DG-Labeler and DGL-MOTS Dataset: Boost the Autonomous Driving Perception

Yiming Cui*
University of Florida
Gainesville, FL 32611
cuiyiming@ufl.edu

Zhiwen Cao*
Purdue University
West Lafayette, IN 47907
cao270@purdue.edu

Yixin Xie
The University of Texas at El Paso
El Paso, TX 79968
yxie4@miners.utep.edu

Xingyu Jiang
Purdue University
West Lafayette, IN 47907
jiang718@purdue.edu

Feng Tao
The University of Texas at San Antonio
San Antonio, TX 78249
feng.tao@my.utsa.edu

Yingjie Victor Chen
Purdue University
West Lafayette, IN 47907
victorchen@purdue.edu

Lin Li
The University of Texas at El Paso
El Paso, TX 79968
lli5@utep.edu

Dongfang Liu†
Rochester Institute of Technology
Rochester, NY 14623
dongfang.liu@rit.edu

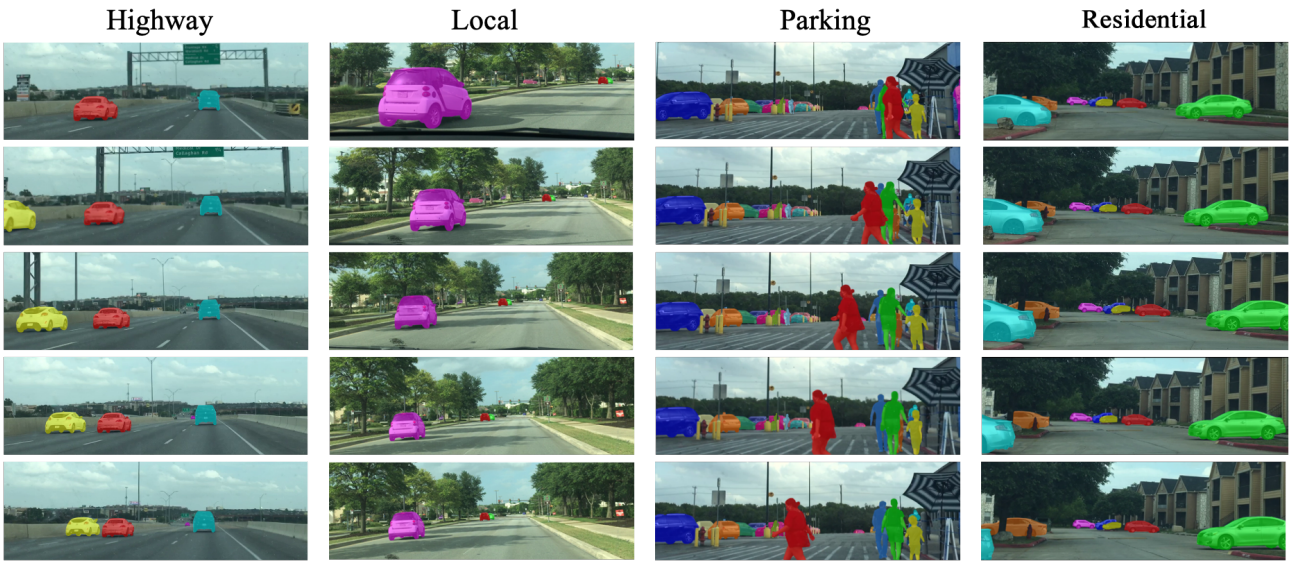

Figure 1. A showcase of the DGL-MOTS dataset. We collect data based on different driving scenarios and organize training data based on different settings in terms of the highway, local, parking, and residential.

Abstract

Multi-object tracking and segmentation (MOTS) is a critical task for autonomous driving applications. The existing MOTS studies face two critical challenges: 1) the published datasets inadequately capture the real-world complexity for network training to address various driving settings; 2) the working pipeline annotation tool is under-studied in the literature to improve the quality of MOTS learning examples. In this work, we introduce the DG-Labeler and DGL-MOTS

*are equal contributions.
†is corresponding author.

dataset to facilitate the training data annotation for the MOTS task and accordingly improve network training accuracy and efficiency. DG-Labeler uses the novel Depth-Granularity Module to depict the instance spatial relations and produce fine-grained instance masks. Annotated by DG-Labeler, our DGL-MOTS dataset exceeds the prior effort (i.e., KITTI MOTS and BDD100K) in data diversity, annotation quality, and temporal representations. Results on extensive cross-dataset evaluations indicate significant performance improvements for several state-of-the-art methods trained on our DGL-MOTS dataset. We believe our DGL-MOTS Dataset and DG-Labeler hold the valuable potential to boost the visual perception of future transportation. Our dataset and code are available here[1].

