-over-Union) which is more commonly used in tracking literature, we use Euclidean distance between object centroids (threshold for missed track at 2.25 meters, which is half of an average family car length in US). We follow the original definition in CLEAR MOT [4] for MOTP (the lower the better). The tracking metrics are explained in the Supplementary Material in detail.

In the experiments, we run our tracker over the 20 logs in the *Argoverse-Tracking-Beta* test set. We are also interested in the relationship between tracking performance and distance. We apply a threshold (30,50,100 m) to the distance between vehicles and our ego vehicle and only evaluate annotations and tracker output within that range. The results in Table 2 show that our tracker performs quite well at short range where the LiDAR sampling density is higher, but struggles for objects beyond 50 meters.

We compare our baseline tracker with three ablations that exclude: 1) Mask R-CNN as pre-filtering for LiDAR 2) lane direction information from the map and 3) map-based ground removal. The results in Table 2 show that Mask-RCNN dramatically improves our detection performance by reducing false positives. Map-based ground removal leads to slightly better detection performance (higher MOTA) than a plane-fitting approach at longer ranges. On the other hand, lane direction from the map doesn't affect our metrics (based on centroid distance), but it helps initialize vehicle direction, as shown in Figure 7.

(a) without lane information (b) with lane information

Figure 7: **Tracking with orientation snapping.** Using lane direction information helps to determine the vehicle orientation for detection and tracking. Ground truth cuboids are green.

We have used relatively simple baselines to track objects in 3D. We believe that our data opens possibilities in map-based and multimodal tracking research.

5. Forecasting

In this section, we describe our pipeline for trajectory forecasting baselines.

1. Preprocessing: As described in Section 3.3, we first mine for "interesting" sequences and then filter out stationary cars from those. Each sequence contains the centroids of tracked objects over 5 seconds.

Forecasting Coordinate System and Normalization. The coordinate system we use for trajectory forecasting is a top-down, bird eye view (BEV). There are three reference coordinate frames of interest to forecasting: (1) The raw trajectory data is stored and evaluated in the *city* coordinate system (See Section 1.1. of the Supp. Material). (2) For models using lane centerlines as a reference path, we define a *2-d curvilinear coordinate system* with axes tangential and perpendicular to the lane centerline. (3) For models without the reference path (without a map), we align everything such that the observed portion of the trajectory starts at the origin and ends somewhere on the positive x axis. If (x_i^t, y_i^t) represent coordinates of trajectory V_i at timestep t, then this makes sure $y_i^{T_{obs}} = 0$, where T_{obs} is last observed timestep of the trajectory (Section 5.1). We find this normalization works better than leaving trajectories in absolute map coordinates or absolute orientations.

2. Feature Engineering: We define additional features to capture social and/or spatial context. For social context, we use minimum distance to the objects in front, in back, and the number of neighbors. Such heuristics are meant to capture the social interaction between vehicles. For spatial

Range threshold	Use Mask-RCNN	Use map lane	Ground removal	MOTA	MOTP	IDF1	MT(%)	ML(%)	# FP	#FN	IDsw	#frag
100 m	Y	Y	map	**37.98**	**0.52**	**0.46**	0.10	0.51	105.40	2455.30	32.55	22.35
	N	Y	map	16.42	0.54	**0.46**	**0.16**	**0.41**	1339.95	**1972.95**	43.30	29.65
	Y	N	map	37.95	**0.52**	**0.46**	0.10	0.51	105.30	2454.85	32.35	22.45
	Y	Y	plane-fitting	37.36	0.53	**0.46**	0.10	0.53	**105.20**	2484.00	**31.10**	**21.25**
50 m	Y	Y	map	**52.74**	**0.52**	**0.58**	0.22	0.29	99.70	1308.25	31.60	21.65
	N	Y	map	21.53	0.54	0.55	**0.38**	**0.18**	1197.30	**897.90**	37.85	24.60
	Y	N	map	52.70	**0.52**	**0.58**	0.22	0.29	99.50	1307.75	31.40	21.75
	Y	Y	plane-fitting	52.05	0.53	**0.58**	0.20	0.31	**98.10**	1335.65	**30.15**	**20.45**
30 m	Y	Y	map	**73.02**	**0.53**	0.73	0.66	0.08	92.80	350.50	19.75	12.80
	N	Y	map	23.28	0.56	0.63	**0.78**	**0.04**	837.45	**238.80**	**19.10**	**11.25**
	Y	N	map	72.99	**0.53**	0.73	0.66	0.09	92.80	349.90	19.65	12.95
	Y	Y	plane-fitting	72.82	**0.53**	**0.74**	0.66	0.09	**92.00**	363.35	19.75	12.85

