Assignment-Space-based Multi-Object Tracking and Segmentation

Anwesa Choudhuri, Girish Chowdhary, Alexander G. Schwing
University of Illinois at Urbana-Champaign
{anwesac2,girishc,aschwing}@illinois.edu

Abstract

Multi-object tracking and segmentation (MOTS) is important for understanding dynamic scenes in video data. Existing methods perform well on multi-object detection and segmentation for independent video frames, but tracking of objects over time remains a challenge. MOTS methods formulate tracking locally, i.e.*, frame-by-frame, leading to sub-optimal results. Classical global methods on tracking operate directly on object detections, which leads to a combinatorial growth in the detection space. In contrast, we formulate a global method for MOTS over the space of assignments rather than detections: First, we find all top-k assignments of objects detected and segmented between any two consecutive frames and develop a structured prediction formulation to score assignment sequences across any number of consecutive frames. We use dynamic programming to find the global optimizer of this formulation in polynomial time. Second, we connect objects which reappear after having been out of view for some time. For this we formulate an assignment problem. On the challenging KITTI-MOTS and MOTSChallenge datasets, this achieves state-of-the-art results among methods which don't use depth data.*

1. Introduction

Multi-Object Tracking and Segmentation (MOTS) not only requires to detect and segment objects in a video, but also asks to assign consistent IDs, *i.e.*, each visible instance of the same object is always given the same ID. MOTS is important for understanding scenes in a video and is crucial for autonomous driving, robotics, agricultural and biomedical data analysis, *etc*. While the MOTS sub-task of multi-object detection and segmentation on individual video frames has received a considerable amount of attention [14, 35, 9, 13, 42, 40, 17, 18, 19, 16], tracking of objects over multiple frames remains a challenge, especially in the presence of occlusions and viewpoint variations.

Tracking of objects over a sequence has been addressed using batch and online-methods. The former assumes an entire sequence is available, while the latter operates frame-by-frame. These methods face two main challenges. Firstly, current MOTS methods (batch or online) formulate tracking locally. We note that accurate global tracking is important to understand complex environments, as it ensures consistency in preserving identities of objects over a long period. Classical global batch methods for multi-object tracking (MOT) have been proposed in the past [20, 3, 59, 4, 49, 2, 55]. These formulations face the second challenge: Tracking is formulated directly on object detections [20, 3, 59, 4, 49, 2, 55], which leads to a combinatorial growth of options. We note that directly formulating tracking on detections seems appealing because those are the objects of interest. However, this form of data representation also complicates optimization because one needs to solve for the best path for each object in the video. Note that the number of objects is generally unknown.

To address both challenges, we propose to *formulate tracking over the space of assignments rather than object detections*. For this, we first find the top-k assignments of object detections (and segmentations) between any two consecutive frames. This is efficiently doable using the Hungarian-Murty algorithm [34]. We then develop a structured prediction formulation which globally scores an *assignment sequence* rather than a detection sequence. By finding the global optimizer of the structured formulation in polynomial time using dynamic programming we can address tracking in MOTS. We jointly learn the tracking parameters and the detection/segmentation network. Further, to establish long term connections, we introduce a post-processing step which associates objects over longer time intervals. This step uses an assignment problem to construct long-term connections between previously unassigned object detections and detection sequences.

On the challenging KITTI-MOTS and MOTSChallenge datasets [54], the method achieves state of the art results on MOTS when compared to other 2D methods, *i.e.*, TrackR-CNN [54] and PointTrack [58]. On the KITTI-MOTS test data, we improve upon PointTrack [58] (the next best 2D method) by 14% (car) and 9% (pedestrian) in association (AssA) and 8% (car) and 4% (pedestrian) overall (HOTA). On the MOTSChallenge test data, we improve upon PointTrack [58] by 5% on sMOTSA. A qualitative comparison with PointTrack [58] is shown in Fig. 1. We also improve upon MOTSFusion [27], a 3D method requiring depth information, on pedestrian tracking on the KITTI-MOTS test data (8% improvement in association (AssA), 5% improve-

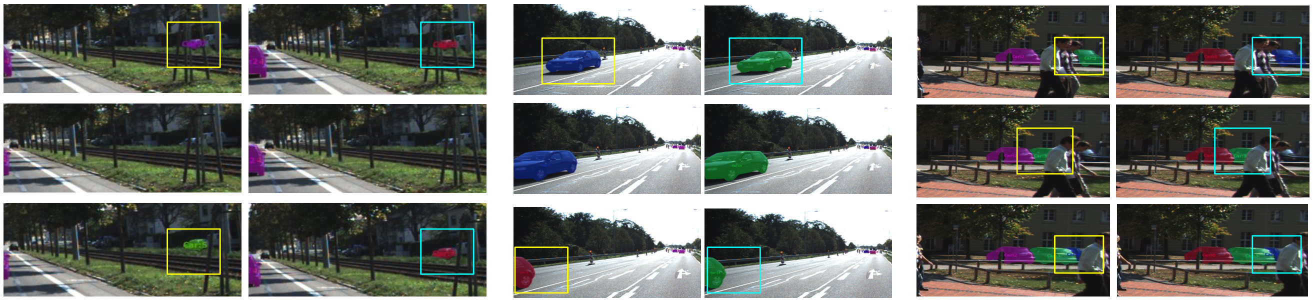

Figure 1. Use of assignment space better preserves identities of objects when compared to PointTrack [58]. We highlight the identity switches from PointTrack using yellow rectangles. Our method is able to recover those identities (highlighted using cyan rectangles). For example, in the last 2 columns (right-most example), Point track wrongly identifies two different cars to be the same (row 1 and 2). The mismatch is continued forward (row 3).

ment overall (HOTA)). Quantitative results are summarized in Tab. 1, Tab. 2 and Tab. 6. To establish generality, we also study our approach on the MOT task and compare with other MOT batch methods (Tab. 7).

2. Related Work

2.1. Multi-Object Tracking

Due to recent progress in object detection [13, 42, 40], *tracking-by-detection* has become a leading paradigm in multiple object tracking (MOT) [56]. MOT methods perform tracking on detected bounding boxes, and are broadly partitioned into two categories: batch-methods and online-methods. We review both of them next.

Batch methods. Batch methods assume all frames in a sequence to be available, and globally solve for object tracks. Hierarchical track association [20], global trajectory optimization via dynamic programming [3] and network flow [59, 4, 49, 2, 55] have been proposed in the past. These approaches work directly on object detections. While elegant solutions were formulated to work with a large number of detections, some of the approaches were limited to tracking only a few objects. For instance, Berclaz *et al.* [3] describe a min-cost flow approach, that tracks up to 6 people in a video. In a typical MOTS task, 100s of objects may be present. Multiple Hypotheses Tracking (MHT) [41] and Joint Probabilistic Data Association (JPDA) [11] were used in classical batch tracking [57, 47, 38]. But they require highly efficient pruning of the detection search space to reduce the combinatoric growth. For instance, Kim *et al.* [21] proposed a modern formulation of a classical MHT, utilizing rich appearance features of objects from deep-nets. Rezatofighi *et al.* [43] proposed a computationally tractable approximation to the original JPDA algorithm and achieved SOTA performance on tracking.

