Learning Multi-Object Tracking and Segmentation from Automatic Annotations

Lorenzo Porzi†, Markus Hofinger‡, Idoia Ruiz∗, Joan Serrat∗, Samuel Rota Bulò†, Peter Kontschieder†
Mapillary Research†, Graz University of Technology‡, Computer Vision Center, UAB∗
research@mapillary.com†, markus.hofinger@icg.tugraz.at‡, {iruiz,joans}@cvc.uab.es∗

Figure 1: KITTI sub-sequences with automatically generated MOTS annotations as color-coded instance masks (left to right).

Abstract

In this work we contribute a novel pipeline to automatically generate training data, and to improve over state-of-the-art multi-object tracking and segmentation (MOTS) methods. Our proposed track mining algorithm turns raw street-level videos into high-fidelity MOTS training data, is scalable and overcomes the need of expensive and time-consuming manual annotation approaches. We leverage state-of-the-art instance segmentation results in combination with optical flow predictions, also trained on automatically harvested training data. Our second major contribution is MOTSNet—a deep learning, tracking-by-detection architecture for MOTS—deploying a novel mask-pooling layer for improved object association over time. Training MOTSNet with our automatically extracted data leads to significantly improved sMOTSA scores on the novel KITTI MOTS dataset (+1.9%/+7.5% on cars/pedestrians), and MOTSNet improves by +4.1% over previously best methods on the MOTSChallenge dataset. Our most impressive finding is that we can improve over previous best-performing works, even in complete absence of manually annotated MOTS training data.

1. Introduction and Motivation

We focus on the challenging task of multi-object tracking and segmentation (MOTS) [43], which was recently introduced as an extension of bounding-box based multi-object tracking. Joining instance segmentation with tracking was shown to noticeably improve tracking performance, as unlike bounding boxes, instance segmentation masks do not suffer from overlapping issues but provide fine-grained, pixel-level information about objects to be tracked.

While this finding is encouraging, it comes with the downside of requiring pixel-level (typically polygon-based) annotations, which are known to be time-consuming in generation and thus expensive to obtain. The works in [9, 29] report annotation times of ≈90 minutes per image to manually produce high-quality panoptic segmentation masks. Analogously, it is highly demanding to produce datasets for the MOTS task where instance segmentation masks need to also contain tracking information across frames.

To avoid generating MOTS labels completely from scratch and in a purely manual way, [43] has followed the semi-automatic annotation procedure from [4], and extended the existing multi-object tracking datasets KITTI tracking [11] (i.e. 21 sequences from the raw KITTI dataset) and MOTChallenge [28] (4/7 sequences). These datasets already provide bounding box-based tracks for cars and pedestrians. Instance segmentation masks were generated as follows. First, two segmentation masks per object were generated by human annotators (which also had to choose the objects based on diversity in the first place). Then, DeepLabV3+ [6], i.e. a state-of-the-art semantic segmentation network, was trained on the initially generated masks in a way to overfit these objects in a sequence-specific way, thus yielding reasonable segmentation masks for the remaining objects in each track. The resulting segmentation masks then underwent another manual correction step to fix remaining errors. Finally, these automatic and manual steps were iterated until convergence.

Including an automated segmentation algorithm is clearly speeding up the MOTS dataset generation process, but still has significant shortcomings. Most importantly, the approach in [43] still depends on the availability of bounding-box based, multi-object tracking information as provided by datasets like [11, 28]. Also, their remaining human annotation effort for generating initial instance
masks was considerable, i.e. [43] reports that ≈8k masks (or 12.3% of all masks in KITTI MOTS) had been manually labeled before fine-tuning a (modified) DeepLabV3+ segmentation network, pre-trained on COCO [21] and Mapillary Vistas [29]. Finally, such an approach cannot be expected to generalize well for generating orders of magnitude more training data.

In this paper we introduce a novel approach for automatically generating high-quality training data (see Fig. 1 for an example) from generic street-level videos for the task of joint multi-object tracking and segmentation. Our methods are conceptually simple and leverage state-of-the-art instance [12] (or Panoptic [8, 35]) segmentation models trained on existing image datasets like Mapillary Vistas [29] for the task of object instance segmentation. We further describe how to automatically mine tracking labels for all detected object instances, building upon state-of-the-art optical flow models [49], that in turn have been trained on automatically harvested flow supervision obtained from image-based 3d modeling (structure-from-motion).

Another important contribution of our paper is MOTSNet, a deep learning, tracking-by-detection approach for MOTS. MOTSNet takes advantage of a novel mask-pooling layer, which allows it to exploit instance segmentation masks to compute a representative embedding vector for each detected object. The tracking process can then be formulated as a series of Linear Assignment Problems (LAP), optimizing a payoff function which compares detections in the learned embedding space.

