Learning to Track with Object Permanence

Pavel Tokmakov   Jie Li   Wolfram Burgard   Adrien Gaidon

Toyota Research Institute

`first.last@tri.global`

Abstract

Tracking by detection, the dominant approach for online multi-object tracking, alternates between localization and association steps. As a result, it strongly depends on the quality of instantaneous observations, often failing when objects are not fully visible. In contrast, tracking in humans is underlined by the notion of object permanence: once an object is recognized, we are aware of its physical existence and can approximately localize it even under full occlusions. In this work, we introduce an end-to-end trainable approach for joint object detection and tracking that is capable of such reasoning. We build on top of the recent CenterTrack architecture, which takes pairs of frames as input, and extend it to videos of arbitrary length. To this end, we augment the model with a spatio-temporal, recurrent memory module, allowing it to reason about object locations and identities in the current frame using all the previous history. It is, however, not obvious how to train such an approach. We study this question on a new, large-scale, synthetic dataset for multi-object tracking, which provides ground truth annotations for invisible objects, and propose several approaches for supervising tracking behind occlusions. Our model, trained jointly on synthetic and real data, outperforms the state of the art on KITTI and MOT17 datasets thanks to its robustness to occlusions.

1. Introduction

Consider the video sequence from the KITTI dataset [24] shown in Figure 1. A man on the left walks behind the moving car and is not visible anymore. Yet, there is no question that he is still there, and did not simply vanish. Moreover, we can approximately infer his location at that moment. This capability is known to cognitive scientists as *object permanence*, and is observed in infants at a very early age [3, 53]. In adults, understanding that occluded objects do not disappear is important for tasks like driving. In this work, we propose a deep learning-based method for multi-object tracking that is capable of such reasoning.
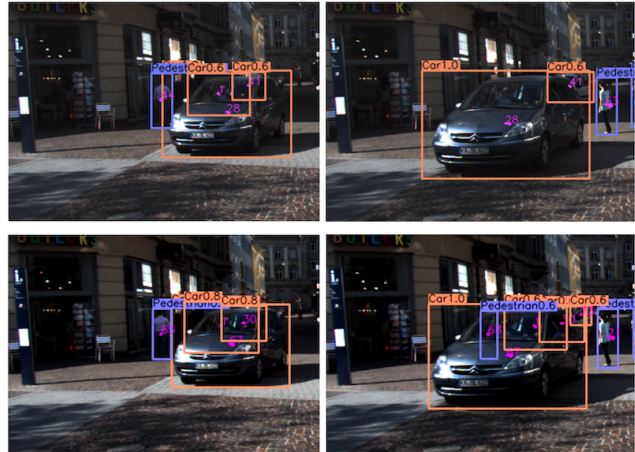
Virtually all modern multi-object tracking algorithms

Figure 1. Video frames from the KITTI dataset with outputs of CenterTrack [66] (above), and our method (below). By modeling object permanence, our approach is able to hallucinate trajectories of fully occluded instances, such as the person behind the car.

operate in the tracking-by-detection paradigm. That is, they use an existing object detector to localize objects of interest in every frame of a video, and then link them into tracks, either in an online [9, 11], or in an offline manner [5, 6, 10, 34, 42]. In this work we focus on the online setting, where a method needs to associate current detections with previously established trajectories [9, 54, 59, 62]. A major limitation of these methods is that the localization step is completely independent from the previous history, thus, once an object becomes partially or fully occluded, the detector fails and the trajectory gets broken (see Figure 1, top). Recently, several approaches combine detection and tracking in a single model [7, 66]. They take pairs of frames as input and output detections together with pairwise associations. While these methods improve tracking robustness, they can only handle single-frame occlusions.

In this work, we propose an end-to-end trainable, online approach for multi-object tracking that leverages object permanence as an inductive prior. To this end, we first extend the recent CenterTrack architecture [66] from pairs of frames as input to *arbitrary video sequences*. The frames are processed by a convolutional gated recurrent unit (Con-

vGRU) [4] that encodes the spatio-temporal evolution of objects in the input video, taking the entire history into account. As a result, it can can reason about locations of partially and fully occluded instances using the object permanence assumption (see Figure 1, bottom).

Supervising this behavior is a major challenge. No tracking dataset currently provides consistent annotations for occluded objects at a large scale (see Figure 5) because of the associated uncertainty. In this paper, we instead propose to use *synthetic data*. Using the Parallel Domain (PD) [1] simulation platform, we generate a dataset of synthetic videos that automatically provides accurate labels for all objects, irrespective of their visibility (see Figure 2). We then use this dataset to analyze various approaches for supervising tracking behind occlusions with both ground-truth and pseudo-ground-truth labels for occluded instances.

Despite the progress in simulation, the domain gap between synthetic and real videos may limit performance. As we show in Section 4.3, a model directly trained on synthetic videos indeed underperforms when applied to real-world multi-object tracking benchmarks such as KITTI [24]. However, we find that this domain gap can be overcome by just training our model jointly on synthetic and real data, supervising invisible objects only on synthetic sequences. This allows to learn complex behaviors across domains, including trajectory hallucination for fully occluded instances although this was never labeled in the real world.

Our contributions are three-fold. (1) We propose an end-to-end trainable architecture for joint object detection and tracking that operates on videos of arbitrary length in Sec. 3.2. (2) We demonstrate how this architecture can be trained to hallucinate trajectories of fully invisible objects in Sec. 3.3. (3) We show how to supervise our method with a mix of synthetic and real data in Sec. 3.4, and validate it on the KITTI [24] and MOT17 [38] real-world benchmarks in Sec. 4, outperforming the state of the art. Source code, models, and data are publicly available at https://github.com/TRI-ML/permatrack.