Online methods. Online MOT methods rely on past video frames to estimate the current state [25, 53, 20, 46, 37, 48]. For example, DeepSORT [56] uses motion estimates from a Kalman filter and appearance features from a convolutional neural net to link detections locally over time. Luo *et al.* [30] perform end-to-end detection and tracking in 3D.

2.2. Multi-Object Tracking and Segmentation

Deep segmentation networks [10, 26, 45, 14, 9, 35] perform remarkably well in image segmentation tasks. Recently mask-based tracking is gaining popularity as it is often more robust than tracking of bounding boxes [54].

Mask-based tracking on the KITTI dataset was performed using stereo information [37]. Alternatively, joint tracking and segmentation of objects via conditional random fields was also discussed [32]. While many of these methods perform well on specific tasks, their performance could not be evaluated comprehensively previously, due to the lack of MOTS spcific datasets [54]. Voigtlaender *et al.* [54] created the KITTI-MOTS and MOTSChallenge datasets, provide new metrics to evaluate the MOTS task and also proposed the TrackRCNN baseline. In TrackR-CNN [54], MaskRCNN [14] is augmented via 3D convolutional filters to generate temporally enhanced features. Tracking is performed by an association head, that is trained to re-identify objects. However, the approach is often prone to ID switches. MOTSFusion [27] is a batch method, that addresses the task of tracking in 2 steps. First, frame-by-frame local assignments of objects are obtained using the Hungarian algorithm [33], which generates short tracks. Following this, 3D reconstruction of objects is performed to join the short tracks based on motion consistencies. MOTS-Fusion [27] depends on multiple data modalities like camera egomotion and depth maps, which may not be available for all datasets. For example, MOTSFusion cannot be tested on the MOTSChallenge dataset [54] which lacks such modalities. The recently introduced PointTrack [58] learns instance embeddings on segments by considering image pixels as unordered 2D point clouds. Tracking is performed locally using the Hungarian algorithm [33].

In prior MOTS works [27, 58], the Hungarian algorithm [33] is used to get local frame-by-frame assignments of detections. However, a set of best local assignments calculated on consecutive frames is not necessarily globally optimal over an entire video. This sets the stage for a dynamic pro-

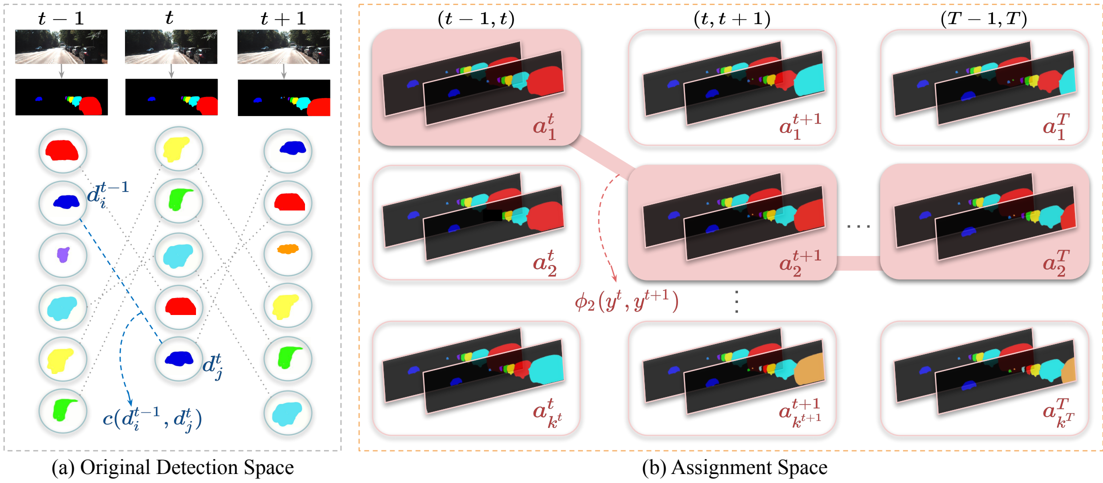

Figure 2. Overview of our approach (Sec. 3). **(a)** shows the detection space $\mathcal{D} = \{\mathcal{D}^t\}_{t=1}^T$ where $\mathcal{D}^t = \{d_1^t, d_2^t, \dots\}$ represents the set of all detections at frame t (Sec. 3.1). We represent detections as nodes. Note that by detections, here, we mean both bounding boxes and segmentations. $c(d_i^{t-1}, d_j^t)$ is the cost of assigning detection d_i^{t-1} to detection d_j^t. **(b)** shows the assignment space $\mathcal{A} = \{\mathcal{A}^t\}$ (Sec. 3.1), where $\mathcal{A}^t = \{a_1^t, a_2^t, \dots, a_{k^t}^t\}$ denotes the set of k^t best assignments for frame-pair $(t-1, t)$, ordered by ascending cost. The global minimum cost path is highlighted in light red (Sec. 3.2). As described in Sec. 3.1, $\phi_2(y^t, y^{t+1})$ denotes the edge cost between assignments $a_{y^t}^t$ and $a_{y^{t+1}}^{t+1}$. The node cost of each assignment node is represented by $\phi_1(y^t)$ (not shown in figure for simplicity).

gramming based approach for MOTS which optimizes for the best track for each object in a video. However, as mentioned earlier, global optimization over the space of object detections is difficult if the number of objects is large and unknown.

Different from the aforementioned approaches, we propose a batch method for MOTS by formulating a structured prediction that directly infers the assignment of detected and segmented objects globally across video frames. We argue that it is much simpler to operate over a pruned space of assignments rather than over the space of detections: this formulation requires to only find a single optimal path.

2.3. Structured Prediction

Structured prediction is particularly suitable for joint prediction of multiple variables: it permits to exploit the dependencies between the involved variables while explicitly modeling their correlations. Structured prediction has been used in numerous computer vision applications like segmentation [1], localization [7, 24], pixel-wise classification [51], *etc*. In [50, 55, 23] a structured prediction-based approach was adopted for the task of MOT. In this work, we use structured prediction for the MOTS task.

3. Structured Prediction using Assignments

The goal of multi-object tracking and segmentation is to detect and segment objects in individual frames and track the detections and segmentations of the same object across frames of a given input sequence. An overview of our two-step batch approach is given in Fig. 2 (step 1) and Fig. 3 (step 2). Step 1 detects, segments and tracks objects that are missed for at most one frame via an end-to-end trained assignment-based formulation. Step 2 links tracks obtained from step 1 that were interrupted for more than one frame, *e.g.*, due to occlusions. As current deep-nets perform multi-object detection and segmentation well [14, 9, 35], we focus on the tracking aspects of the MOTS task and follow prior work w.r.t. detection and segmentation.