We demonstrate the superior quality for each step of our proposed MOTS training data extraction pipeline, as well as the efficacy of our novel mask-pooling layer. The key contributions and findings of our approach are:

- Automated MOTS training data generation – instance detection, segmentation and actual tracking – is possible at scale and with very high fidelity, approximating the quality of previous state-of-the-art results [43] without even directly training for it.
- Deep-learning based, supervised optical flow [49] can be effectively trained on pixel-based correspondences obtained from Structure-from-Motion and furthermore used for tracklet extraction.
- Direct exploitation of instance segmentation masks through our proposed mask-pooling layer yields to significant improvements of sMOTSA scores
- The combination of our general-purpose MOTS data generation pipeline with our proposed MOTSNet improves by up to 1.2% for cars and 7.5% for pedestrians over the previously best method on KITTI MOTS in absolute terms, while using 13.7% fewer parameters.

We provide extensive ablation studies on the KITTI MOTS, MOTSChallenge, and Berkeley Deep Drive BDD100k [50] datasets and obtain results consistently improving with a large margin over the prior state-of-the-art [43].

2. Related Works

Since the MOTS task was only very recently introduced in [43], directly related works are scarce. Instead, the bulk of related works in a broader sense comes from the MOT (multi-object tracking) and VOS (video object segmentation) literature. This is true in terms of both, datasets (see e.g. [11, 18, 28] for MOT and [20, 33, 36] for VOS/VOT, respectively), and methods tackling the problems ([40] for MOT and [25, 31, 44, 48] for VOS).

Combining motion and semantics. The work in [16] uses semantic segmentation results from [23] for video frames together with optical flow predictions from EpicFlow [37] to serve as ground truth, for training a joint model on future scene parsing, i.e. the task of anticipating future motion of scene semantics. In [24] this task is extended to deal with future instance segmentation prediction, based on convolutional features from a Mask R-CNN [12] instance segmentation branch. In [15] a method for jointly estimating optical flow and temporally consistent semantic segmentation from monocular video is introduced. Their approach is based on a piece-wise flow model, enriched with semantic information through label consistency of superpixels.

Semi-automated dataset generation. Current MOT [11, 28, 45] and VOS [34, 47] benchmarks are annotated based on human efforts. Some of them, however, use some kind of automation. In [47], they exploit the temporal correlation between consecutive frames in a skip-frame strategy, although the segmentation masks are manually annotated. A semi-automatic approach is proposed in [1] for annotating multi-modal benchmarks. It uses a pre-trained VOS model [17] but still requires additional manual annotations and supervision on the target dataset. The MOTS training data from [43] is generated by augmenting an existing MOT dataset [11, 28] with segmentation masks, using the iterative, human-in-the-loop procedure we briefly described in Sec. 1. The tracking labels on [11, 28] were manually annotated and, in the case of [28], refined by adding high-confidence detections from a pedestrian detector.

The procedures used in these semi-automatic annotation approaches often resemble those of VOS systems which require user input to select objects of interest in a video. Some proposed techniques in this field [5, 7, 27] are extensible to multiple object scenarios. A pixel-level embedding is built in [7] to solve segmentation and achieve state-of-the-art accuracy with low annotation and computational cost.

To the best of our knowledge, the only work that generates tracking data from unlabeled large-scale videos is found in [32]. By using an adaptation of the approach in [31], Osep et al. perform object discovery on unlabeled videos from the KITTI Raw [10] and Oxford RobotCar [26] datasets. Their approach however cannot provide joint an-
notations for instance segmentation and tracking.

**Deep Learning methods for MOT and MOTS.**

Many deep learning methods for MOTS/VOS are based on “tracking by detection”, i.e. candidate objects are first detected in each frame, and then joined into tracks as a post-processing step (e.g. [42]). A similar architecture, i.e. a Mask R-CNN augmented with a tracking head, is used in [48] to tackle Video Instance Segmentation. A later work [25] improved over [48] by adapting the UnOvOST [51] model that, first in an unsupervised setting, uses optical flow to build short tracklets later merged by a learned embedding. For the plain MOT task, Sharma et al. [40] incorporated shape and geometry priors into the tracking process.

Other works instead, perform detection-free tracking. CAMOT [31] tracks category-agnostic mask proposals across frames, exploiting both appearance and geometry cues from scene flow and visual odometry. Finally, the VOS work in [44], follows a correlation-based approach, extending popular tracking models [3, 19]. Their SiamMask model only requires bounding box initialization but is pre-trained on multiple, human-annotated datasets [21, 39, 47].

### 3. Dataset Generation Pipeline

Our proposed data generation approach is rather generic w.r.t. its data source, and here we focus on the KITTI Raw [10] dataset. KITTI Raw contains 142 sequences (we excluded the 9 sequences overlapping with the validation set of KITTI MOTS), for a total of ~44k images, captured with a professional rig including stereo cameras, LiDAR, GPS and IMU. Next we describe our pipeline which, relying only on monocular images and GPS data is able to generate accurate MOTS annotations for the dataset.