2. Related Work

Our approach addresses the problem of *multi-object tracking* by designing a joint model for *object detection and tracking in videos* and training it on *synthetic data*. Below, we review the most relevant works in each of these fields.

Multi-object tracking is the problem of localizing objects from a predefined list of categories in a video and associating them over time based on identity. Most existing approaches treat these two tasks separately, in a paradigm known as tracking-by-detection. The main difference between the methods in this category is whether the association step is performed online or offline.

State-of-the-art online object trackers [7, 9, 59, 62, 66]

keep a set of active trajectories as they progress through a video. In every frame, a new list of object detections is processed by either associating them with an existing trajectory, or starting a new one. Early approaches, such as SORT [9], used Kalman filter to associate detections based on bounding box overlap, or appearance features from a deep network [59]. Recent methods proposed to utilize more complex features for association, such as human pose [54], or trajectory representations learned with spatio-temporal graph convolutions [62]. Although some of the methods in this category use a linear motion model [11, 40, 64] to propagate a trajectory hypothesis behind occlusions, this heuristic ignores changes in the scene context. In contrast, our approach learns to hallucinate trajectories of occluded objects in an end-to-end manner, outperforming the aforementioned heuristic.

On the other hand, offline approaches [5, 6, 10, 34] first build a spatio-temporal graph spanning the whole video, with object detections as nodes [5]. Edge costs are defined based on overlap between detections [30, 42, 65], their appearance similarity [10, 39, 47, 62], or motion-based models [2, 13, 16, 34, 44]. The association can then be formulated as maximum flow [6] or, minimum cost problem [30, 34]. While these methods can handle complex scenarios, they are not practical due to their non-casual nature and computational complexity. In contrast, our approach does not require future frames and runs in real time.

A few methods have recently attempted to combine object detection and tracking in a single end-to-end learnable framework. Bergman *et al.* [7] utilize the box regression branch in the Faster RCNN detector [45] to propagate objects from frame $t - 1$ to t. Zhou *et al.* [66] take a pair of frames as input, and directly output the detections and association vectors, resulting in a simpler architecture. However, both these methods only capture short-term object correspondences. Our approach builds on top of [66], but modifies it to model spatio-temporal evolution of objects in video sequences of arbitrary length, and uses synthetic data to learn to detect and track even under full occlusions.

Several classical methods [26, 28, 41] have attempted to capture the notion of object permanence through heuristics, such as correlation between the motion of visible and invisible instances in [26]. However, these rule-based methods lack in flexibility. Recently, Shamsian *et al.* [51] proposed a fully learning-based approach with promising results on toy, synthetic examples. In contrast, our method is capable of handling full occlusions in the wild.

Video object detection is primarily concerned with improving the robustness of detectors in videos. Early approaches processed frames individually, but used a Siamese network to establish association between detections and smooth their scores [21]. Later, Kang *et al.* [31] proposed to pass a stack of several frames to a network and directly
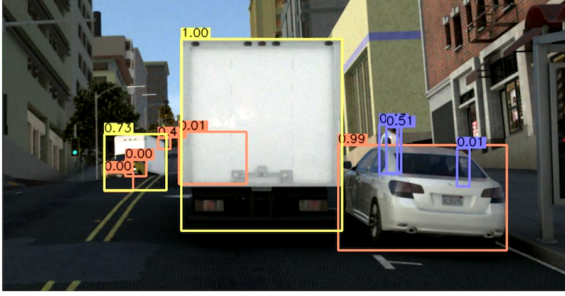
Figure 2. A sample from our synthetic dataset (Section 4.1) with ground truth visibility labels. Unlike real datasets, virtual ones provide accurate annotations for all the objects, irrespective of their visibility.

output short object tubelets. Finally, Xiao *et al.* [61] augmented an object detector with a spatio-temporal memory module, allowing it to process videos of arbitrary length. Notice that none of these methods tackled the problem of multi-object tracking. Instead, they used short-term associations to improve detection robustness. Similarly to [61], our architecture also combines an object detector with a spatio-temporal memory module, however, we use a more recent CenterPoint [67] detector framework and train the model to hallucinate trajectories of fully occluded objects using synthetic data.

Synthetic data has been used in the past to circumvent the need for manually labeling images [18, 46, 48] or videos [19, 22, 55]. Most approaches have focused on the setting in which no labels are available in the real world, leveraging unsupervised domain adaptation techniques such as adversarial training [23, 27, 35, 56] and self-training [33, 50, 68, 69]. Although significant progress has been achieved, the performance of these models remains significantly below their counterparts trained on real data in a fully supervised way. In videos, the most successful approaches combined large amounts of synthetic data with small sets of labeled real videos [29, 55]. In this work, we follow a similar route and utilize synthetic data to obtain the expensive labels for occlusion scenarios, while relying on visible object annotations in multi-object tracking datasets to minimize the domain gap. Synthetic datasets with object track labels have been proposed in the past [20, 22]. We choose to capitalize on the recent progress in simulation tools and collect a new, more realistic one.

3. Methodology

3.1. Background

We build our method on top of the recent Center-Track [66] architecture. Their approach addresses tracking from a local perspective. In particular, CenterTrack takes a pair of frames $\{I^{t-1}, I^t\}$ as input together with H^{t-1}

- an encoding of locations of previously detected objects in frame $t - 1$. Objects are represented with their center points $\mathbf{p} \in \mathbb{R}^2$, thus H^{t-1} is compactly encoded with a heatmap. The three input tensors are concatenated and passed through a backbone network f, producing a feature map $F^t = f(H^{t-1}, I^{t-1}, I^t)$, which is used to both localize the object centers in the current frame $\{\hat{\mathbf{p}}_0^t, \hat{\mathbf{p}}_1^t, ...\}$, regress their bounding box sizes $\{\hat{s}_0^t, \hat{s}_1^t, ...\}$, and predict their displacement vectors with respect to the previous frame $\{\hat{\mathbf{d}}_0^t, \hat{\mathbf{d}}_1^t, ...\}$. At test time, displacement vectors are used to project each center to the previous frame via $\hat{\mathbf{p}}_i^t - \hat{\mathbf{d}}_i^t$, and then greedily match it to the closest available center $\hat{\mathbf{p}}_*^{t-1}$, thus recovering the tracks (see [66] for more details).