Overview. As shown in Fig. 2, after a deep-net yields detections and segmentations for T frames, we compute the top-k (the k best) detection assignments between the $T-1$ consecutive frame pairs using the Hungarian-Murty algorithm [34]. Subsequently, we formulate a structured prediction over the assignment space (k assignments for $T-1$ consecutive frame pairs) as discussed in Sec. 3.1. To solve this formulation and obtain a globally optimal minimum-cost path, we use dynamic programming. This is discussed in Sec. 3.2. We describe end-to-end learning of the parameters of the structured prediction formulation and the detection/segmentation deep-net in Sec. 3.3. Finally and as illustrated in Fig. 3, we uncover long-range assignments in a post-processing step via an assignment formulation over the space of tracks from step 1. See Sec. 3.4 for details.

3.1. Assignment-based Formulation

Assume a deep-net yields a set of detections $\mathcal{D}^t = \{d_1^t, d_2^t, \dots\}$ for every frame t in a video. Fig. 2a uses nodes to illustrate the detections obtained in all T frames, *i.e.*, $\mathcal{D} = \{\mathcal{D}^t\}\ \forall t \in \{1, \dots, T\}$. Each detection is linked to a set of node attributes, which includes the corresponding segmentation mask, an appearance feature vector, the video frame and the optical flow computed between the previous frame and the current frame in the video.

We first construct the assignment space using the k best assignments for the $T-1$ consecutive frame pairs. Then we devise a cost function for a sequence of assignments.

Constructing assignment space. Formally, we use the matrix $a^t \in \{0,1\}^{|\mathcal{D}^{t-1}|\times|\mathcal{D}^t|}$ to represent an assignment between detections $\mathcal{D}^{t-1}$ at time $t-1$ and detections $\mathcal{D}^t$ at time t. Specifically, $a^t(i,j) = 1$ indicates that detection d_i^{t-1} is assigned to detection d_j^t. The row sum and column sum of a^t are enforced to equal 1. To enable that detections don't always have to be assigned, we introduce auxilliary detections, representing 'no assignment.' Since it is hard to optimize over all possible assignment matrices between all consecutive frame pairs, we first reduce the space of assignments between consecutive frames. For this we find the k^t best assignments for every pair $(t-1,t)$ of consecutive frames from the set of $(\max\{|\mathcal{D}^{t-1}|,|\mathcal{D}^t|\}+1)\cdot\ldots\cdot(\max\{|\mathcal{D}^{t-1}|,|\mathcal{D}^t|\}-\min\{|\mathcal{D}^{t-1}|,|\mathcal{D}^t|\}+1)$ possible assignments. This is efficiently possible using the Hungarian-Murty algorithm [34]. We use $\mathcal{Y}^t = \{1,\ldots,k^t\}$ to represent the indices of the top k^t best local assignments between frame pair $(t-1,t)$. In our case k^t is chosen to equal the minimum of 20 and the number of possible assignments. Every $y^t \in \mathcal{Y}^t$ refers to one assignment matrix $a^t_{y^t}$. *E.g.*, $y^t = 1$ refers to the best local assignment

$$a_1^t = \arg\min_{a^t\in\mathcal{A}^t} \sum_{d_i^{t-1}\in\mathcal{D}^{t-1}, d_j^t\in\mathcal{D}^t} a^t(i,j)c(d_i^{t-1},d_j^t),$$

which can also be obtained using the Hungarian algorithm [33], while $y^t = 2$ refers to the second best assignment a_2^t, *etc*. The constraint set $\mathcal{A}^t$ ensures that a^t is a valid assignment by enforcing that the row sum and the column sum of a^t are equal to 1. One might argue that the actual assignment space is much larger, and pruning the local space to the 20 best assignments is sub-optimal. Empirically we find that any $k^t \geq 15$ is a reasonable choice. Our cost function is well optimized, so that the optimal assignment lies within the first 20 best local assignments. (This is demonstrated with the help of Tab. 4 and Fig. 4.)

The cost of assigning detection d_i^{t-1} to detection d_j^t is given by

$$\begin{aligned} c(d_i^{t-1},d_j^t) = \lambda_{\text{iou}} f_{\text{iou}}(d_i^{t-1},d_j^t) + \\ \lambda_{\text{app}} f_{\text{app}}(d_i^{t-1},d_j^t) + \lambda_{\text{dist}} f_{\text{dist}}(d_i^{t-1},d_j^t), \end{aligned} \tag{1}$$

where $f_{\text{iou}}(d_i^{t-1},d_j^t)$ is the intersection over union calculated between the segmentation d_j^t and the segmentation d_i^{t-1} warped to frame t using optical flow. The optical flow is calculated using RAFT [52]. We use $f_{\text{app}}(d_i^{t-1},d_j^t)$ to denote the Euclidean distance between the appearance feature vectors for detections d_i^{t-1} and d_j^t. We get this feature from PointTrack [58]. $f_{\text{dist}}(d_i^{t-1},d_j^t)$ denotes the Euclidean distance between the bounding box centers of the detections d_i^{t-1} and d_j^t. The parameters λ_{iou}, λ_{app} and λ_{dist} are trainable. We discuss learning in Sec. 3.3.

Assignment sequence cost. Fig. 2b represents the assignment space discussed above. Each possible assignment $a^t_{y^t}$ with $y^t \in \mathcal{Y}^t$ is illustrated via a plate. Note that each plate, *i.e.*, each assignment between two consecutive frames, has a *unary* cost when being selected:

$$\phi_1^t(y^t) = \sum_{d_i^{t-1}\in\mathcal{D}^{t-1}, d_j^t\in\mathcal{D}^t} a^t_{y^t}(i,j)c(d_i^{t-1},d_j^t). \tag{2}$$

Intuitively, we accumulate the costs of all assignments (d_i^{t-1},d_j^t) that are indicated by assignment matrix $a^t_{y^t}$. Moreover, a pair of consecutive plates (y^t,y^{t+1}) representing frame pairs $(t-1,t)$ and $(t,t+1)$ has a *pairwise* cost

$$\phi_2^t(y^t,y^{t+1}) = \sum_{d_i^{t-1}\in\mathcal{D}^{t-1}, d_s^t\in\mathcal{D}^{t+1}} a^{t,t+1}_{y^t,y^{t+1}}(i,s)c_2(d_i^{t-1},d_s^{t+1}). \tag{3}$$

Here, matrix $a^{t,t+1}_{y^t,y^{t+1}} = a^t_{y^t}\cdot a^{t+1}_{y^{t+1}}$ represents the local track of assignments from time $t-1$ to time $t+1$. Intuitively, if there exists a local track of assignments from detection d_i^{t-1} via d_j^t to d_s^{t+1} given the assignment matrices $a^t_{y^t}$ and $a^{t+1}_{y^{t+1}}$, then we pay cost $c_2(d_i^{t-1},d_s^{t+1})$. This cost is computed via:

$$\begin{aligned} c_2(d_i^{t-1},d_s^{t+1}) = \lambda_{\text{iou},2} f_{\text{iou},2}(d_i^{t-1},d_s^{t+1}) + \\ \lambda_{\text{app},2} f_{\text{app}}(d_i^{t-1},d_s^{t+1}) + \lambda_{\text{dist},2} f_{\text{dist}}(d_i^{t-1},d_s^{t+1}). \end{aligned} \tag{4}$$

Note that in Eq. (4) we refer to the intersection over union using $f_{\text{iou},2}$ as the detections are warped by two frames, *i.e.*, from $t-1$ to $t+1$. The parameters $\lambda_{\text{iou},2}$, $\lambda_{\text{app},2}$ and $\lambda_{\text{dist},2}$ are trained, which is discussed in Sec. 3.3.