1. Introduction

A major contributing factor behind recent success in deep learning is the availability of large-scale annotated datasets [37]. Despite the existing performance gap to humans, deep-learning-based computer vision methods have become essential advancement of real-world systems. A particularly challenging and emerging application is autonomous driving, which requires system performance with extreme reliability [25, 24, 6, 26]. However, leveraging the power of deep learning for autonomous driving is nontrivial, due to the lack of datasets.

Consequently, significant research efforts have been invested into autonomous driving datasets such as KITTI [9], Cityscapes [5], and BDD100K [47] datasets, which serve as the driving force to the development of visual technologies for understanding complex traffic scenes and driving scenarios. As the computer vision community has made impressive advances in increasingly difficult tasks ($i.e.$, object detection, instance segmentation, and multi-object tracking) in recent years, a new task named multi-object tracking and segmentation (MOTS) is proposed in order to consider detection, segmentation and tracking together as interconnected problems [39]. Consequently, the KITTI MOTS dataset [39] is introduced to assess the proposed visual task.

Although the existing MOTS datasets [39, 47] fills the gap of data shortage for MOTS task, they are limited in three significant drawbacks in the training data: *1) including no challenging cases (i.e., motion blur or defocus) to address a general driving setting; 2) focusing on local roads of inner cities and thus lack diversity.* These limits may cause problems in training as the data does not fully capture the real-world traffic complexity. In addition, *the annotation for MOTS data is highly labor-intensive as it requires the pixel-level mask as well as the temporal tracking label across frames.* In order to produce the MOTS data, [39] uses a refinement network to generate an initial segmentation mask followed by human corrections. Afterward, tracking labels are created by delineating the temporal coherence based on instance masks across video frames [39].

To this end, we, therefore, propose the DGL-MOTS dataset to maximize synergies tailored for the MTOS task in autonomous driving. In order to improve the annotation quality and reduce the annotation cost, we devise DG-Labeler to produce fine-grained instance masks. Concretely, our work delivers the following contributions:

- We create a **D**epth-**G**ranularity **L**abeled **MOTS**, thus the name of our dataset, **DGL-MOTS** (Figure 2). Compared to the KITTI MOTS and BDD100K, DGL-MOTS significantly exceeds the previous effort in terms of annotation quality, data diversity, and temporal representation, which boosts the training accuracy and efficiency.
- We perform cross-dataset evaluations. Extensive experiment results indicate the benefits of our datasets. Networks trained on our dataset outperform their counterparts (with the same architecture) trained on KITTI MOTS [39] and BDD100K [47] on the same test set. Also, improvement for networks trained on our dataset is reached with less training schedule (Table 2).
- We propose an end-to-end annotator named DG-Labeler (Figure 2), whose architecture includes a novel depth-granularity module to model the spatial relation of instances and assist to produce fine-grained instance mask. With limited correction iterations, DG-Labeler can generate high-quality MOTS annotation.
- DG-Labeler leverages the depth information to depict the instance spatial relation and retain finer details at the instance boundary (Figure 3). On both KITTI MOTS and DGL-MOTS datasets, DG-Labeler outperforms TrackR-CNN [39] in accuracy by a significant margin. For its simplicity, we hope DG-Labeler can also serve as a new strong baseline for the MOTS task.

2. Related Work

This section summarizes the related datasets for autonomous driving, multi-object tracking and segmentation as well as annotation methods for dataset creation.