The local nature of CenterTrack is both its strength and its weakness. While only considering a pair of consecutive frames simplifies the architecture of the model, it limits its representational power. In particular, it cannot capture the notion of object permanence in videos, and learn to localize and associate objects under full occlusions. To address this limitation, we first extend [66] to a video-level model in Section 3.2. We then describe how to train such a model to track invisible objects using synthetic data in Section 3.3, and detail our domain adaptation approach is Section 3.4.

3.2. A video-level model for tracking

Our model, shown in Figure 3, takes a sequence of frames $\{I^1, I^2, ..., I^n\}$ as input. Each frame is passed through the backbone f individually to obtain feature maps $\{F^1, F^2, ..., F^n\}$, which, per CenterTrack formalism, encode the locations of *visible* objects in that frame - an instantaneous representation. To learn a permanent representations of objects in a video, we augment our network with a convolutional gated recurrent unit (ConvGRU) [4]. It is an extension of the classical GRU [14], which replaces a 1D state vector with a 2D feature map M, and fully connected layers, used to compute state updates, with convolutions.

At each time step t, the corresponding feature map F^t is passed to the ConvGRU, together with the previous state M^{t-1} to compute the updated state $M^t = GRU(M^{t-1}, F^t)$. Intuitively, the state matrix M represents the entire history of the previously seen objects $\{o_1, o_2, ...\}$ in frames $\{1, ..., t - 1\}$ and is updated with the encoding of the visible objects in frame t via a series of learnable, multiplicative transformations (see [4] for further details). It can thus model the spatio-temporal evolution of objects in the input video sequence by guiding their localization and association in frame t using previous history. Moreover, it can predict locations of the objects that were seen in the past, but are currently occluded. Notice that with this architecture there is no need to pass the explicit encoding of the previous frame centers H^{t-1}, since they are already captured in the ConvGRU state M^{t-1}.

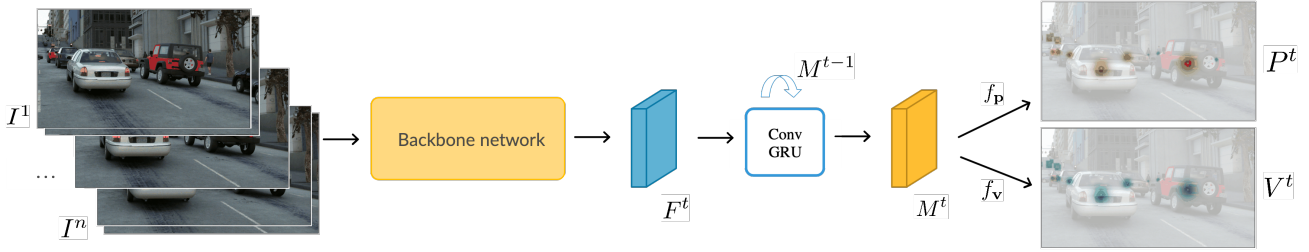In practice, to generate the tracks on-line, M^t

Figure 3. Our method takes a sequence of frames as input and processes them individually with a backbone network (shown in yellow). The resulting feature maps (shown in blue) are passed to the ConvGRU module which aggregates a representation of the scene, encoding all the previously seen objects, even if they are fully occluded in the current frame. The memory state at time t, (shown in gold), is then used to decode object centers, and estimate their visibility (other outputs directly adapted from [66] are not shown for readability).

is recurrently passed through separate sub-networks $f_{\mathbf{p}}, f_{off}, f_s, f_{\mathbf{d}}$, which following [66], are used to decode the bounding boxes of the objects and link them into tracks. We augment those with a new visibility head $f_{\mathbf{v}}$, which produces an output map $V^t \in [0, 1]^{H \times W}$. It is a binary classifier, predicting whether the object center at a particular location corresponds to a visible, or a fully occluded instance. See, for example, the person walking behind the red SUV on the right in Figure 3. His location behind occlusion is supervised as a positive for the localization head P^t, but as a negative for V^t. This distinction is important for evaluation, since annotations for invisible objects are not provided (we also remove them from the validation set of PD for a fair comparison). To avoid being penalized for 'false positive' predictions, we only output bounding boxes that are classified as visible by our model, but use the invisible ones to recover object identities after occlusions.

All the operations detailed here are fully differentiable, and thus the model can be trained in an end-to-end fashion with backpropagation through time [58]. Following [66], we use focal loss [36] to supervise P^t and V^t, and L1 loss for the latter three heads. The overall training objective is:

$$L = \frac{1}{N} \sum_{t=1}^{N} L_{\mathbf{p}}^t + L_{\mathbf{v}}^t + \lambda_{off} L_{off}^t + \lambda_s L_s^t + \lambda_{\mathbf{d}} L_{\mathbf{d}}^t, \quad (1)$$

where N is the length of the input sequence, and $\lambda_{off}, \lambda_s, \lambda_{\mathbf{d}}$ are hyper-parameters that balance the contribution of the corresponding losses in the overall objective.

As we discuss in Section 4.3, training our model on visible objects alone results in noticeable improvements over [66] due to increased robustness to noise in instantaneous observations. Next, we discuss our approach to supervising fully occluded objects.