We now discuss how to find the optimal sequence of assignments given the costs defined in Eq. (2) and Eq. (3).

3.2. Inference via Dynamic Programming

Given the costs defined in Eq. (2) and Eq. (3), our objective is to find the minimum-cost assignment sequence $y^* = (y^{1,*},\ldots,y^{T,*}) \in \mathcal{Y} = \prod_{t=1}^T \mathcal{Y}^t$, picking exactly one assignment per frame pair. Formally, we address

$$y^* = \arg\min_{y\in\mathcal{Y}} \mathcal{L}(y) \triangleq \sum_{t=1}^{T} \phi_1^t(y^t) + \sum_{t=1}^{T-1} \phi_2(y^t,y^{t+1}). \tag{5}$$

Importantly, because the domain of the program given in Eq. (5) is discrete and because the loss $\mathcal{L}(y)$ only consists of functions which depend on pairs (y^t,y^{t+1}) of successive variables $\forall t$, classical dynamic programming is directly applicable. It yields the global minimizer y^* for the program given in Eq. (5) in polynomial time.

Note, the global minimizer y^* points to an assignment $y^{t,*}$ per frame pair $(t-1,t)$, which points to an assignment matrix $a^t_{y^{t,*}}$, which in turn indicates the chosen assignment between detections $\mathcal{D}^{t-1}$ at time $t-1$ and $\mathcal{D}^t$ at time t.

3.3. Parameter Learning

Note that the program solved during inference, *i.e.*, Eq. (5), depends on tracking parameters $\lambda = [\lambda_{\text{iou}}, \lambda_{\text{app}}, \lambda_{\text{dist}}, \lambda_{\text{iou},2}, \lambda_{\text{app},2}, \lambda_{\text{dist},2}]$. In addition, it also depends on deep-net parameters θ via the detections that are used in the cost functions shown in Eq. (1) and Eq. (4). We didn't make the dependence of detections on θ explicit for readability. We jointly train λ and θ end-to-end.

To derive the learning objective, we follow the learning goal of classical structured prediction: the ground truth configuration $y_{\text{GT}} \in \mathcal{Y}$ should have a lower cost than any other configuration $y \in \mathcal{Y}$. Intuitively, if we find parameters (λ, θ) which achieve this lowest cost for a large number of training samples, we have a reasonable tracker. Hence, following classical structured prediction we use an objective to linearly penalize the trainable parameters whenever our learning goal is not satisfied. Further, this objective permits to back-propagate all the way into the detection-segmentation network. Formally, the learning objective is

$$\min_{\lambda,\theta} \mathcal{R}(\lambda,\theta) + \sum_{y_{\text{GT}}\in\mathcal{T}} \max_{y\in\mathcal{Y}} (\Delta(y, y_{\text{GT}}) - \mathcal{L}(y;\lambda,\theta)) + \mathcal{L}(y_{\text{GT}};\lambda,\theta). \tag{6}$$

Hereby we use $\mathcal{R}(\lambda,\theta)$ to denote a regularizer. In our case, the regularizer consists of the segmentation loss that was used to initially train the deep-net parameters θ. Δ denotes the loss function which compares a configuration y to the ground truth y_{GT} and $\mathcal{T}$ is the training set. Note, for readability we ignore any dependence on data such as images.

We use stochastic gradient descent to update the parameters (λ, θ). The gradients w.r.t. λ are easily computable. The gradients w.r.t. θ are

$$\nabla_\theta \phi_1^t(y^t) = \sum_{d_i^{t-1}\in\mathcal{D}^{t-1}, d_j^t\in\mathcal{D}^t} [a_{y^t}^t(i,j)\nabla_\theta c(d_i^{t-1}, d_j^t) + c(d_i^{t-1}, d_j^t)\nabla_\theta a_{y^t}^t(i,j)], \text{ and} \tag{7}$$

$$\nabla_\theta \phi_2^t(y^t, y^{t+1}) = \sum_{d_i^{t-1}\in\mathcal{D}^{t-1}, d_s^{t+1}\in\mathcal{D}^t} [a_{y^t,y^{t+1}}^{t,t+1}(i,s)\nabla_\theta c_2(d_i^{t-1}, d_s^{t+1}) + c(d_i^{t-1}, d_s^{t+1})\nabla_\theta a_{y^t,y^{t+1}}^{t,t+1}(i,s)]. \tag{8}$$

Note, the gradients $\nabla_\theta a_{y^t}^t$ and $\nabla_\theta a_{y^t,y^{t+1}}^{t,t+1}$ are hard to compute: they require to backprop through an optimization. To simplify we assume no dependence of the assignment matrices on detections, and hence on θ. We verified on synthetic data that this simplification leads to meaningful results while avoiding complex and slow computation.

Loss Function Δ. The loss $\Delta(y, y_{\text{GT}})$ compares the ground truth configuration y_{GT} to any other configuration y. Intuitively, it refers to the margin of separation between the ground truth and other configurations. In our case, we obtain Δ by summing the number of wrong assignments per frame across all frames. Formally, this loss is the squared Frobenius norm applied to the difference of the assignment matrices referred to via y and y_{GT}, *i.e.*,

$$\Delta(y, y_{\text{GT}}) = \frac{1}{2}\sum_{t=2}^{T} \|a_{y^t}^t - a_{y_{\text{GT}}^t}^t\|_F^2. \tag{9}$$

This counts the number of identity switches of the predicted assignments, based on the ground truth assignments. Crucially, the loss given in Eq. (9) decomposes into a sum of local terms across time. Consequently, maximization of $\Delta - \mathcal{L}$ w.r.t. y during learning, as used in Eq. (6), is also possible via dynamic programming.

Training data $\mathcal{T}$. To optimize the training objective given in Eq. (6), our training set is $\mathcal{T} = \{(x, y_{\text{GT}})\}$. Here x is a video clip of T frames and $y_{\text{GT}} = (y_{\text{GT}}^2, \ldots, y_{\text{GT}}^T)$ denotes a sequence of elements $y_{\text{GT}}^t \in \mathcal{Y}^t$ that refer to the ground truth assignment $a_{y_{\text{GT}}^t}^t$ of objects $\mathcal{D}^{t-1}$ and $\mathcal{D}^t$ between frames $t-1$ and t.