#### 3.1. Generation of Instance Segmentation Results

We begin with segmenting object instances in each frame per video and consider a predefined set of 37 object classes that belong to the Mapillary Vistas dataset [29]. To extract the instance segments, we run the Seamless Scene Segmentation method [35], augmented with a ResNeXt-101-32×8d [46] backbone and trained on Mapillary Vistas. By doing so, we obtain a set of object segments per video sequence. For each segment $s \in S$, we denote by $t_s \geq 0$ the frame where the segment was extracted, by $y_s \in \mathcal{Y}$ the class it belongs to and by $\phi_s : \mathbb{R}^2 \rightarrow \{0,1\}$ a pixel indicator function representing the segmentation mask, i.e. $\phi_s(i,j) = 1$ if and only if pixel $(i,j)$ belongs to the segment. For convenience, we also introduce the notation $S_t$ to denote the set of all segments extracted from frame $t$ of a given video. For KITTI Raw we roughly extracted 1.25M segments.

#### 3.2. Generation of Tracklets using Optical Flow

After having automatically extracted the instance segments from a given video in the dataset, our goal is to leverage on optical flow to extract tracklets, i.e. consecutive sequences of frames where a given object instance appears.

**Flow training data generation.** We automatically generate ground-truth data for training an optical flow network by running a Structure-from-Motion (SfM) pipeline, namely OpenSfM, on the KITTI Raw video sequences and densify the result using PatchMatch [41]. To further improve the quality of the 3d reconstruction, we exploit the semantic information that has been already extracted per frame to remove spurious correspondences generated by moving objects. Consistency checks are also performed in order to retain correspondences that are supported by at least 3 images. Finally, we derive optical flow vectors between pairs of consecutive video frames in terms of the relative position of correspondences in both frames. This process produces sparse optical flow, which will be densified in the next step.

**Flow network.** We train a modified version of the HD$^2$ flow network [49] on the dataset that we generated from KITTI Raw, without any form of pre-training. The main differences with respect to the original implementation are the use of In-Place Activated Batch Norm (iABN) [38], which provides memory savings that enable the second difference, namely the joint training of forward and backward flow. We then run our trained flow network on pairs of consecutive frames in KITTI Raw in order to determine a dense pixel-to-pixel mapping between them. In more detail, for each frame $t$ we compute the backward mapping $\mathbf{f}_t : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, which provides for each pixel of frame $t$ the corresponding pixel in frame $t − 1$.

**Tracklet generation.** In order to represent the tracklets that we have collected up to frame $t$, we use a graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, where vertices are all segments that have been extracted until the $t$th frame, i.e. $\mathcal{V}_t = \bigcup_{s=1}^{t} S_s$, and edges $\mathcal{E}_t \subseteq \mathcal{V}_t^2$ provide the matching segments across consecutive frames. The vertex set $\mathcal{V}_t$ of graph $\mathcal{G}_t$ is given by the union of the vertex set $\mathcal{V}_{t-1}$ of $\mathcal{G}_{t-1}$ and $S_t$, i.e. $\mathcal{V}_t = \mathcal{V}_{t-1} \cup S_t$. The edge set is computed by solving a linear assignment problem between $S_{t-1}$ and $S_t$, where the payoff function is constructed by comparing segments in $S_t$ against segments in $S_{t-1}$ warped to frame $t$ via the mapping $\mathbf{f}_t$. We design the payoff function $\pi(\hat{s}, s)$ between a segment $\hat{s} \in S_{t-1}$ in frame $t − 1$ and a segment $s \in S_t$ in frame $t$ as follows:

$$\pi(\hat{s}, s) = \text{IoU}(\phi_{\hat{s}}, \phi_s \circ \mathbf{f}_t) + \eta(\hat{s}, s),$$

where $\text{IoU}(\cdot, \cdot)$ computes the Intersection-over-Union of the two masks given as input, and $\eta(\hat{s}, s) \in \{−\infty, 0\}$ is a
characteristic function imposing constraints regarding valid mappings. Specifically, \( \eta(\tilde{s}, s) \) returns 0 if and only if the two segments belong to the same category, i.e. \( y_s = y_{\tilde{s}} \), and none of the following conditions on \( s \) hold:

- \( b_1(s) < b_2(s) < \tau_0 \)
- \( b_1(s) < \tau_1 \)
- \( b_1(s)/r(s) < \tau_2 \),

where \( b_1(s) \) and \( b_2(s) \) denote the largest and second-largest areas of intersection with \( s \) obtained by segments warped from frame \( t - 1 \) and having class \( y_s \), and \( r(s) \) denotes the area of \( s \) that is not covered by segments of class \( y_s \) warped from frame \( t - 1 \). We solve the linear assignment problem using the Hungarian algorithm, by maximizing the total payoff under a relaxed assignment constraint, which allows segments to remain unassigned. In other terms, we solve the following optimization problem:

\[
\begin{align*}
\text{maximize} & \quad \sum_{s \in S_t} \sum_{\tilde{s} \in S_{t-1}} \pi(\tilde{s}, s) \alpha(\tilde{s}, s) \\
\text{s.t.} & \quad \sum_{s \in S_{t-1}} \alpha(\tilde{s}, s) \leq 1 \quad \forall \tilde{s} \in S_{t-1} \quad (2) \\
& \quad \sum_{\tilde{s} \in S_t} \alpha(\tilde{s}, s) \leq 1 \quad \forall s \in S_t \quad (3) \\
& \quad \alpha(\tilde{s}, s) \in \{0, 1\} \quad \forall (\tilde{s}, s) \in S_{t-1} \times S_t. \quad (4)
\end{align*}
\]