3.3. Learning to track behind occlusions

3.3.1 Disambiguating levels of visibility

To generate training labels for a video sequence of length N, our method takes object annotations $\{O^1, O^2, ..., O^N\}$,

with $O^t = \{o_1^t, o_2^t, ..., o_m^t\}$ as input. Each object $o_i^t = (\mathbf{p}, \mathbf{s}, id, vis)$ is described by its center $\mathbf{p} \in \mathbb{R}^2$, bounding box size $\mathbf{s} \in \mathbb{R}^2$, identity $id \in \mathbb{I}$, which is used, together with $\mathbf{p}$, to supervise the displacement vectors $\mathbf{d}$, and visibility level $vis \in [0, 1]$, indicating what fraction of the object is visible in the current frame. Naively, one could simply ignore the visibility levels and supervise all the objects in every frame. This would, however, result in the model being forced to hallucinate objects trajectories before they first become visible (e.g. the car driving in front of the truck in Figure 2 for the whole duration of the video). As we show in our experiments (Table 2), such supervision is effectively label noise and decreases model's performance.

To avoid this, we pre-process the annotations to only start supervising occluded objects after they have been visible for at least 2 frames. This is the minimal sufficient time for the model to both localize an object and estimate its velocity - the required information for predicting its position under occlusion.

Concretely, we introduce two thresholds T_{vis} and T_{occl}. Then, starting from the first frame in a sequence O^1, for every object o_i^1, if $vis_i^1 < T_{vis}$ the object is treated as a negative, if $T_{vis} < vis_i^1 < T_{occl}$ it is ignored (the model is not penalized for predicting it), and finally, if $vis_i^1 > T_{occl}$ its marked as visible and used to produce the labels. The same procedure is repeated for every frame in a sequence, with the only difference that, starting from frame 3, objects that were marked as visible for two consecutive frames in the past are treated as positives regardless of their visibility status in the current frame. This procedure reduces the ambiguity of supervision for the model. In particular, the second threshold T_{occl} allows for a soft transition between visible and invisible objects.

3.3.2 Supervising the invisible

The ambiguity of the location of an invisible object is not fully addressed by the algorithm above. Consider the ground truth trajectory of a person shown in white in Figure 4. She walks behind the bus stop, and then stops. In
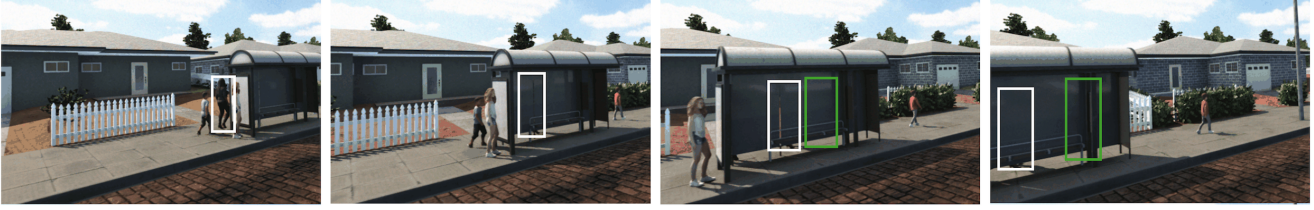
Figure 4. Illustration of the ambiguity of ground truth object locations under full occlusions. The woman, shown in white, walks behind the bus stop and then stops. Instead of trying to predict this random event, we propose to supervise the model with deterministic pseud-ground-truth in such scenarios (shown in green, matches the ground truth in the first two frames).

the absence of observations it is impossible for the model to predict this behavior. Consequently, such examples also constitute label noise. In fact, the only deterministic assumption both a person and a neural network can make about the trajectory of an occluded object is that it will maintain constant velocity.

Propagating the last observed object location with its constant velocity in the camera frame is also the state of the art approach for handling occlusions in multi-object tracking literature [7, 66]. It is however, not robust to changes in camera motion. Instead, we propose to generate pseudo-ground-truth labels for supervising our model by propagating the occluded object locations with their last observed velocity in 3D, and projecting the resulting centers to the camera frame, which is made possible by the availability of the full ground truth information in our synthetic dataset.

Concretely, for an object i getting occluded at time t, we take its ground truth centers in the previous two frames in the world coordinate system $\mathbf{P}_i^{t-1}, \mathbf{P}_i^{t-2} \in \mathbb{R}^3$, and compute the object velocity $\mathbf{V}_i = \mathbf{P}_i^{t-1} - \mathbf{P}_i^{t-2}$. We then use it to estimate the location of the object center at time t under the constant velocity assumption via $\tilde{\mathbf{P}}_i^t = \mathbf{P}_i^{t-1} + \mathbf{V}_i$. Finally, this estimated center is projected to the camera frame via $\tilde{\mathbf{p}}_i^t = \mathbf{K}[\mathbf{R}|\mathbf{t}]\tilde{\mathbf{P}}_i^t$, where $\mathbf{K}$ is the camera intrinsics, and $[\mathbf{R}|\mathbf{t}]$ is the camera extrinsic matrix, and is used to replaced the ground-truth center $\mathbf{p}_i^t$ in the corresponding label set o_i^t. The same procedure is repeated for all the frames during which the object remains invisible. This principled approach, shown in green in Figure 4, results in a deterministic supervision for invisible objects.

If the object does not re-appear at the expected location, the model is supervised to keep hallucinating the box as described above. Re-identification is then performed at the time of dis-occlusion t by supervising the displacement vector d_t to project the ground-truth, visible location to the hallucinated location at $t - 1$.

3.4. Bridging the sim-to-real domain gap

Analyzing the approaches for supervising tracking behind occlusion described above requires a large video dataset with objects densely labeled regardless of whether they are visible, together with precise visibility scores, 3D

coordinates and camera matrices. No real dataset with such labels exists due to the cost and complexity of collecting these annotations.