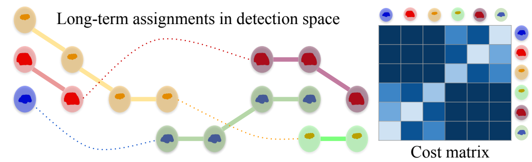

Figure 3. Post-processing for long range assignments (Sec. 3.4). Linking of tracklets is shown on the left (with dotted lines). Linking is posed as an assignment problem with a cost matrix (right). Dark blue cost matrix entries represent time-inconsistent impossible assignments. Lighter color denotes lower cost.

3.4. Long-Range Assignments

Using the approach discussed in Sec. 3.2, we obtain a path in the assignment space connecting every frame-pair. This path is globally optimal for the formulated cost and results in multiple tracklets being formed in the detection space, as shown in Fig. 3 (left). However, the formulation in Sec. 3.2 does not recover links between detections missing for more than one frame. In the case of occlusions and faulty detections, objects may reappear in the video after multiple frames. To account for this situation, we devise a global assignment based approach to connect the obtained tracklets. This is illustrated in Fig. 3 and described next.

Consider r tracklets obtained by optimizing the program given in Eq. (5). Note that the r tracklets also contain detections that haven't yet been assigned. We construct an $r \times r$ cost matrix $c_{\text{lr}} \in \mathbb{R}^{r\times r}$ where the $(i,j)^{\text{th}}$ element denotes the cost of joining the i^{th} tracklet to the j^{th} tracklet. The cost is based on detections in tracklet i and detections in tracklet j. Specifically, our cost function is

$$c_{\text{lr}}(i,j) = \lambda_{\text{app,lr}} f_{\text{app,lr}}(i,j) + \lambda_{\text{dist,lr}} f_{\text{dist,lr}}(i,j).$$

Here, $f_{\text{app,lr}}(i,j)$ denotes the Euclidean distance between the average of the appearance feature vectors for detections in tracklets i and j. $f_{\text{dist,lr}}(i,j)$ is the Euclidean distance between bounding box centers of the last occurring detection in tracklet i and the first occurring detection in tracklet j. We obtain the appearance features from PointTrack [58]. The parameters $\lambda_{\text{app,lr}}$ and $\lambda_{\text{dist,lr}}$ are trainable. Specifically, the learning objective follows Eq. (6) with the notable difference that we are not considering a temporal sequence here, *i.e.*, dynamic programming isn't needed for long-range assignments. The task loss Δ is either 0 or 1 depending on whether the particular long-range assignment is a valid assignment or not a valid assignment.

Note, linking of tracklets should be time-consistent, *i.e.*, tracklets that end at or before frame t cannot be merged with tracklets that begin at or before frame $t+1$. We also assume that objects that don't appear in the scene for more than 40 consecutive frames have left the scene, and won't appear again. In other words, tracklets that end at or before frame t cannot be merged with tracklets that begin after frame $t+40$. These constraints are imposed via a very high cost ($=10^5$) in the corresponding positions of the cost matrix. In Fig. 3, the dark blue positions in the cost matrix represent the time-inconsistent assignments.

After constructing the cost matrix, we use the Hungarian algorithm [33] to solve this assignment problem. Tracks are linked according to the assignment solution, if the corresponding cost is less than an empirically determined threshold. Otherwise, a new track ID is assigned to the tracklet.

4. Experiments

We study our proposed method on the MOTS task. The datasets and evaluation metrics for these tasks are discussed in Sec. 4.1. We provide quantitative analysis and comparisons to other MOTS methods in Sec. 4.2. To demonstrate generality, we also study the approach on the MOT task and compare to recent batch-methods in Sec. 4.2. In Sec. 4.3 we present a qualitative comparison of our approach to PointTrack [58], the next best among the MOTS methods which do not require additional depth information. Next, we also discuss some failure cases of our approach.

4.1. Datasets and Evaluation Metrics

We use the KITTI-MOTS and MOTSChallenge datasets to assess results on the MOTS task. We use the MOT17 dataset to study our approach on the MOT task. The datasets are described in the supplementary.

The main evaluation metric on the official KITTI-MOTS benchmark is the recently introduced higher order tracking accuracy (HOTA) [28]. To compare trackers, HOTA explicitly balances detection accuracy (DetA), association accuracy (AssA) and localization accuracy (LocA) [28] into a single unified metric. For completeness we also report performance on the previously common evaluation metrics for MOTS: MOTSA (MOTS Accuracy) and sMOTSA (soft MOTS Accuracy) [54]. MOTSA assesses the true positive detections (TP) and penalizes false positives (FP), false negatives (FN) and ID switches (IDS) between objects. sMOTSA is a variant of MOTSA that accounts for the segmentation accuracy by incorporating the intersection over union of the predicted segmentations and ground truth segmentations. sMOTSA and MOTSA overemphasize the importance of detections in the task of MOTS. For this reason, we also report the ID F1 scores (IDF1) [44] whenever available, which quantifies identity preservation, an important aspect of tracking. For the MOT task, Multiple Object Tracking Accuracy (MOTA) [6], IDF1 [44] and ID switches are reported, which are official metrics for MOTChallenge.

4.2. Quantitative Results

We quantitatively analyze the approach on the KITTI-MOTS (Tab. 1, Tab. 2) and MOTSChallenge (Tab. 6) datasets for the MOTS task. For the comparison we use the evaluation metrics discussed in Sec. 4.1. Note, in all experiments "Ours (JT)" represents jointly training (λ, θ) as discussed in Sec. 3.3. "Ours" refers to training of tracking parameters λ, while fixing pre-trained deep-net parameters θ. Specifics of the deep-net are provided in the appendix.

We also study the effects of each component and parameter in our method carefully through an ablation study on the KITTI-MOTS car validation set (Tab. 3, Tab. 4 and Tab. 5).

Results on KITTI-MOTS. In Tab. 1, we compare the approach to all published methods on the KITTI-MOTS test data available on the leaderboard. ViP-DeepLab [39] and MOTSFusion [27] use additional depth information and are hence reported separately. Our method outperforms other available 2D methods on the overall HOTA metric. The main improvement is due to associations. We improve AssA by 14% (car) and 9% (pedestrian) compared to PointTrack [58]. On pedestrian tracking we also improve upon MOTSFusion [27], which uses 3D information. The improvement is again due to associations (8% improvement in AssA) which translates to 4.8% improvements for HOTA.