MOTS dataset. The multi-object tracking (MOT) is a critical task for autonomous driving, as it needs to perform object detection as well as object tracking in a video. A large array of datasets have been created focusing on driving scenarios, for example, KITTI tracking [10], MOTChallenge [31], UA-DETRAC [42], PathTrack [30], and PoseTrack [1]. None of these datasets provide segmentation masks for the annotated objects and thus do not depict pixel-level representations and complex interactions like MOTS

[1] https://goodproj13.github.io/DGL-MOTS/

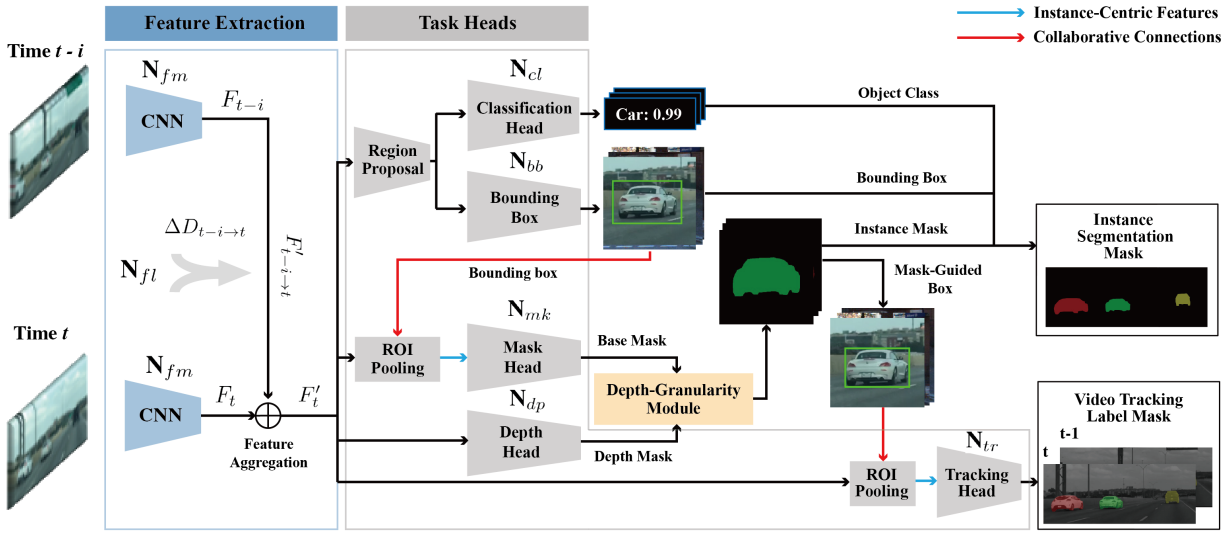

Figure 2. Illustration of DG-Labeler architecture. We craft our DG-Labeler on TrackR-CNN. In the feature extraction phase, we replace the original heavy 3D convolution with a more efficient flow network to increase the feature representation. In the task phase, we devise the collaborative connections to propagate information across each task so the upper-level task heads (a.k.a. the mask and tracking head) can perform accurate and efficient predictions on instance-centric features. Moreover, we propose a depth-granularity module, which greatly improves the segmentation behavior of our DG-Labeler.

data. More progressive datasets come from Cityscapes [5], ApolloScape [15], BDD100K [47], and KITTI MOTS dataset [39] which provide instance segmentation data for autonomous driving. However, Cityscapes only provides instance annotations for a small subset (i.e., 5,000 images) while ApolloScape offers no temporal object descriptions over time. Thus, the two datasets cannot be utilized for joint training of MOTS algorithms. In contrast, KITTI MOTS [39] is the first public dataset which fills the gap of data shortage for the MOTS task but it only includes a few thousand learning data for training; to date, BDD100K has the largest data scale from intensive sequential frames which are redundant for training. Compared to the aforementioned two datasets, our DGL-MOTS dataset includes more diverse data and fine-grained annotations.

Multi-object tracking and segmentation. The majority of MOTS methods [4, 41, 48, 18, 20, 46, 32] intuitively extend from Mask R-CNN. Although the extension paradigm is simple, it encounters several performance bottlenecks: 1) feature sharing across each task is insufficient for joint optimization; 2) the mask head struggles to produce fine-grained instance boundaries; 3) the RoI representation (proposal-based features) is redundant which impact the inference speed with the increasing number of proposals. Compared to the existing methods, our method explicitly models spatial relation of instance, which helps us achieve high granularity for instance masks. We circumvent RoI operations for high-level tasks (a.k.a tracking and segmentation) by using collaborative connections, which link each task head interdependently for joint optimization and boost the computational tractability.