Instead of going the expensive route, we use synthetic data for which it is easy to automatically generate physically accurate ground truth. However, generalization to real videos remains a challenge. A few approaches that used synthetic videos for training in the past addressed the domain discrepancy by simply finetuning the resulting model on a small real dataset [22, 29, 55]. Recall, however, that real datasets in our scenario do not provide consistent annotations for occluded objects, thus such finetuning would result in forgetting the trajectory hallucination behaviour.

To mitigate this issue, we propose to jointly train our approach on synthetic and real data, where at each iteration a batch is sampled from one of the datasets at random. Moreover, we cannot use real sequences of length more than 2 for the same reason mentioned above (we want the invisible objects supervision to remain consistent). As a result, we sample synthetic clips of length N and real ones of length 2 during training, with the final loss being:

$$L = \frac{1}{N}\sum_{t=1}^{N} L_{sym}^t + \frac{1}{2}\sum_{t=1}^{2} L_{real}^t, \qquad (2)$$

where L_{sym} and L_{sym} are defined in the same way as in Equation 1. Effectively, we use synthetic videos to learn the desired behavior, and real frame pairs to minimize the domain discrepancy.

4. Experiments

4.1. Datasets and evaluation

We use two real datasets in the experimental analysis to compare to prior work: KITTI [24] and MOT17 [38]. In addition, we use a virtual dataset - ParallelDomain (PD), to learn to track behind occlusions. Details of these datasets are provided in the supplementary material. While our work is focused on the 2D setting, our approach can be directly extended to 3D tracking, as we show on the nuScenes [12] benchmark in the supplementary.

Evaluation metrics Traditionally, multi-object tracking datasets have been using the CLEAR MOT metrics for eval-

uation [8], with MOTA being the main metric used to compare the methods. It has recently been shown, however, that it overemphasizes detection over association [37]. Instead, supplementary metrics, such as fraction of tracks that are maintained for at least 80% of their duration ('mostly tracked') have to be used in conjunction with MOTA.

The lack of a single metric that combines detection and association accuracy has been addressed by track intersection-over-union (IoU) based metrics [17, 49, 63], and HOTA [37]. In our analysis we use the former due to its larger emphasis on association accuracy. To formally define track IoU, let $G = \{g_1, \ldots, g_T\}$ and $D = \{d_1, \ldots, d_T\}$ be a groundtruth and a corresponding predicted track for a video with T frames. Importantly, only one predicted track can be assigned to each ground truth trajectory, and all the unassigned ones are treated as false positives. Track IoU is then defined as: $\text{IoU}(D, G) = \frac{\sum_{t=1}^{T} g_t \cap d_t}{\sum_{t=1}^{T} g_t \cup d_t}$. Similarly to standard object detection metrics, track IoU together with track confidence can be used to compute mean average precision (mAP) across categories using a predefined IoU threshold. Following [17] we use a threshold of 0.5, referring to this metric as Track AP. When comparing to the state of the art, we report the standard metrics for each dataset.

4.2. Implementation details

For the components of our model shared with Center-Track [66] we follow their architecture and protocol exactly. Here we only provide the values of the new hyperparameters. Additional details, such as joint training on synthetic and real data, are reported in the supplementary.

The ConvGRU has a feature dimension of 256, and uses 7×7 convolutions. We train the model on synthetic sequences of length 17 for 21 epochs with a batch size 16 using the Adam optimizer. The choice of the clip length is defined by GPU memory constraints. The learning rate is set to $1.25e-4$ and decreased by a factor of 10 every 7 epochs for 1 epoch. It is then increased back to the original value. We have found such a periodic schedule to speed up convergence. We set the visibility threshold T_{vis} to 0.05 and the occlusion threshold T_{occl} to 0.15, corresponding to 5% and 15% of the object being visible respectively. During evaluation, our model is applied in a fully online way, processing all the frames from a video one by one. It runs on a single Tesla V100 GPU at around 10FPS.

4.3. Ablation analysis

In this section, we analyze our proposed approach. All the variants are trained using exactly the same hyperparameters and learning rate schedules. Invisible object annotations are ignored in the validation set of PD, so only visible parts of the trajectories are used for evaluation. This allows to fairly compare methods that do and do not have access to invisible object labels during training.

	GRU	H^{t-1}	Input	Car AP	Person AP	mAP
CenterTrack	-	✓	2 fr	66.2	54.4	60.3
Ours	3×3	✓	2 fr	64.6	49.7	57.1
Ours	7×7	✓	2 fr	65.2	54.0	59.6
Ours	7×7	✗	2 fr	65.7	55.6	60.6
Ours	7×7	✗	Video	**66.8**	**57.9**	**62.4**

Table 1. Analysis of the architecture of our our model using Track AP on the validation set of PD. We ablate the effect of the filter size in ConvGRU, explicitly passing the encoding of object centers in the previous frame H^{t-1}, and training on videos vs frame pairs.

	Invis. sup.	Post-proc.	Car AP	Person AP	mAP
Ours	All GT	-	66.0	58.3	62.2
Ours	Filtered GT	-	71.1	60.6	65.9
Ours	2D const v	-	70.7	60.8	65.7
Ours	3D const v	-	**71.0**	**63.0**	**67.0**
CenterTrack	-	2D const v	67.6	54.9	61.2
Ours	3D const v	2D const v	**72.7**	**63.1**	**67.9**

Table 2. Comparison of different approaches for handling full occlusions using Track AP on the validation set of PD. We evaluate several supervision strategies, and compare the best variant to the heuristic-based constant velocity track propagation.