Tab. 2 shows the results for cars and pedestrians respectively on the KITTI-MOTS validation data, when the detections and segmentations are obtained from the TrackR-CNN [54] deep-net. For a fair comparison, we fix TrackRCNN [54] to be the deep-net for all the tracking methods shown in Tab. 2. Note that HOTA, DetA, AssA and LocA metrics are not available for these trackers on the validation data. We hence use the old metrics sMOTSA, MOTSA and IDS. We observe that our method outperforms PointTrack [58] by 10% on cars, and by 7% on pedestrians in terms of IDS. sMOTSA and MOTSA are not available for PointTrack on TrackRCNN detections. Tab. 3 shows additional analysis of tracking on PointTrack detections, detailed later in the ablation study.

Ablation Study on KITTI-MOTS. In Tab. 3, we compare configurations of PointTrack [58] and our approach. In PointTrack(liv:1) only 2 frames are considered at a time. 'Ours (WL)' (without long-range associations) refers to the

KITTI-MOTS Test: Car												
Method	**3D**	**Off.**	**HOTA↑**	**DetA↑**	**AssA↑**	**DetRe↑**	**DetPr↑**	**AssRe↑**	**AssPr↑**	**LocA↑**	**sMOTSA↑**	**MOTSA↑**	**IDS↓**
ViP-DeepLab [39]	✓		76.38	82.70	70.93	88.70	88.77	75.86	86.00	90.75	81.03	90.74	392
MOTSFusion [27]	✓	✓	73.63	75.44	72.39	78.32	90.78	75.53	89.97	90.29	74.98	84.12	201
TrackR-CNN [54]			56.63	69.90	46.53	74.63	84.18	63.13	62.33	86.60	66.97	79.67	692
PointTrack [58]			61.95	**79.38**	48.83	**85.77**	85.66	**79.07**	56.35	88.52	**78.50**	**90.88**	346
Ours (JT)		✓	**68.73**	76.43	**62.24**	79.73	**89.62**	71.01	**77.77**	**89.65**	75.51	85.46	**300**
KITTI-MOTS Test: Pedestrian													
Method	**3D**	**Off.**	**HOTA↑**	**DetA↑**	**AssA↑**	**DetRe↑**	**DetPr↑**	**AssRe↑**	**AssPr↑**	**LocA↑**	**sMOTSA↑**	**MOTSA↑**	**IDS↓**
ViP-DeepLab [39]	✓		64.31	70.69	59.48	75.71	81.77	67.52	74.92	84.40	68.76	84.52	209
MOTSFusion [27]	✓	✓	54.04	60.83	49.45	64.13	81.47	56.68	70.44	83.71	58.75	72.89	279
TrackR-CNN [54]			41.93	53.75	33.84	57.85	72.51	45.30	50.74	78.03	47.31	66.14	482
PointTrack [58]			54.44	**62.29**	48.08	**65.49**	81.17	**64.97**	58.66	83.28	**61.47**	**76.51**	176
Ours (JT)		✓	**58.81**	60.72	**57.67**	63.10	**84.41**	64.47	**79.96**	**85.20**	60.27	72.65	**143**

Table 1. Comparison on the KITTI-MOTS Test data. All published methods on the KITTI-MOTS leaderboard are displayed. '3D' represents additional depth information. 'Off.' represents offline methods. 'Ours (JT)' performs the best in terms of association (AssA) and overall (HOTA) when 2D methods are compared.

KITTI-MOTS Validation: Car						
Tracking	**Detections**	**3D**	**Off.**	**sMOTSA↑**	**MOTSA↑**	**IDS ↓**
CIWT [36]	TRCNN [54]	✓		68.1	79.4	106
BePix [48]	TRCNN [54]	✓		76.9	89.7	88
MOTSFusion [27]	TRCNN [54]	✓	✓	78.2	90	36
CAMOT [37]	TRCNN [54]			67.4	78.6	220
TRCNN [54]	TRCNN [54]			76.2	87.8	93
PointTrack [58]	TRCNN [54]			-	-	46
Ours	TRCNN [54]		✓	**77.4**	**89.6**	**41**
KITTI-MOTS Validation: Pedestrian						
Tracking	**Detections**	**3D**	**Off.**	**sMOTSA↑**	**MOTSA↑**	**IDS ↓**
CIWT [36]	TRCNN [54]	✓		42.9	61	42
MOTSFusion [27]	TRCNN [54]	✓	✓	50.1	68	34
CAMOT [37]	TRCNN [54]			39.5	57.6	131
TRCNN [54]	TRCNN [54]			46.8	65.1	78
PointTrack [58]	TRCNN [54]			-	-	30
Ours	TRCNN [54]		✓	**48.9**	**66.7**	**28**

Table 2. Results on the KITTI-MOTS Validation using old metrics (HOTA is not available for most methods). '3D' represents additional depth information. 'Off.' represents offline method. We use '-' for numbers that weren't reported. Note: sMOTSA and MOTSA are detection heavy metrics, and hence IDS is a more accurate indicator of tracking performance in this case.

Tracking	**Dets.**	**HOTA**	**DetA**	**AssA**	**LocA**	**IDF1**	**sMOTSA**	**IDS ↓**
Pt.(liv:1)	Pt. [58]	70.1	**85.6**	59.3	**91.1**	70.1	84.7	84
Pt. [58]	Pt. [58]	73.8	**85.6**	64.0	**91.1**	74.8	**85.5**	22
Ours (WL)	Pt. [58]	81.3	**85.6**	77.8	**91.1**	85.3	84.9	77
Ours	Pt. [58]	83.4	**85.6**	81.8	**91.1**	89.4	85.4	22
Ours (JT)	Pt. [58]	**85.0**	85.3	**86.1**	90.9	**92.2**	85.3	**20**

Table 3. Different configurations of our approach and PointTrack (Pt.) [58] on PointTrack detections. 'Pt.(liv:1)' and 'Ours (WL)' only consider consecutive frames.

same setting where consecutive frames are considered via dynamic programming, but long-term associations are ignored. 'Ours (WL)' outperforms PointTrack (liv:1) establishing the effectiveness of the dynamic programming based tracking. In the last 2 rows, we show how joint-training ('Ours (JT)'), as discussed in Sec. 3.3, improves the results over 'Ours.' Note that the detection-segmentation network is fixed to be PointTrack for Tab. 3. There is a 12% improvement in HOTA from PointTrack to 'Ours (JT).' 'Ours (JT)' outperforms PointTrack by 22% on the association accuracy (AssA). This establishes the efficacy of the structured learning approach discussed in Sec. 3.

	K	**HOTA**	**DetA**	**AssA**	**LocA**	**IDF1**	**sMOTSA**	**IDS↓**
Ours	1	80.6	**85.6**	76.4	**91.1**	85.8	**85.5**	25
Ours	2	82.5	**85.6**	79.9	**91.1**	88.8	**85.5**	24
Ours	5	82.5	**85.6**	80.0	**91.1**	88.8	**85.5**	**22**
Ours	15	**83.4**	**85.6**	**81.8**	**91.1**	**89.4**	85.4	**22**
Ours	20	**83.4**	**85.6**	**81.8**	**91.1**	**89.4**	85.4	**22**

Table 4. Study on how k affects tracking on the KITTI-MOTS validation set (cars). Detections are fixed (from PointTrack [58]).