Data annotator. MOTS data requires the instance mask as well as the temporal tracking label across frames. So far, many attempts for semi-automated annotation [2, 12, 44, 8, 16, 43] have been made to reduce the annotation overhead. Aside from heavy engagement of human correction efforts [40], these methods generally have arduous implementations for the annotator and require multiple steps to achieve a desirable result. Moreover, their annotators cannot operate on a tracking level and only create instance masks in a single image. To produce the MOTS data, [39] leverages a refinement network to generate an initial segmentation mask followed by human corrections. The tracking labels are then created by delineating the temporal coherence across video frames [39]. To our best knowledge, [39, 34] are two methods available for MOTS annotation, both built on Mask R-CNN by simply adding a tracking branch [13]. Compared to [39, 34], our DG-Labeler explicitly models spatial relation to achieving fine-grained instance boundary. We also devise a collaborative connection, which uses the detection results to guide the high-level tasks (segmentation and tracking) to accurately fire on the task-relevant pixels. With limited human corrections, our annotation protocol can produce MOTS labels with appealing quality.

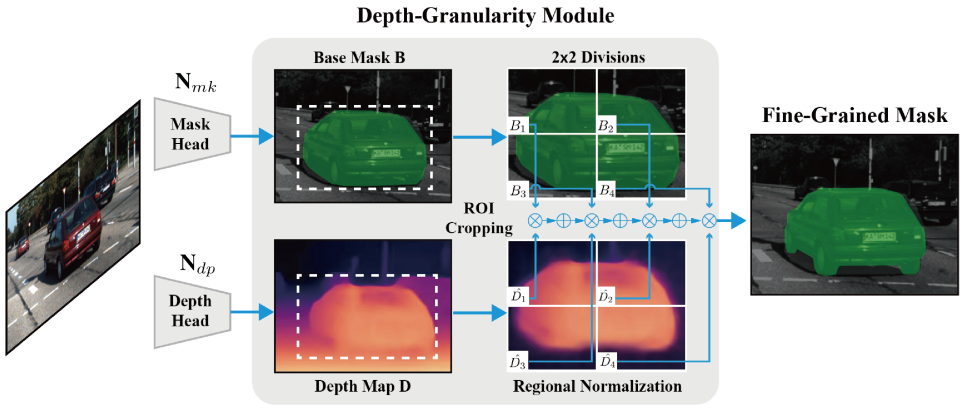

Figure 3. In the depth-granularity module, base mask and depth map are first divided into 2×2 sub-regions. Then, each corresponding sub-region from the base mask and depth map are organically blended to produce the final fine-grained mask.

3. DG-Labeler

3.1. Overall Architecture

Built on the TrackR-CNN [39], our DG-Labeler includes architectures of feature extraction, task heads, and depth-granularity module, which collaboratively perform detection, segmentation, and tracking. Our overall architecture is shown in Figure 2.

Feature extraction uses the ResNet [14] backbone $\mathbf{N}_{fm}$ to compute per frame feature maps F and leverage a flow network $\mathbf{N}_{fl}$ to model temporal features over time on video. Feature maps from the previous moment F_{t-i} are warped into the current time t based on the flow field $\Delta D_{t-i \to t}$ to obtain $F'_{t-i \to t}$. Afterwards, F_t and $F'_{t-i \to t}$ are aggregated for F'_t to increase the feature representation of the current frame. In default, the temporal range is three, namely, adjacent frames are used for the feature aggregation.

Task heads are consisted of four task heads (a.k.a. the classification head $\mathbf{N}_{cl}$, the bounding box head $\mathbf{N}_{bb}$, the mask head $\mathbf{N}_{mk}$, and the tracking head $\mathbf{N}_{tr}$). The aforementioned task heads follow the implementation in TrackR-CNN. Besides, we craft a new depth head $\mathbf{N}_{dp}$ [11] into our network to model the spatial relation of each detected object on the video frame. Since our feature extraction is built on ResNet [14], we replace the U-Net architecture in [11] with feature pyramid networks (FPNs) [22] to predict depth maps. Other implementations are the same in [11].