Model variants. We begin by studying the variants of our video-level tracking model. To this end, we first train CenterTrack on PD using the code provided by the authors and report the results in row 1 of Table 1. The basic variant our our model, shown in row 2, like CenterTrack, takes pairs of frames as input together with H^{t-1}, the encoding of detections in the previous frame, but processes them recurrently with a ConvGRU. It performs significantly worse than the baseline, due to the fact that CenterTrack uses a deep network with a large field-of-view to combine the two frames and establish associations between the objects, whereas our model relies on a few convolutional layers in the ConvGRU with 3×3 filters. Increasing the filter size (row 3 in Table 1) indeed results in a noticeable performance improvement.

Next, we observe that the additional H^{t-1} input is redundant in our case. Moreover, removing this input allows to avoid the corresponding heat map augmentation strategy proposed by [66]. This simplification (show in row 4 in Table 1) further improves the performance of our model, matching CenterTrack. Finally, training and evaluating on longer sequences (last row in Table 1) unlocks the full potential of our architecture to capture object permanence in videos. Even when trained only on visible objects, this variant significantly outperforms the baseline due to its robustness to the noise in instantaneous observations.

Tracking behind occlusions. In Table 2 we now compare various strategies for supervising full occlusion scenarios. Firstly, we observe that the naive approach of training the model to detect and track all the invisible object results in a slight decrease in performance compared to variant trained only on visible ones (last row in Table 1). As discussed in Section 3.3, such supervision is highly ambiguous, as the model cannot localize the fully occluded objects it has not seen before. Accounting for this fact with our proposed an-
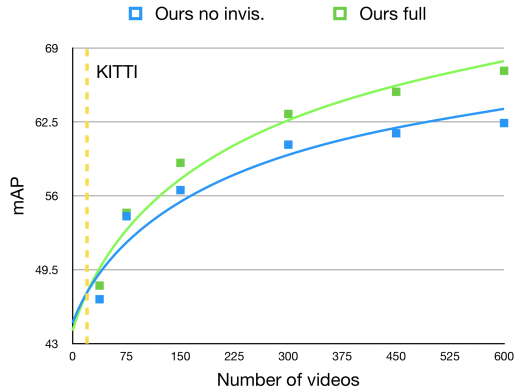
Figure 5. Evaluation of the effect of the number of training videos using Track mAP on the validation set of PD. The gap between our full approach (shown in green), and the variant trained without invisible object labels (shown in blue), increases consistently with the dataset size. This demonstrates that a large dataset is required to learn to hallucinate trajectories of fully occluded objects, and KITTI (shown in yellow) would not be sufficient.

notation filtering strategy (second row in Table 2) results in a 3.5 mAP improvement, validating its importance.

Next, we compare using ground truth location of invisible object to pseudo-ground-truth obtained via propagating the occluded object with its constant velocity in 2D, or in 3D (see Section 3.3 for details). Constant velocity in 2D (row 3 in Table 2) is not robust to camera changes and results in a lower performance than the ground truth locations. In contrast, propagating the target with its last observed velocity in 3D world coordinates (row 4 in the table), results in labels which are both consistent with the observations and fully deterministic, further improving the performance.

Finally, we compare our learning-based approach to the constant velocity post-processing, which is a common way of handling occlusions in the tracking literature [7, 66], in the last two rows in Table 2. Firstly, this heuristic-based step does indeed improve the performance of [66], but it remains 5.8 mAP points below our principled method. Secondly, applying this post-processing to the outputs of our method also improves its performance. We do not use any post-processing in the remainder of the experiments.

Effect of the dataset size. In Figure 5 we plot the validation performance of two variants of our model: the full one, and the one trained on visible objects only, while increasing the number of videos in the training set of PD. One can easily see that the gap between the two variants consistently increases. In fact, below 75 videos it is close to 0. This demonstrates that a large number of examples is required to learn the tracking behind occlusions behaviour.

Yellow dashed line corresponds to the number of videos in the training set of KITTI, and illustrates that even if this dataset provided all required annotations, it would not be sufficient to train our model. MOT17 contains only 7 train-

	KITTI	PD	Car AP	Person AP	mAP
CenterTrack [66]	✓	✗	73.8	39.4	56.6
Ours	✗	✓	83.3	38.2	60.8
Ours (tune)	✓	✓	75.7	44.6	60.2
Ours (joint)	✓	✓	**84.7**	**56.3**	**70.5**
CenterTrack (joint)	✓	✓	77.2	51.6	64.4

Table 3. Domain adaptation analysis using Track AP on the validation set of KITTI. We demonstrate the effectiveness of our simple strategy, and confirm that the improvements mainly come from better occlusion handling.

ing videos, making it even less practical for our purposes.

Domain adaptation. We now demonstrate how the model learned on the large-scale, synthetic PD dataset can be transferred to a real-world benchmark such as KITTI in Table 3, and compare to the CenterTrack model released by the authors. Firstly, we directly evaluate our model trained on PD (fourth row in Tabel 2) and report the results in the second row of Table 3. Despite the significant domain gap between the synthetic and real videos, this variant manages to outperform CenterTrack without seeing a single frame from KITTI. The gap in performance on the Person category remains large, due to its higher visual variability.

Directly fine-tuning our model on KITTI (shown in the third row in the table), helps to reduce the domain discrepancy, improving the Person performance, but also results in un-learning the tracking behind occlusions behaviour, as reflected in the drop in Car AP. In contrast, jointly training on the two datasets achieves the best results overall, validating the effectiveness of the proposed approach.

Finally, we apply our domain adaptation approach to the CenterTrack model trained on PD (first row in Table 1), and report the results in the last row of Table 3. One can see that synthetic data also improves CenterTrack results, but they remain 6.1 mAP point below those of our approach. This demonstrates that the improvements mainly come from our model's ability to better handle occlusions.

4.4. Comparison to the state of the art

In this section, we compare our full method, which we refer to as PermaTrack, to the state of the art on the KITTI [24], and MOT17 [38] benchmarks. All our models are pre-trained on PD and adapted to the real domain using the training set of the corresponding dataset.