Table 2: **Tracking accuracy at different ranges.** From top to bottom, accuracy for objects within 100m, 50m, and 30m.

context, we compute everything in the lane segment coordinate system. We compute the lane centerline corresponding to each trajectory and then map (x_i^t, y_i^t) coordinates to distance along the centerline (a_i^t) and offset from the centerline (o_i^t). In the subsequent sections, we denote social features and map features for trajectory V_i at timestep t by s_i^t and m_i^t, respectively.

3. Prediction Algorithm: We implement weighted Nearest Neighbors and LSTM Encoder-Deconder models using different combinations of features. The results are analyzed in Section 5.3.

5.1. Problem Description

The forecasting task is framed as: *given the past input coordinates of a vehicle trajectory V_i as $X_i = (x_i^t, y_i^t)$ for time steps $t = \{1, \ldots, T_{obs}\}$, predict the future coordinates $Y_i = (x_i^t, y_i^t)$ for time steps $\{t = T_{obs+1}, \ldots, T_{pred}\}$*. For a car, 5 seconds is sufficient to capture the required part of trajectory, e.g. crossing an intersection. Furthermore, it is unlikely for a typical driving maneuver to last more than 5 seconds. In this paper, we define the forecasting task as observing 20 past frames (2 seconds) and then predicting 10-30 frames (1-3 seconds) into the future. Each trajectory can leverage the trajectories of other vehicles in the same sequence to capture the social context and map information for spatial context.

5.2. Multimodal Evaluation

Predicting the future is difficult. Often, there are several plausible future actions for a given observation. In the case of autonomous vehicles, it is important to predict *many* plausible outcomes and not simply the *most likely* outcome. While some prior works have evaluated forecasting in a deterministic, unimodal way, we believe a better approach is to follow the evaluation methods of DESIRE [19] and Social GAN [11] and encourage algorithms to output multiple predictions.

Our vector map is a semantic graph. The first step in prediction with a vector map is to localize oneself on the semantic graph. We define two subsequent phases: (1) a *hypothesis phase* and (2) a *generation phase*. The semantic graph makes the generation phase trivial because we can quickly generate hypothesized trajectories via Breadth-First-Search on the semantic graph. However, the hypothesis phase is still challenging due to the multimodal nature of the problem, e.g. it's difficult to know *which* lane segment a vehicle will follow in an intersection.

Among the variety of metrics evaluated in DESIRE was the *oracle error over top K number of samples* metric, where $K = 50$. We follow the same approach and use *top-K* Average Displacement Error (ADE) and Final Displacement Error (FDE) as our metrics. The map-based baselines that we report have access to a semantic vector map. As such, they can generate *K* different hypotheses based on the branching of the road network along a particular observed trajectory. On average, our heuristics generate $K = 5.9$ hypotheses. We generate more than 25 hypotheses for less than 2% of the scenarios. Our map gives us an easy way to produce a compact yet diverse set of forecasts. Other baselines don't have such an option and are restricted to a single prediction. We also provide an *oracle* version of the map-based baselines wherein the model produces the best possible hypothesis by having access to (x_i^t, y_i^t) for $t = \{T_{obs+1}, \ldots, T_{pred}\}$, along with the observed trajectory. Note that an oracle-based hypothesis can still generate an imperfect trajectory, e.g. if a car wasn't following any lane.