	Cost			**HOTA**	**DetA**	**AssA**	**LocA**	**IDF1**	**sMOTSA**	**IDS↓**
	iou	app	dist							
Ours (WL)		✓	✓	80.5	**85.6**	76.4	**91.1**	84.1	84.4	108
Ours (WL)	✓		✓	80.6	**85.6**	76.8	**91.1**	84.5	84.6	96
Ours (WL)	✓	✓		81.2	**85.6**	77.4	**91.1**	85.0	84.5	100
Ours (WL)	✓	✓	✓	**81.3**	**85.6**	**77.8**	**91.1**	**85.3**	**84.9**	**77**

Table 5. Study on how the cost modalities affect tracking on the KITTI-MOTS validation set (cars). Detections are fixed (from PointTrack [58]). 'Ours (WL)' represents our method without long-range connections (discussed in Sec. 3.4).

Tab. 4 shows the effect of choosing k (Sec. 3.2). Here $k = 1$ defaults to local frame-by-frame assignments obtained from the Hungarian algorithm [33]. No pairwise cost (Eq. (3)) is involved when $k = 1$. The performance improves by increasing k, but beyond $k = 15$, we observe no improvement on the KITTI-MOTS car validation set. We fix $k = 20$ in all experiments. A high value of k doesn't incur any noticeable additional computational cost. Fig. 4 further highlights this. It shows the best paths $\{y^{t,*}\}$, obtained from Sec. 3.2, for each video of the KITTI-MOTS car validation set. Notice that $y^{t,*}$ is often higher than 1 but never higher than 7, indicating that $k = 20$ is suitable.

Tab. 5 shows the effect of different modalities: (a) intersection over union (iou), (b) the appearance features (app), and (c) distance measures (dist), as discussed in Sec. 3. In the first, second and third rows, we remove the parameters corresponding to "iou", "app" and "dist" respectively and

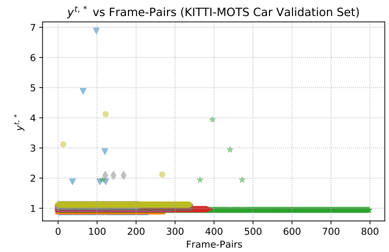

Figure 4. We show the optimal path $\{y^{t,*}\}$ obtained from Eq. (5) for the KITTI-MOTS validation set on cars. Different markers represent different videos.

MOTSChallenge Val				
Method	**sMOTSA↑**	**IDF1 ↑**	**MOTSA↑**	**IDS↓**
TRCNN [54]	52.7	81.7	66.9	315
PointTrack [58]	58.1	-	70.6	-
Ours	**63.2**	**85.2**	**72.9**	**289**

MOTSChallenge Test				
Method	**sMOTSA↑**	**IDF1 ↑**	**MOTSA↑**	**IDS↓**
TRCNN [54]	40.6	42.4	55.2	567
Ours	**55.0**	**64.0**	**64.2**	**330**

Table 6. Results on the MOTSChallenge dataset ('-' indicates numbers that weren't reported).

analyze the effects of each missing term. Note that WL refers to the "without long-term" configuration, *i.e.*, we do not use long range assignments described in Sec. 3.4.

Results on MOTSChallenge. Tab. 6 compares tracking methods on the MOTSChallenge validation and test dataset. To evaluate generality we use the same tracking parameters (λ) that were used for evaluating on the KITTI-MOTS dataset. Our approach improves TrackRCNN [54] and PointTrack [58] on all the official metrics (sMOTSA, MOTSA, IDF1 and IDS). Note that IDF1 and IDS scores of PointTrack on the MOTSChallenge validation set have not been reported by Xu *et al*. [58]. PointTrack results on the test set are not available on the leaderboard. 3D methods [27, 39] could not be evaluated on the MOTSChallenge data because this dataset lacks depth information.

Batch Methods on MOT17. Tab. 7 compares different batch methods on the MOT17 train dataset. Following [15], our method is trained in a leave-one-out fashion, so that results can be compared meaningfully. We notice that our method consistently outperforms other methods in terms of IDF1, which is a good indicator of identity preservation. Note that Lif-T [15], MPNTrack [8] and 'Ours' use additional detections from [5] following [15]. Our MOTA score is slightly lower than [15], which is due to higher false positive (FP) and false negetive (FN) detections. We also compare a variant, where we use only linear interpolation as a post processing step, rather than obtaining predictions from [5] (indicated by '(lin)' in Tab. 7). We outperform Lif-T [15] in both MOTA and IDF1 in this setting.

4.3. Qualitative Results

Success cases. Fig. 1 shows some qualitative examples where our method preserves the ID of objects, misidentified by PointTrack [58]. The color of a mask represents the object's ID. In the first example (left), PointTrack misidentifies the purple car as green when it reappears. In the second example (middle), PointTrack misidentifies the blue car as red, even though the car is always visible and detected. In the third example (right), a car rightly classified as blue in the 3rd frame, is misidentified as green in the 1st frame.

MOT17							
Method	**MOTA ↑**	**IDF1↑**	**MT↑**	**ML↓**	**FP↓**	**FN↓**	**IDS.↓**
Tracktor [5] (o)	55.7	65.2	550	374	1732	124291	903
MHT-bLSTM [22]	51.5	59.9	410	590	10583	151561	1233
MHT-DAM [21]	53.0	53.2	441	555	8798	148108	1237
TT17 [60]	56.5	67.1	465	571	8097	137789	600
MPNTrack [8]	64.4	71.2	649	360	5715	113738	**482**
Lif-T [15]	**66.9**	72.3	**679**	364	**2655**	**107803**	791
Ours	66.7	**72.6**	650	**360**	2791	108738	771
Lif-T [15] (lin)	59.5	68.8	556	412	7518	**128153**	774
Ours (lin)	**59.6**	**69.4**	**562**	**410**	**7381**	128695	**768**

Table 7. Results on the MOT17 train data. To ensure a fair comparison, our method is trained in a leave-one-out fashion, following Lif-T [15]. '(o)' represents online method. All the other methods are batch methods. '(lin)' indicates linear interpolation.

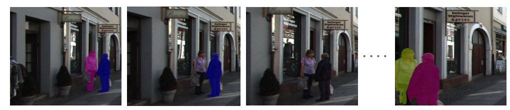

Figure 5. Failure case: The person previously assigned pink is assigned yellow when it reappears. An ID switch also happens for the other person (blue switches to pink).

Failure cases. Our method fails when the appearance of reappearing objects changes significantly. Fig. 5 shows one such case. The person previously assigned pink gets assigned yellow after being occluded for multiple frames. The person initially assigned blue is assigned pink later. This could be addressed by using more robust appearance features. We defer this to future work. Several other reasons for failure modes are: (1) The threshold for long-range assignments (Sec. 3.4) is determined heuristically. (2) The cost function for parameter training (Sec. 3.3) is a linear function of its constituent terms. Exploring better functions is part of our future work.