Finally, the solution \( \alpha \) of the assignment problem can be used to update the set of edges \( \mathcal{T}_t \), i.e. \( \mathcal{T}_t = \mathcal{T}_{t-1} \cup \alpha^{-1}(1) \), where \( \alpha^{-1}(1) \) is the set of matching segments, i.e. the pre-image of 1 under \( \alpha \).

Note that this algorithm cannot track objects across full occlusions, since optical flow is unable to infer the trajectories of invisible objects. When training MOTSNet however, we easily overcome this limitation with a simple heuristic, described in Sec. 4.2, without needing to join the tracklets into full tracks.

4. MOTSNet

Here we describe our multi-object tracking and segmentation approach. Its main component is MOTSNet (see Sec. 4.1), a deep net based on the Mask R-CNN [12] framework. Given an RGB image, MOTSNet outputs a set of instance segments, each augmented with a learned embedding vector which represents the object’s identity in the sequence (see Sec. 4.2). The object tracks are then reconstructed by applying a LAP-based algorithm, described in Sec. 4.3.

4.1. Architecture

MOTSNet’s architecture follows closely the implementation of Mask R-CNN described in [35], but additionally comprises a Tracking Head (TH) that runs in parallel to the Region Segmentation Head (RSH) (see Figure 2). The “backbone” of the network is composed of an FPN [22] component on top of a ResNet-50 body [13], producing multi-scale features at five different resolutions. These features are fed to a Region Proposal Head (RPH), which predicts candidate bounding boxes where object instances could be located in the image. Instance specific features are then extracted from the FPN levels using ROI Align [12] in the areas defined by the candidate boxes, and fed to the RSH and the TH. Finally, for each region, the RSH predicts a vector of class probabilities, a refined bounding box and an instance segmentation mask. Synchronized InPlace-ABN [38] is employed throughout the network after each layer with learnable parameters, except for those producing output predictions. For an in-depth description of these components we refer the reader to [35].

Tracking Head (TH). Instance segments, together with the corresponding ROI Align features from the FPN, are fed to the TH to produce a set of \( N_c \)-dimensional embedding vectors. As a first step, the TH applies “mask-pooling” (described in the section below) to the input spatial features, obtaining 256-dimensional vector features. This operation
is followed by a fully connected layer with 128 channels and synchronized InPlace-ABN. Similarly to masks and bounding boxes in the RSH, embedding vectors are predicted in a class-specific manner by a final fully connected layer with $N_c \times C$ outputs, where $C$ is the number of classes, followed by L2-normalization to constrain the vectors on the unit hyper-sphere. The output vectors are learned such that instances of the same object in a sequence are mapped close to each other in the embedding space, while instances of other objects are mapped far away. This is achieved by minimizing a batch hard triplet loss, described in Sec. 4.2.

**Mask-pooling.** ROI Align features include both foreground and background information, but, in general, only the foreground is actually useful for recognizing an object across frames. In our setting, by doing instance segmentation together with tracking, we have a source of information readily available to discriminate between the object and its background. This can be exploited in a straightforward attention mechanism: pooling under the segmentation mask. Formally, given the pixel indicator function $\phi(s(i, j))$ for a segment $s$ (see Sec. 3.1), and the corresponding input $N$-dimensional ROI-pooled feature map $X_s: \mathbb{R}^2 \rightarrow \mathbb{R}^N$, mask-pooling computes a feature vector $x_s \in \mathbb{R}^N$ as:

$$x_s = \sum_{i} \sum_{j} \phi_s(i, j) X_s(i, j).$$

During training we pool under the ground truth segmentation masks, while at inference time we switch to the masks predicted by the RSH.

### 4.2. Training Losses

MOTSNet is trained by minimizing the following loss function:

$$L = L_{TH} + \lambda (L_{RPH} + L_{RSH}),$$

where $L_{RPH}$ and $L_{RSH}$ represent the Region Proposal Head and Region Segmentation Head losses as defined in [35], and $\lambda$ is a weighting parameter.