5.3. Results

In this section, we evaluate the effect of adding social context and spatial context (from the vector map) to improve trajectory forecasting over horizons of 1 and 3 seconds into the future. We evaluate teh following models:

- *Constant Velocity*: Compute the mean velocity (v_{xi}, v_{yi}) from $t = \{1, \ldots, T_{obs}\}$ and then forecast (x_i^t, y_i^t) for $t = \{T_{obs+1}, \ldots, T_{pred}\}$ using (v_{xi}, v_{yi}) as the constant velocity.
- *NN*: Weighted Nearest Neighbor regression where trajectories are queried by (x_i^t, y_i^t) for $t = \{1, \ldots, T_{obs}\}$.
- *NN+map(oracle)*: Weighted Nearest Neighbor regression where trajectories are queried by (a_i^t, o_i^t) for $t = \{1, \ldots, T_{obs}\}$ obtained from oracle centerline.
- *NN+map*: Similar to *NN+map(oracle)* but uses *top-K*

Baseline	1 second ADE	1 second FDE	3 seconds ADE	3 seconds FDE
Constant Velocity	1.04	1.89	3.55	7.89
NN	0.75	1.28	2.46	5.60
NN+map(oracle)	0.82	1.39	2.39	5.05
NN+map	0.72	1.33	2.28	4.80
LSTM ED	**0.68**	1.78	2.27	5.19
LSTM ED+social	0.69	**1.20**	2.29	5.22
LSTM ED+map(oracle)	0.82	1.38	2.32	4.82
LSTM ED+map	0.80	1.35	**2.25**	**4.67**
LSTM ED+social+map(oracle)	0.89	1.48	2.46	5.09

Table 3: Forecasting Errors for different prediction horizons

hypothesized centerlines.

- *LSTM ED*: LSTM Encoder-Decoder model where the input is (x_i^t, y_i^t) for $t = \{1, \ldots, T_{obs}\}$ and output is (x_i^t, y_i^t) for $t = \{T_{obs+1}, \ldots, T_{pred}\}$
- *LSTM ED+social*: Similar to *LSTM ED* but with input as (x_i^t, y_i^t, s_i^t), where s_i^t denotes social features
- *LSTM ED+map(oracle)*: Similar to *LSTM ED* but with input as (a_i^t, o_i^t, m_i^t) and output as (a_i^t, o_i^t), where m_i^t denotes the map features obtained from oracle centerline. Distances (a_i^t, o_i^t) are then mapped to (x_i^t, y_i^t) for evaluation.
- *LSTM ED+map*: Similar to *LSTM ED+map(oracle)* but uses *top-K* hypothesized centerlines.
- *LSTM ED+social+map (oracle)*: Similar to *LSTM ED+map(oracle)* but with input features being $(a_i^t, o_i^t, s_i^t, m_i^t)$.

The results of these baselines are reported in Table 3. Below, we focus on the ADE and FDE for a prediction horizon of 3 seconds to understand which baselines are less impacted by accumulating errors. *Constant Velocity* is outperformed by all the other baselines because it cannot capture typical driving behaviors like acceleration, deceleration, turns etc. *NN+map* has lower ADE and FDE than *NN* because it is leveraging useful cues from the vector map. *NN+map* has lower error than *NN+map(oracle)* as well, emphasizing the multimodal nature of predictions. *LSTM ED* does better than *NN*. *LSTM ED+social* performs similar to *LSTM ED*, implying that the social context does not add significant value to forecasting. A similar observation was made on KITTI [10] in DESIRE [19], wherein their model with social interaction couldn't outperform the one without it. We observe that *LSTM ED+map* outperforms all the other baselines for a prediction horizon of 3 sec. This proves the importance of having a vector map for distant future prediction and making multimodal predictions. Moreover, *NN+map* has a lower FDE than *LSTM ED+social* and *LSTM ED* for higher prediction horizon (3 secs). This suggests that even a shallow model working on top of a vector map works better than a *deep* model with social features and no vector map. Figure 8 shows qualitative forecasting results from our best performing model.

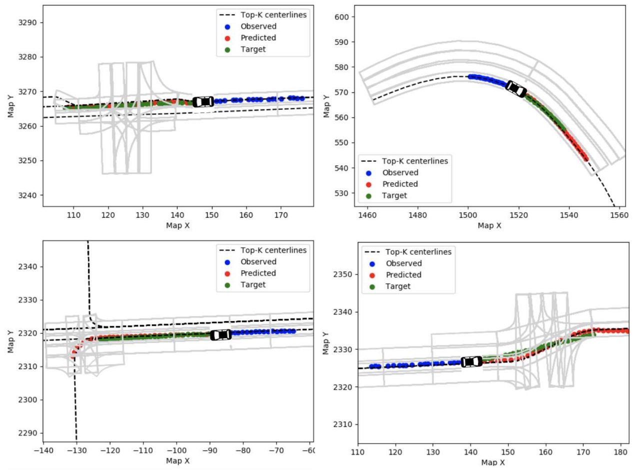

Figure 8: **Qualitative results from *LSTM ED+map* forecasting baseline.** Top left: the model correctly predicts that the car will go straight at the intersection. Top right: the model correctly predicts a smooth right turn never going out of the lane, which might have been difficult if there were no map. Bottom left: demonstration of the multimodal nature of predictions, where the model considers all top-K possibilities. Bottom right: the predictions are on a non-typical lane which takes a slight left and then a slight right. Again, this is hard to predict without a map.