Depth-granularity module is the key component of our DG-Labeler. The next section will detail this module.

3.2. Depth-Granularity Module

Inspired by [27], we blend the base mask B with the corresponding depth map D to generate the final fine-grained mask in the depth-granularity module (Figure 3). In our implementation, both B and D have the same shape of $H \times W \times 1$. We crop out each region of interest (RoI) on the base mask B and the depth map D based on the detected bounding boxes and divide each RoI into $k \times k$ regions of the same size. The division is arbitrary but we find that $k = 2$ has the best speed-accuracy tradeoff. Afterward, each sub-region of the depth map is normalized by:

$$\hat{D}_i = \frac{D_i - \min(D_i)}{\max(D_i) - \min(D_i)}, \quad \forall i \in \{1, ..., k\}, \tag{1}$$

where D_i and $\hat{D}_i$ represent one sub-region of the depth map and its normalized depth map respectively. $\hat{D}_i$ renders the spatial relation (foreground and background) and boundary details of the target instance (Figure 3).

Finally, we apply element-wise productions between the base mask and the normalized depth map from each corresponding sub-region, and sum along the $k \times k$ regions to obtain the final mask M_j of the j^{th} instance on the frame:

$$M_j = \sum_{i=1}^{k \times k} \sigma(B_i \times \hat{D}_i), \tag{2}$$

where B_i and $\hat{D}_i$ are one sub-region of the base mask and the normalized depth map respectively, and σ is `sigmoid` activation. In our implementation, our base mask B_i uses floating point and the depth map $\hat{D}_i$ encodes the relative spatial relation, not the absolute depth values. With the spatial relation modeling, our final mask is more fine-grained.

3.3. Collaborative Connections

Unlike TrackR-CNN and its variants [39, 29, 21] whose task heads operate independently and ignore the intrinsic correlations among each task, we devise collaborative connections (the red lines in Figure 2) across detection, segmentation, and tracking heads to facilitate the information prorogation across tasks. Compared to TrackR-CNN and its variants, this implementation offers

us two improvements for the network behaviors: 1). our segmentation and tracking head fire on instance-centric features (the blue lines in Figure 2) governed by the bounding boxes and the mask-guided boxes respectively, thus can perform more accurate predictions; 2) we improve the runtime performance by avoiding encoding redundant features based on proposals produced by RPN and thus reducing computational cost per instance.

3.4. Training Objective

GIoU learning [36] is used in our training. Since our method follows the top-down paradigm, we argue that the improved bounding box regression can benefit the instance segmentation and tracking task. Bear this in mind, we leverage the `GIoU` loss in [27] to organize our learning. Particularly, we propose a modified `GIoU` loss using a logarithmic function to increase the bounding box losses in order to facilitate hard sample learning ($i.e.$, small GIoU):

$$\mathcal{L}_{box} = -\ln \frac{1 + GIoU}{2} \tag{3}$$

Consequently, our overall loss can be defined as:

$$\mathcal{L}_{all} = \mathcal{L}_{box} + \mathcal{L}_{cls} + \mathcal{L}_{mask} + \mathcal{L}_{track} + \mathcal{L}_{depth} \tag{4}$$

where $\mathcal{L}_{cls}, \mathcal{L}_{mask}$, and $\mathcal{L}_{track}$ are from [39], $\mathcal{L}_{box}$ is the modified `GIoU` loss from Eq. 3, and $\mathcal{L}_{depth}$ is the average of per-pixel smoothness and masked photo- metric loss in [11]. Our architecture is trained in an end-to-end fashion.

4. DGL-MOTS Dataset

4.1. Data Acquisition

Our data acquisition is carefully designed to capture the high variability of driving scenarios, such as highway, local, residential, and parking. Our raw data is acquired from a moving vehicle with a span of two months, covering different lighting conditions in four different states in the USA. Images are recorded with a GoPro HERO8 at a frame rate of 17 `Hz`, behind the windshield of the vehicle. We deliberately skip post-processing ($i.e.$, rectification or calibration) and keep data with motion blur and defocus to increase data diversity. We argue that data with a low-degree motion blur and defocus can better reflect the driving scenarios. However, severely compromised video frames are excluded from annotations. 40 video sequences are manually selected for dense annotations, aiming for a high diversity of foreground objects (vehicles and pedestrians) and overall scene layout. Our annotation is elaborated in the next section.