$L_{TH}$ is the loss component associated with the Tracking Head, i.e. the following batch hard triplet loss [14]:

$$L_{TH} = \frac{1}{|S_B|} \sum_{s \in S_B} \max \left( \max_{\hat{s} \in \theta_B(s)} \| a_s^{y_s} - a_{\hat{s}}^{y_{\hat{s}}} \|, \min_{\hat{s} \in \theta_B(s)} \| a_s^{y_s} - a_{\hat{s}}^{y_{\hat{s}}} \| + \beta, 0 \right),$$

where $S_B$ is the set of predicted, positive segments in the current batch, $a_s^{y_s}$ is the class-specific embedding vector predicted by the TH for a certain segment $s$ and ground truth class $y_s$, and $\beta$ is a margin parameter. The definition of a “positive” segment follows the same logic as in the RSH (see [35]), i.e. a segment is positive if its bounding box has high IoU with a ground truth segment’s bounding box. The functions $\mu_B(s)$ and $\bar{\mu}_B(s)$ map $s$ to the sets of its “matching” and “non-matching” segments in $S_B$, respectively. These are defined as:

$$\mu_B(s) = \{ \hat{s} \in S_B \mid y_s = y_{\hat{s}} \land M(s, \hat{s}) \}, \quad \bar{\mu}_B(s) = \{ \hat{s} \in S_B \mid y_s = y_{\hat{s}} \land \neg M(s, \hat{s}) \},$$

where $M(s, \hat{s})$ is true iff $s$ and $\hat{s}$ belong to the same tracklet in the sequence. Note that we are restricting the loss to only compare embedding vectors of segments belonging to the same class, effectively letting the network learn a set of class-specific embedding spaces.

In order to overcome the issue due to occlusions mentioned in Sec. 3.2, when training on KITTI Synth we also apply the following heuristic. Ground truth segments are only considered in the tracking loss if they are associated with a tracklet that appears in more than half of the frames in the current batch. This ensures that, when an occlusion causes a track to split in multiple tracklets, these are never mistakenly treated by the network as different objects.

### 4.3. Inference

During inference, we feed frames $t$ through the network, obtaining a set of predicted segments $S_t$, each with its predicted class $y_s$ and embedding vector $a_s^{y_s}$. Our objective is now to match the segments across frames in order to reconstruct a track for each object in the sequence. To do this, we follow the algorithmic framework described in Sec. 3.2, with a couple of modifications. First, we allow matching segments between the current frame and a sliding window of $N_w$ frames in the past, in order to be able to “jump” across occlusions. Second, we do not rely on optical flow anymore, instead measuring the similarity between segments in terms of their distance in the embedding space and temporal offset.

More formally, we redefine the payoff function in Eq. (1) as:

$$\pi(\hat{s}, s) = -\pi^*(\hat{s}, s) + \eta(s, \hat{s}),$$

$$\pi^*(\hat{s}, s) = \| a_s^{y_s} - a_{\hat{s}}^{y_{\hat{s}}} \| + \frac{|t_s - t_{\hat{s}}|}{N_w},$$

where the characteristic function $\eta(s, \hat{s})$ is redefined as:

$$\eta(s, \hat{s}) = \begin{cases} 0 & \text{if } y_s = y_{\hat{s}} \land \pi^*(\hat{s}, s) \leq \tau, \\ -\infty & \text{otherwise}, \end{cases}$$

and $\tau$ is a configurable threshold value. Furthermore, we replace $S_{t-1}$ in Eq. (2) and Eq. (4) with the set:

$$U_t = \{ s \in V_{t-1} \mid t_s \geq t - N_w \land (s, \hat{s}) \notin T_{t-1} \forall \hat{s} \},$$

which contains all terminal (i.e. most recent) segments of the track seen in the past $N_w$ frames. As a final step after processing all frames in the sequence, we discard the tracks with fewer than $N_t$ segments, as, empirically, very short tracks usually arise from spurious detections.
5. Experiments

We provide a broad experimental evaluation assessing i) the quality of our automatically harvested KITTI Synth dataset on the MOTS task by evaluating it against the KITTI MOTS dataset\(^2\), ii) demonstrate the effectiveness of MOTSNet and our proposed mask-pooling layer against strong baselines, iii) demonstrate the generality of our MOTS label generation process by extending the BDD100k tracking dataset with segmentation masks to become a MOTS variant thereof and iv) provide an ablation about the contribution of association terms for the final track extraction. While we can benchmark the full MOTS performance on KITTI MOTS and MOTSChallenge, we are able to infer the tracking performance based on the ground truth box-based tracking annotations available for BDD100k.

5.1. Evaluation Measures

The CLEAR MOT [2] metrics, including Multi-Object Tracking Accuracy (MOTA) and Precision (MOTP), are well established as a standard set of measures for evaluating Multi-Object Tracking systems. These, however, only account for the bounding box of the tracked objects. Voigtlaender et al. [43] describe an extension of MOTA and MOTP that measure segmentation as well as tracking accuracy, proposing the sMOTSA, MOTSA and MOTSP metrics. In particular, MOTSA and MOTSP are direct equivalents of MOTA and MOTP, respectively, where the prediction-to-ground-truth matching process is formulated in terms of mask IoU instead of bounding box IoU. Finally, sMOTSA can be regarded as a “soft” version of MOTSA where the contribution of each true positive segment is weighted by its mask IoU with the corresponding ground truth segment. Please refer to the supplementary material for a formal description of these metrics.