KITTI. Table 4 lists the results on the KITTI test set, comparing to vision-based, online methods. Our method outperforms the state of the art on all metrics on both categories, except for ML (Mostly Lost) on Car, where we are 0.4 points below [43]. Notice that we are 5.7 point above this method on the main HOTA metric, and are outperforming it by a large margin on all metrics on Person. Our improvements over the state-of-the-art CenterTrack are 5 HOTA points on Car and 8.4 on Person, which is notable, given that typical differences between the methods on the

Figure 6. Qualitative results on a test sequence from the KITTI benchmark. Our approach is able to successfully track a fully occluded car on the left (`id 6`, highlighted in white). Please see supplementary video for a more comprehensive analysis on both datasets.

	Car					Person				
	HOTA ↑	MOTA ↑	MT ↑	PT ↓	ML↓	HOTA ↑	MOTA ↑	MT ↑	PT ↓	ML ↓
MASS [32]	68.3	84.6	74.0	23.1	2.9	-	-	-	-	-
IMMDP [60]	68.7	82.8	60.3	27.5	12.2	-	-	-	-	-
AB3D [57]	69.8	83.5	67.1	21.5	11.4	35.6	38.9	17.2	41.6	41.2
TuSimple [15]	71.6	86.3	71.1	22.0	6.9	45.9	57.6	30.6	44.3	25.1
SMAT [25]	71.9	83.6	62.8	31.2	6.0	-	-	-	-	-
TrackMPNN [43]	72.3	87.3	84.5	13.4	**2.2**	39.4	52.1	35.1	46.1	18.9
CenterTrack [66]	73.0	88.8	82.2	15.4	2.5	40.4	53.8	35.4	43.3	21.3
PermaTrack (Ours)	**78.0**	**91.3**	**85.7**	**11.7**	2.6	**48.6**	**66.0**	**48.8**	**35.4**	**15.8**

Table 4. Comparison to the state of the art on the test set of the KITTI benchmark using aggregate metrics. Some methods only report results on Car. Our approach outperforms all the other methods by a significant margin on all the metrics except for ML on Car.

		T.R.	IDF1 ↑	MOTA ↑	MT ↑	PT ↓	ML ↓
Public	CenterTrack [66]	✗	63.2	63.1	37.5	38.1	24.5
	PermaTrack (Ours)	✗	67.0	67.8	**43.7**	**36.3**	20.1
	CenterTrack [66]	✓	66.4	63.8	37.2	38.1	24.8
	PermaTrack (Ours)	✓	**71.1**	**68.2**	41.0	39.5	**19.5**
Private	CenterTrack [66]	✗	64.2	66.1	41.3	37.5	21.2
	PermaTrack (Ours)	✗	68.2	69.4	**46.3**	**36.0**	17.7
	CenterTrack [66]	✓	69.4	66.2	39.5	38.3	22.1
	PermaTrack (Ours)	✓	**71.9**	**69.5**	42.5	**39.8**	17.7

Table 5. Comparison to the state of the art on the validation set of the MOT17 using private and public detections. Our method outperforms the state-of-the-art CenterTrack approach on all metrics both with and without the Track Rebirth (T.R.) post-processing.

leader-board are within 1 point, and that CenterTrack is pre-trained on a large-scale, *real-world* nuScenes dataset [12].

MOT17. The policy on this dataset is that only methods that don't use external data for training can be evaluated on the test server with public detections. With just 7 videos in the training set, MOT17 is not sufficient to learn the complex tracking behind occlusions behaviour. For fairness, we compare our method to the variant of the state-of-the-art CenterTrack which is pre-trained on the *real-world* Crowd-Human dataset [52] using the validation set of MOT17 in Table 5. The results on the test set using private detections are reported in the supplementary material.

Without post-processing, our approach outperforms [66] using both public and private detections. The improvements are especially significant in the public evaluation (the same bounding boxes used by both methods), emphasizing our method's better tracking abilities. Finally, adding Track Re-birth post-processing from [66] (T.R. in the table, a variant

of constant velocity) improves the performance of both approaches and does not change the conclusions.

Qualitative results. We provide an example of the output of our method on a sequence from the test set of KITTI in Figure 6. In this challenging scenario, as the ego-vehicle drives forward, the car on the left (`id 6`, highlighted in white) is fully occluded by another moving vehicle (`id 2`), but our method manages to correctly localize it and maintain the trajectory. A more comprehensive analysis of our approach is presented in the supplementary video[1].

5. Conclusion

In this work, we propose PermaTrack, an end-to-end-trainable approach for joint object detection and tracking. Thanks to its recurrent memory module, it is capable of reasoning about the location of objects using the entire previous history, and not just the current observation. Combined with supervision from synthetic data, this allows to train the model to track objects when they are fully occluded - a key concept in cognitive science known as object permanence.

Our method obtains state-of-the-art results on the KITTI and MOT17 multi-object tracking benchmarks. While the ablation analysis demonstrates that hallucinating trajectories of invisible objects is a crucial factor in the final performance, knowledge about the full history also increases robustness to partial occlusions, and other low visibility scenarios, such as motion blur.