5. Conclusion

We designed an assignment-space-based formulation for MOTS. It differs from prior work which operates on object detections. We use dynamic programming to find the global minimum cost path which associates detections over consecutive frames. Parameters are learned end-to-end. Long-range associations are formed in a 2nd assignment task.

Acknowledgements: This work is supported in party by Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/project accession no.1024178 from the USDA National Institute of Food and Agriculture: NSF/USDA National AI Institute: AIFARMS. We also thank the Illinois Center for Digital Agriculture for seed funding for this project.

References

[1] Karteek Alahari, Pushmeet Kohli, and Philip HS Torr. Reduce, reuse & recycle: Efficiently solving multi-label mrfs. In *CVPR*, 2008. 3

[2] Anton Andriyenko, Konrad Schindler, and Stefan Roth. Discrete-continuous optimization for multi-target tracking. In *CVPR*, 2012. 1, 2

[3] Jerome Berclaz, Francois Fleuret, and Pascal Fua. Robust people tracking with global trajectory optimization. In *CVPR*, 2006. 1, 2

[4] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 2011. 1, 2

[5] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 8

[6] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP JIVP*, 2008. 6

[7] Matthew B Blaschko and Christoph H Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008. 3

[8] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *CVPR*, 2020. 8

[9] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *CVPR*, 2019. 1, 2, 3

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017. 2

[11] Thomas Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE-JOE*, 1983. 2

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 12

[13] Ross Girshick. Fast r-cnn. In *CVPR*, 2015. 1, 2

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 2, 3, 11

[15] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. In *ICML*, 2020. 8

[16] Yuan-Ting Hu, Hong-Shuo Chen, Kexin Hui, Jia-Bin Huang, and Alexander Schwing. SAIL-VOS: Semantic Amodal Instance Level Video Object Segmentation - A Synthetic Dataset and Baselines. In *CVPR*, 2019. 1

[17] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. MaskRNN: Instance Level Video Object Segmentation. In *NIPS*, 2017. 1

[18] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. Unsupervised Video Object Segmentation using Motion Saliency-Guided Spatio-Temporal Propagation. In *ECCV*, 2018. 1

[19] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. VideoMatch: Matching based Video Object Segmentation. In *ECCV*, 2018. 1

[20] Chang Huang, Bo Wu, and Ramakant Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008. 1, 2

[21] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *ICCV*, 2015. 2, 8

[22] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *ECCV*, 2018. 8

[23] Suna Kim, Suha Kwak, Jan Feyereisl, and Bohyung Han. Online multi-target tracking by large margin structured learning. In *ACCV*, 2012. 3

[24] Christoph H Lampert, Matthew B Blaschko, and Thomas Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *TPAMI*, 2009. 3

[25] Byungjae Lee, Enkhbayar Erdenee, Songguo Jin, Mi Young Nam, Young Giu Jung, and Phill Kyu Rhee. Multi-class multi-object tracking using changing point detection. In *ECCV*, 2016. 2

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2

[27] Jonathon Luiten, Tobias Fischer, and Bastian Leibe. Track to reconstruct and reconstruct to track. *RAL*, 2020. 1, 2, 6, 7, 8

[28] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixe, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *IJCV*, 2020. 6

[29] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *ACCV*, 2018. 11

[30] 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 *CVPR*, 2018. 2

[31] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv:1603.00831*, 2016. 12

[32] Anton Milan, Konrad Schindler, and Stefan Roth. Detection- and trajectory-level exclusion in multiple object tracking. In *CVPR*, 2013. 2

[33] James Munkres. Algorithms for the assignment and transportation problems. *J-SIAM*, 1957. 2, 4, 6, 7

[34] Katta Murty. An algorithm for ranking all the assignments in order of increasing cost. *Operations Research*, 1968. 1, 3, 4, 11

[35] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *CVPR*, 2019. 1, 2, 3, 11

[36] Aljoša Osep, Wolfgang Mehner, Markus Mathias, and Bastian Leibe. Combined image-and world-space tracking in traffic scenes. In *ICRA*, 2017. 7

[37] Aljoša Ošep, Wolfgang Mehner, Paul Voigtlaender, and Bastian Leibe. Track, then decide: Category-agnostic vision-based multi-object tracking. *ICRA*, 2018. 2, 7

[38] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 2

[39] Siyuan Qiao, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. In *CVPR*, 2021. 6, 7, 8

[40] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1, 2

[41] Donald Reid. An algorithm for tracking multiple targets. *TAC*, 1979. 2

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2

[43] Seyed Hamid Rezatofighi, Anton Milan, Zhen Zhang, Qinfeng Shi, Anthony Dick, and Ian Reid. Joint probabilistic data association revisited. In *ICCV*, 2015. 2

[44] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV*, 2016. 6

[45] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2

[46] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *CVPR*, 2017. 2

[47] Aleksandr V Segal and Ian Reid. Latent data association: Bayesian model selection for multi-target tracking. In *ICCV*, 2013. 2

[48] Sarthak Sharma, Junaid Ahmed Ansari, J Krishna Murthy, and K Madhava Krishna. Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking. In *ICRA*, 2018. 2, 7

[49] Horesh Ben Shitrit, Jérôme Berclaz, François Fleuret, and Pascal Fua. Multi-commodity network flow for tracking multiple people. *PAMI*, 2013. 1, 2

[50] Francesco Solera, Simone Calderara, and Rita Cucchiara. Learning to divide and conquer for online multi-target tracking. In *ICCV*, 2015. 3

[51] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning crfs using graph cuts. In *ECCV*, 2008. 3

[52] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 4

[53] Wei Tian, Martin Lauer, and Long Chen. Online multi-object tracking using joint domain information in traffic scenarios. *T-ITS*, 2019. 2

[54] Paul Voigtlaender, Michael Krause, Aljoša Ošep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. MOTS: Multi-object tracking and segmentation. In *CVPR*, 2019. 1, 2, 6, 7, 8, 12

[55] Shaofei Wang and Charless C Fowlkes. Learning optimal parameters for multi-target tracking. In *BMVC*, 2015. 1, 2, 3

[56] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017. 2

[57] Zheng Wu, Thomas H Kunz, and Margrit Betke. Efficient track linking methods for track graphs using network-flow and set-cover techniques. In *CVPR*, 2011. 2

[58] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Liusheng Huang. Segment as points for efficient online multi-object tracking and segmentation. In *ECCV*, 2020. 1, 2, 4, 6, 7, 8, 11, 14

[59] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008. 1, 2, 11

[60] Yang Zhang, Hao Sheng, Yubin Wu, Shuai Wang, Weifeng Lyu, Wei Ke, and Zhang Xiong. Long-term tracking with deep tracklet association. *TIP*, 2020. 8