5.2. Experimental Setup

All the results in the following sections are based on the same MOTSNet configuration, with a ResNet-50 backbone and embedding dimensionality fixed to \(N_e = 32\). Batches are formed by sampling a subset of \(N_w\) full-resolution, contiguous frames at a random offset in one of the training sequences, for each GPU. During training we apply scale augmentation in the \([0.8, 1.25]\) range and random flipping. Training by SGD follows the following linear schedule:

\[
lr_t = lr_0(1 - \frac{t}{\text{no steps}}),
\]

where \(lr_0\) is the starting learning rate and \(t\) is the training step (i.e. batch) index.\(^3\) The network weights are initialized from an ImageNet-pretrained model (“I” in the tables), or from a Panoptic Segmentation model trained on the Mapillary Vistas dataset (“M” in the tables), depending on the experiment. Differently from [43], we do not pre-train on MS COCO [21] (“C” in the tables).

The hyper-parameters of the inference algorithm are set as follows: threshold \(\tau = 1\); window size \(N_w\) equal to the per-GPU batch size used during training; minimum track length \(N_t = 5\). Note that, in contrast with [43], we do not fine-tune these parameters, instead keeping them fixed in all experiments, and independent of class. All experiments are run on four V100 GPUs with 32GB of memory.

5.3. KITTI MOTS

The KITTI MOTS dataset was introduced in [43], adding instance segmentation masks for cars and pedestrians to a subset of 21 sequences from KITTI raw [11]. It has a total of 8.008 images (5.027 for training and 2.981 for validation), containing approximately 18.8k/8.1k annotated masks for 431/99 tracks of cars/pedestrians in training, and roughly 8.1k/3.3k masks for 151/68 tracks on the validation split. Approximately 12.3% of all masks were manually annotated and the rest are human-verified and -corrected instance masks. As the size of this dataset is considerably larger and presumably more informative than MOTSChallenge, we focused our ablations on it. Table 1 summarizes the analyses and experiments we ran on our newly generated KITTI Synth dataset, by evaluating on the official KITTI MOTS validation set. We discuss three types of ablations in what follows.

KITTI Synth data quality analysis. The two topmost rows of Tab. 1 show the results when generating the validation set results solely with our proposed dataset generation pipeline, described in Section 3, i.e. no learned tracking component is involved. Our synthetically extracted validation data considerably outperforms the sMOTSA scores on pedestrians and almost obtains the performance on cars for CAMOT [31], as reported in [43]. When training the flow network on the training data generated from SIM as described in Section 3, we match the performance obtained by the best-performing HD\(^3\) [49] model for KITTI\(^4\). While this validates our tracklet generation process and the way we harvest optical flow training data, we investigate further the effects of learning from imperfect data (see Sec. 3.2).

MOTSNet tracking head ablations. The center block in Tab 1 compares different configurations for the Tracking Head described in 4. The first variant (AVEBOX+TH) replaces the mask-pooling operation in the Tracking Head with average pooling on the whole box. The second and third variants (AVEMSK-TH, AVEBOX-TH) completely remove the Tracking Head and its associated loss, instead directly computing the embeddings by pooling features under the detected mask or box, respectively. All variants perform reasonably well and improve over [30, 31] on the primary

\(^2\)MOTSChallenge results are available in the supplementary material.

\(^3\)Please refer to the supplementary material for a full specification of the training parameters.

\(^4\)Pre-trained models available at https://github.com/ucbdrive/hd3.
### Table 1: Results on the KITTI MOTS validation set when training on KITTI Synth. First section: ablation results. Second section: main results and comparison with state of the art. Note that all MOTSNet variants are trained exclusively on machine generated annotations, while TrackR-CNN is trained on human annotated ground truth.

<table>
<thead>
<tr>
<th>Method</th>
<th>Pre-training</th>
<th>sMOTSA Car</th>
<th>sMOTSA Ped</th>
<th>MOTSA Car</th>
<th>MOTSA Ped</th>
<th>MOTSP Car</th>
<th>MOTSP Ped</th>
<th>mAP Box</th>
<th>mAP Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>KITTI Synth (val) + HD[^49] model zoo</td>
<td>inference only</td>
<td>65.0</td>
<td>45.7</td>
<td>77.3</td>
<td>66.3</td>
<td><strong>87.6</strong></td>
<td>76.6</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>KITTI Synth (val) + HD[^3], KITTI-SfM</td>
<td>inference only</td>
<td>65.5</td>
<td>45.4</td>
<td>77.4</td>
<td>66.0</td>
<td><strong>87.6</strong></td>
<td>76.6</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>MOTSNet with:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AVEBOX+TH</td>
<td>I</td>
<td>73.7</td>
<td>44.4</td>
<td>85.8</td>
<td>62.8</td>
<td>86.7</td>
<td>76.7</td>
<td>57.4</td>
<td>50.9</td>
</tr>
<tr>
<td>AVEMSK+TH</td>
<td>I</td>
<td>76.4</td>
<td>44.0</td>
<td>88.5</td>
<td>60.3</td>
<td>86.8</td>
<td>76.6</td>
<td>57.8</td>
<td>51.3</td>
</tr>
<tr>
<td>AVEBOX-TH</td>
<td>I</td>
<td>75.4</td>
<td>44.5</td>
<td>87.3</td>
<td>60.8</td>
<td>86.9</td>
<td>76.7</td>
<td>57.5</td>
<td>51.0</td>
</tr>
<tr>
<td>KITTI MOTS train sequences only</td>
<td>I</td>
<td>72.6</td>
<td>45.1</td>
<td>84.9</td>
<td>62.9</td>
<td>86.1</td>
<td>75.6</td>
<td>52.5</td>
<td>47.6</td>
</tr>
<tr>
<td>MOTSNet</td>
<td>I</td>
<td>77.6</td>
<td>49.1</td>
<td>89.4</td>
<td>65.6</td>
<td>87.1</td>
<td>76.4</td>
<td>58.1</td>
<td>51.8</td>
</tr>
<tr>
<td>MOTSNet</td>
<td>I, M</td>
<td><strong>77.8</strong></td>
<td><strong>54.5</strong></td>
<td><strong>89.7</strong></td>
<td><strong>70.9</strong></td>
<td><strong>87.1</strong></td>
<td><strong>78.2</strong></td>
<td><strong>60.8</strong></td>
<td><strong>54.1</strong></td>
</tr>
</tbody>
</table>