---

[1]https://www.youtube.com/watch?v=Dj2gSJ_xILY

References

[1] Parallel domain. https://paralleldomain.com/, March 2021. 2

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *CVPR*, 2016. 2

[3] Renee Baillargeon, Elizabeth S Spelke, and Stanley Wasserman. Object permanence in five-month-old infants. *Cognition*, 20(3):191–208, 1985. 1

[4] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. In *ICLR*, 2016. 2, 3

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

[6] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011. 1, 2

[7] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 1, 2, 5, 7

[8] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 6

[9] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 1, 2

[10] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *CVPR*, 2020. 1, 2

[11] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*, 2009. 1, 2

[12] 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. In *CVPR*, 2020. 5, 8

[13] Boyu Chen, Dong Wang, Peixia Li, Shuang Wang, and Huchuan Lu. Real-time 'actor-critic' tracking. In *ECCV*, 2018. 2

[14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014. 3

[15] Wongun Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015. 8

[16] Wongun Choi and Silvio Savarese. Multiple target tracking in world coordinate with single, minimally calibrated camera. In *ECCV*, 2010. 2

[17] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. TAO: A large-scale benchmark for tracking any object. In *ECCV*, 2020. 6

[18] César Roberto de Souza, Adrien Gaidon, Yohann Cabon, Naila Murray, and Antonio Manuel López. Generating human action videos by coupling 3D game engines and probabilistic graphical models. *IJCV*, pages 1–32, 2019. 3

[19] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *CVPR*, 2015. 3

[20] Matteo Fabbri, Fabio Lanzi, Simone Calderara, Andrea Palazzi, Roberto Vezzani, and Rita Cucchiara. Learning to detect and track visible and occluded body joints in a virtual world. In *ECCV*, 2018. 3

[21] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, 2017. 2

[22] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016. 3, 5

[23] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 3

[24] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 1, 2, 5, 7

[25] Nicolas Franco Gonzalez, Andres Ospina, and Philippe Calvez. SMAT: Smart multiple affinity metrics for multiple object tracking. In *ICIAR*, 2020. 8

[26] Helmut Grabner, Jiri Matas, Luc Van Gool, and Philippe Cattin. Tracking the invisible: Learning where the object might be. In *CVPR*, 2010. 2

[27] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 3

[28] Yan Huang and Irfan Essa. Tracking multiple objects through occlusions. In *CVPR*, 2005. 2

[29] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 3, 5

[30] Hao Jiang, Sidney Fels, and James J Little. A linear programming approach for multiple object tracking. In *CVPR*, 2007. 2

[31] Kai Kang, Hongsheng Li, Tong Xiao, Wanli Ouyang, Junjie Yan, Xihui Liu, and Xiaogang Wang. Object detection in videos with tubelet proposal networks. In *CVPR*, 2017. 2

[32] Hasith Karunasekera, Han Wang, and Handuo Zhang. Multiple object tracking with attention to appearance, structure, motion and size. *IEEE Access*, 7:104423–104434, 2019. 8

[33] Jiaqi Zou Ke Mei, Chuang Zhu and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation. In *ECCV*, 2020. 3

[34] Laura Leal-Taixé, Michele Fenzi, Alina Kuznetsova, Bodo Rosenhahn, and Silvio Savarese. Learning an image-based motion context for multiple people tracking. In *CVPR*, 2014. 1, 2

[35] Kuan-Hui Lee, German Ros, Jie Li, and Adrien Gaidon. SPIGAN: privileged adversarial learning from simulation. In *ICLR*, 2019. 3

[36] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 4

[37] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. HOTA: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, pages 1–31, 2020. 6

[38] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 2, 5, 7

[39] Anton Milan, Seyed Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *AAAI*, 2017. 2

[40] Dennis Mitzel, Esther Horbert, Andreas Ess, and Bastian Leibe. Multi-person tracking with sparse detection and continuous segmentation. In *ECCV*, 2010. 2

[41] Vasilis Papadourakis and Antonis Argyros. Multiple objects tracking in the presence of long-term occlusions. *Computer Vision and Image Understanding*, 114(7):835–846, 2010. 2

[42] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 1, 2

[43] Akshay Rangesh, Pranav Maheshwari, Mez Gebre, Siddhesh Mhatre, Vahid Ramezani, and Mohan M Trivedi. TrackMPNN: A message passing graph neural architecture for multi-object tracking. *arXiv preprint arXiv:2101.04206*, 2021. 7, 8

[44] Liangliang Ren, Jiwen Lu, Zifeng Wang, Qi Tian, and Jie Zhou. Collaborative deep reinforcement learning for multi-object tracking. In *ECCV*, 2018. 2

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

[46] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 3

[47] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. In *CVPR*, 2018. 2

[48] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The Synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 3

[49] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6

[50] Antoine Saporta, Tuan-Hung Vu, M. Cord, and P. Pérez. Esl: Entropy-guided self-supervised learning for domain adaptation in semantic segmentation. *ArXiv*, 2020. 3

[51] Aviv Shamsian, Ofri Kleinfeld, Amir Globerson, and Gal Chechik. Learning object permanence from video. In *ECCV*, 2020. 2

[52] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. CrowdHuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018. 8

[53] Elizabeth S Spelke. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990. 1

[54] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, 2017. 1, 2

[55] Pavel Tokmakov, Cordelia Schmid, and Karteek Alahari. Learning to segment moving objects. *International Journal of Computer Vision*, 127(3):282–301, 2019. 3, 5

[56] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Mathieu Cord, and Patrick Pérez. DADA: Depth-aware domain adaptation in semantic segmentation. In *ICCV*, 2019. 3

[57] Xinshuo Weng and Kris Kitani. A baseline for 3D multi-object tracking. *arXiv preprint arXiv:1907.03961*, 2019. 8

[58] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. 4

[59] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017. 1, 2

[60] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi- object tracking by decision making. In *ICCV*, 2015. 8

[61] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *ECCV*, 2018. 3

[62] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *ICCV*, 2019. 1, 2

[63] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 6

[64] Qian Yu, Gérard Medioni, and Isaac Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *CVPR*, 2007. 2

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

[66] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[67] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 3

[68] Yang Zou, Zhiding Yu, Xiaofeng Liu, B.V.K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. In *ICCV*, 2019. 3

[69] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 3