6. Discussion

Argoverse is a large dataset for autonomous driving research. Unique among such datasets, Argoverse contains rich map information such as lane centerlines, ground height, and driveable area. We examine baseline methods for 3D tracking with map-derived context. We also mine one thousand hours of fleet logs to find diverse, real-world object trajectories which constitute our motion forecasting benchmark. We examine baseline forecasting methods and see that map data significantly improves accuracy. We will maintain a public leaderboard for 3D object tracking and motion forecasting. The sensor data, map data, annotations, and code which make up Argoverse are available at *Argoverse.org*.

Acknowledgements. We thank our Argo AI colleagues – Ben Ballard, Brett Browning, Alex Bury, Dave Chekan, Kunal Desai, Patrick Gray, Larry Jackson, Etienne Jacques, Gang Pan, Kevin Player, Peter Rander, Bryan Salesky, Philip Tsai, Ian Volkwein, Ersin Yumer and many more – for their invaluable assistance in supporting Argoverse.

Patsorn Sangkloy is supported by a a Royal Thai Government Scholarship. James Hays receives research funding from Argo AI, which is developing products related to the research described in this paper. In addition, the author serves as a Staff Scientist to Argo AI. The terms of this arrangement have been reviewed and approved by Georgia Tech in accordance with its conflict of interest policies.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. People-tracking-by-detection and people-detection-by-tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[3] Andrew Bacha, Cheryl Bauman, Ruel Faruque, Michael Fleming, Chris Terwelp, Charles Reinholtz, Dennis Hong, Al Wicks, Thomas Alberi, David Anderson, Stephen Cacciola, Patrick Currier, Aaron Dalton, Jesse Farmer, Jesse Hurdus, Shawn Kimmel, Peter King, Andrew Taylor, David Van Covern, and Mike Webster. Odin: Team victortango's entry in the darpa urban challenge. *J. Field Robot.*, 25(8):467–492, Aug. 2008.

[4] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP J. Image and Video Processing*, 2008.

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

[6] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 947–956. PMLR, 29–31 Oct 2018.

[7] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[8] Nachiket Deo and Mohan M Trivedi. Convolutional social pooling for vehicle trajectory prediction. *arXiv preprint arXiv:1805.06771*, 2018.

[9] Martin Ester, Hans peter Kriegel, JÃűrg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.

[10] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[11] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, Oct 2017.

[13] Simon Hecker, Dengxin Dai, and Luc Van Gool. End-to-end learning of driving models with surround-view cameras and route planners. In *European Conference on Computer Vision (ECCV)*, 2018.

[14] David Held, Devin Guillory, Brice Rebsamen, Sebastian Thrun, and Silvio Savarese. A probabilistic framework for real-time 3d segmentation using spatial, temporal, and semantic cues. In *Proceedings of Robotics: Science and Systems*, 2016.

[15] David Held, Jesse Levinson, and Sebastian Thrun. Precision tracking with sparse 3d and dense color 2d data. In *ICRA*, 2013.

[16] David Held, Jesse Levinson, Sebastian Thrun, and Silvio Savarese. Combining 3d shape, color, and motion for robust anytime tracking. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[17] M. Himmelsbach and H. j. WuÌĹnsche. Lidar-based 3d object perception. In *Proceedings of 1st International Workshop on Cognition for Technical Systems*, 2008.

[18] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *arXiv:1803.06184*, 2018.

[19] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher Bongsoo Choy, Philip H. S. Torr, and Manmohan Krishna Chandraker. DESIRE: distant future prediction in dynamic scenes with interacting agents. *CoRR*, abs/1704.04394, 2017.