sMOTSA metric. Notably, AVEMSK-TH, *i.e.* the variant using our mask-pooling layer and no tracking head, is about on par with TrackR-CNN on cars despite being pre-trained only on ImageNet, using a smaller, ResNet-50 backbone and not exploiting any tracking supervision. The last variant in this block shows the performance obtained when MOTSNet is trained only on images also in the KITTI MOTS training set, and with our full model. Interestingly, the scores only drop to an extent where the gap to [43] might be attributed to their additional pre-training on MS COCO and Mapillary Vistas and a larger backbone (cf. Tab. 2). Finally, comparing the latter directly to our MOTSNet results from Tab. 2 (ImageNet pre-trained only), where we directly trained on KITTI MOTS training data, our KITTI Synth-based results improve substantially on cars, again confirming the quality of our automatically extracted dataset.

The bottommost part of this table shows the performance when training on the full KITTI Synth dataset and evaluating on KITTI MOTS validation, using different pre-training settings. The first is pre-trained on ImageNet and the second on both, ImageNet and Mapillary Vistas. While there is only little gap between them on the car class, the performance on pedestrians rises by over 5% when pre-training on Vistas. The most encouraging finding is that using our solely automatically extracted KITTI Synth dataset we obtain significantly improved scores (+1.6% on cars, +7.4% on pedestrians) in comparison to the previous state-of-the-art method from [43], trained on manually curated data.

**MOTSNet on KITTI MOTS.** In Tab. 2 we directly compare against previously top-performing works including TrackR-CNN [43] and references from therein, *e.g.* CAMOT [31], CIWT [30], and BeyondPixels [40]. It is worth noting that the last three approaches were reported in [43] and partially built on. We provide all relevant MOTS measures together with separate average AP numbers to estimate the quality of bounding box detections or instance segmentation approaches, respectively. For our MOTSNet we again show results under different pre-training settings (ImageNet, Mapillary Vistas and KITTI Synth). Different pre-training affects the classes cars and pedestrians in a different way, *i.e.* the ImageNet and Vistas pre-trained model performs better on pedestrians while the KITTI Synth pre-trained one is doing better on cars. This is most likely due to the imbalanced distribution of samples in KITTI Synth while our variant with pre-training on both, Vistas and KITTI Synth yields the overall best results. We obtain absolute improvements of 1.9%/7.5% for cars/pedestrians over the previously best work in [43] despite using a smaller backbone and no pre-training on MS COCO. Finally, also the recognition metrics for both, detection (box mAP) and instance segmentation (mask mAP) significantly benefit from pre-training on our KITTI Synth dataset, and in conjunction with Vistas rise by 7.2% and 6.4% over the ImageNet pre-trained variant, respectively.

**5.4. MOTSNet on BDD100k.**

Our MOTS training data generation pipeline can be directly ported to other datasets like the recently released BDD100k tracking dataset [50]. It comprises 61 sequences, for a total of ~11k frames and comes with bounding box based tracking annotations. We focus on cars and pedestrians, for compatibility with KITTI Synth and KITTI MOTS. We split the available data into training and validation sets of, respectively, 50 and 11 sequences, and generate segmentation masks for each annotated bounding box, again by using [35] augmented with a ResNeXt-101-32×8d backbone. During data generation, the instance segmentation pipeline detected many object instances missing in the annotations, which is why we decided to provide *ignore* box annotations for detections with very high confidence but missing in the ground truth. Since there are only bounding-box based tracking annotations available, we present MOTS rather than MOTS tracking scores. The results are listed in Tab. 3, comparing pooling strategies AVEBOX+LOSS and
<table>
<thead>
<tr>
<th>Method</th>
<th>Pre-training</th>
<th>sMOTSA</th>
<th>MOTSA</th>
<th>MOTSP</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Car</td>
<td>Ped</td>
<td>Car</td>
<td>Ped</td>
</tr>
<tr>
<td>TrackR-CNN [43]</td>
<td>I, C, M</td>
<td>76.2</td>
<td>47.1</td>
<td>87.8</td>
<td>65.5</td>
</tr>
<tr>
<td>CAMOT [31]</td>
<td>I, C, M</td>
<td>67.4</td>
<td>39.5</td>
<td>78.6</td>
<td>57.6</td>
</tr>
<tr>
<td>CIWT [30]</td>
<td>I, C, M</td>
<td>68.1</td>
<td>42.9</td>
<td>79.4</td>
<td>61.0</td>
</tr>
<tr>
<td>BeyondPixels [40]</td>
<td>I, C, M</td>
<td>76.9</td>
<td>–</td>
<td>89.7</td>
<td>–</td>
</tr>
</tbody>
</table>

| MOTSNet            | I             | 69.0   | 45.4  | 78.7  | 61.8| 88.0  | 76.5 | 55.2 | 49.3 |
|                    | I, M          | 74.9   | 53.1  | 83.9  | 67.8| 89.4  | 79.4 | 60.8 | 54.9 |
|                    | I, KS         | 76.4   | 48.1  | 86.2  | 64.3| 88.7  | 77.2 | 59.7 | 53.3 |

Table 2: Results on the KITTI MOTS validation set when training on the KITTI MOTS training set. First section: state of the art results using masks and detections from [43]. Second section: our results under different pre-training settings.

<table>
<thead>
<tr>
<th>MOTSNet variant</th>
<th>Pre-training</th>
<th>MOTA</th>
<th>MOTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>AveBox+LOSS</td>
<td>I</td>
<td>53.8</td>
<td>83.1</td>
</tr>
<tr>
<td></td>
<td>I, M</td>
<td>56.9</td>
<td>83.9</td>
</tr>
<tr>
<td>AveMSk+LOSS</td>
<td>I</td>
<td>53.9</td>
<td>83.1</td>
</tr>
<tr>
<td></td>
<td>I, M, KS</td>
<td>58.2</td>
<td>84.0</td>
</tr>
</tbody>
</table>

Table 3: MOT ablation results on the BDD100K dataset.

<table>
<thead>
<tr>
<th>sIoU</th>
<th>Embedding</th>
<th>Time</th>
<th>sMOTSA Car</th>
<th>sMOTSA Ped</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>76.7</td>
<td>50.8</td>
</tr>
<tr>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>78.2</td>
<td>54.4</td>
</tr>
<tr>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>77.0</td>
<td>51.8</td>
</tr>
<tr>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>78.1</td>
<td>54.6</td>
</tr>
</tbody>
</table>

Table 4: Ablation results on the KITTI MOTS dataset, using our best performing model from Tab. 2 when switching between different cues in the payoff function for inference.

AveMSk+LOSS and for different pre-trainings. While the improvement is moderate compared to the ImageNet-only pre-trained backbone, the Vistas pre-trained model benefits noticeably from mask-pooling.

5.5. Ablations on Track Extraction

Here we analyze the importance of different cues in the payoff function used during inference (see Section 4.3): distance in the embedding space (Embedding), distance in time (Time) and signed intersection over union [42] (sIoU) between bounding boxes. As a basis, we take the best performing MOTSNet model from our experiments on KITTI MOTS, listed at the bottom of Table 2. This result was obtained by combining Embedding and Time, as in Eq. (6). As can be seen from differently configured results in Tab. 4, the embedding itself already serves as a good cue, and can be slightly improved on pedestrians when combined with information about proximity in time (with 0.1 drop on cars), while outperforming sIoU. Figure 3 shows a visualization of the embedding vectors learned by this model.

Figure 3: t-SNE visualization of the embedding vectors computed by the Tracking Head for sequence “0014” of KITTI MOTS. Points corresponding to detections of the same object have the same color.

6. Conclusions

In this work we addressed and provided two major contributions for the novel task of multi-object tracking and segmentation (MOTS). First, we introduced an automated pipeline for extracting high-quality training data from generic street-level videos to overcome the lack of MOTS training data, without time- and cost-intensive, manual annotation efforts. Data is generated by solving the linear assignment on a causal tracking graph, where instance segmentations per frame define the nodes, and optical-flow based compatibilities represent edges as connections over time. Our second major contribution is a deep-learning based MOTSNet architecture to be trained on MOTS data, exploiting a novel mask-pooling layer that guides the association process for detections based on instance segmentation masks. We provide exhaustive ablations for both, our novel training data generation process and our proposed MOTSNet. We improve over the previously best work [43] for the KITTI MOTS dataset (1.9%/7.5% for cars/pedestrians) and on MOTSChallenge (4.1% for pedestrians), respectively.

Acknowledgements. Idoia and Joan acknowledge financial support by the Spanish grant TIN2017-88709-R (MINECO/AEI/FEDER, UE).
References