[20] John Leonard, Jonathan How, Seth Teller, Mitch Berger, Stefan Campbell, Gaston Fiore, Luke Fletcher, Emilio Frazzoli, Albert Huang, Sertac Karaman, Olivier Koch, Yoshiaki Kuwata, David Moore, Edwin Olson, Steve Peters, Justin Teo, Robert Truax, Matthew Walter, David Barrett, Alexander Epstein, Keoni Maheloni, Katy Moyer, Troy Jones, Ryan Buckley, Matthew Antone, Robert Galejs, Siddhartha Krishnamurthy, and Jonathan Williams. A perception-driven autonomous urban vehicle. *J. Field Robot.*, 25(10):727–774, Oct. 2008.

[21] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Sören Kammel, J. Zico Kolter, Dirk Langer, Oliver Pink, Vaughan R. Pratt, Michael Sokolsky, Ganymed Stanek, David Michael Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: Systems and algorithms. In *IEEE Intelligent Vehicles Symposium (IV), 2011, Baden-Baden, Germany, June 5-9, 2011*, pages 163–168, 2011.

[22] Justin Liang and Raquel Urtasun. End-to-end deep structured models for drawing crosswalks. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[23] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. *arXiv preprint arXiv:1806.01411*, 2019.

[24] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[25] Suraj M S, Hugo Grimmett, Lukas Platinsky, and Peter Ondruska. Visual vehicle tracking through noise and occlusions using crowd-sourced maps. In *Intelligent Robots and Systems (IROS), 2018 IEEE international conference on*, pages 4531–4538. IEEE, 2018.

[26] Yuexin Ma, Xinge Zhu, Sibo Zhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the 33rd National Conference on Artifical Intelligence*, AAAI'19. AAAI Press, 2019.

[27] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford Robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017.

[28] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831.

[29] Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, Doug Johnston, Stefan Klumpp, Dirk Langer, Anthony Levandowski, Jesse Levinson, Julien Marcil, David Orenstein, Johannes Paefgen, Isaac Penny, Anna Petrovskaya, Mike Pflueger, Ganymed Stanek, David Stavens, Antone Vogt, and Sebastian Thrun. Junior: The stanford entry in the urban challenge. *J. Field Robot.*, 25(9):569–597, Sept. 2008.

[30] Gaurav Pandey, James R Mcbride, and Ryan M Eustice. Ford campus vision and lidar data set. *Int. J. Rob. Res.*, 30(13):1543–1552, Nov. 2011.

[31] Abhishek Patil, Srikanth Malla, Haiming Gang, and Yi-Ting Chen. The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes. In *International Conference on Robotics and Automation*, 2019.

[32] Luis Patino, Tom Cane, Alain Vallee, and James Ferryman. Pets 2016: Dataset and challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, 2016.

[33] Akshay Rangesh, Kevan Yuen, Ravi Kumar Satzoda, Rakesh Nattoji Rajaram, Pujitha Gunaratne, and Mohan M. Trivedi. A multimodal, full-surround vehicular testbed for naturalistic studies and benchmarking: Design, calibration and deployment. *CoRR*, abs/1709.07502, 2017.

[34] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving. *CoRR*, abs/1811.12222, 2018.

[35] Christopher Urmson, Joshua Anhalt, J. Andrew (Drew) Bagnell, Christopher R. Baker, Robert E. Bittner, John M. Dolan, David Duggins, David Ferguson, Tugrul Galatali, Hartmut Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas Howard, Alonzo Kelly, David Kohanbash, Maxim Likhachev, Nick Miller, Kevin Peterson, Raj Rajkumar, Paul Rybski, Bryan Salesky, Sebastian Scherer, Young-Woo Seo, Reid Simmons, Sanjiv Singh, Jarrod M. Snider, Anthony (Tony) Stentz, William (Red) L. Whittaker, and Jason Ziglar. Tartan racing: A multi-modal approach to the darpa urban challenge. Technical report, Carnegie Mellon University, Pittsburgh, PA, April 2007.

[36] Shenlong Wang, Min Bai, Gellert Mattyus, Hang Chu, Wenjie Luo, Bin Yang, Justin Liang, Joel Cheverie, Sanja Fidler, and Raquel Urtasun. Torontocity: Seeing the world with a million eyes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[37] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *CVPR*, 2018.

[38] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 146–155. PMLR, 29–31 Oct 2018.