4.2. Annotation Protocol

To keep our annotation effort manageable, we use an iterative semi-automated annotation protocol based on our DG-Labeler. At the first iteration, we use the pre-trained DG-Labeler to automatically perform annotations for our data, followed by a manual correction step. Per iteration, we fine-tune our DG-Labeler using the annotated data after manual corrections. We iterate the aforementioned process until pixel-level accuracy for all instances has been reached.

To initialize DG-Labeler, we use ResNet-101 [14] pre-trained on COCO [23] and Mapillary [33] datasets as our feature extraction backbone; FlowNet pre-trained on the Flying Chair dataset [7] is used to predict flow filed; and the depth network [11] pre-trained on the KITTI depth dataset [38] is used to predict depth map. Note training the depth network [11] uses a self-supervised manner and only needs video sequences (3 consecutive frames without ground truth) to train. At the initial training, the weights of ResNet-101 and FlowNet are fixed, and the other weights related to different task heads are updated by learning on KITTI MOTS and BDD100K. We train the initial model for 40 epochs with a learning rate of 5×10^{-7} with Adam [19] optimizer and mini-batch size of 8. After each correction, the refined annotations are used to fine-tune our DG-Labeler.

Eventually, we use 6 iterations to finalize the annotation process. *We perform further processing on the annotated data and select learning examples for training and testing in every 5 frames. Following this design, our dataset generally has longer temporal representations and descriptions.* Instead of splitting our annotated data randomly, we want to ensure that the training, validation, and test sets include the data representation for different driving scenarios, such as highway, local, residential, and parking areas (Figure 1).

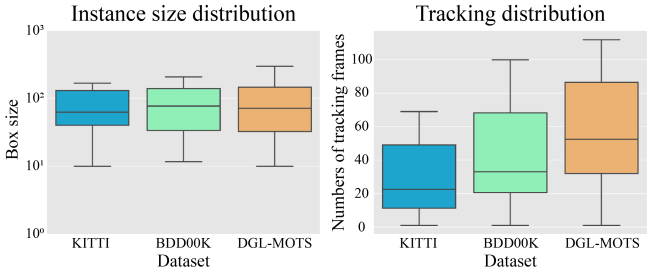

Figure 4. Distribution of the instance size (left) based on the bounding box size and track length (right) based on the duration of instances that appeared on video. Our dataset is more diverse in object scale and tracking length than counterparts.

5. Experiments

5.1. Implementation Details

In order to evaluate the proposed dataset and annotator, we perform cross-dataset evaluations and compare against three recent state-of-the-art MOTS methods[2] (a.k.a. Point-

[2] All the compared methods use ResNet101 [14].

Datasets	Video clip	Total frames	Identities	Instances	Ins./Fr.
KITTI MOTS	21	8K	749	38K	4.78
BDD100K MOTS	70	14K	6.3K	129K	9.20
Ours	40	12K	1.6K	68K	6.23

Table 1. Annotation statistics. Our dataset outperforms the KITTI MOTS in annotation volume and density. BDD100K offers the largest training data but selected sequentially from video frames, which include redundant temporal information.

Method	Ep.	Training Dataset	Testing Dataset	Cars				Pedestrians			
				HOTA ↑	sMOTSA↑	MOTSA↑	IDS↓	HOTA ↑	sMOTSA↑	MOTSA↑	IDS↓
PointTrack++	40	KITTI	KITTI	67.28	82.82	92.61	**36**	56.67	68.13	83.67	36
PointTrack++	40	DGL-MOTS	KITTI	**68.40**$_{\uparrow 1.12}$	**84.33**$_{\uparrow 1.51}$	**93.68**$_{\uparrow 1.07}$	**32**$_{\downarrow 4}$	**57.87**$_{\uparrow 1.2}$	**69.30**$_{\uparrow 1.17}$	**84.51**$_{\uparrow 0.84}$	**33**$_{\downarrow 3}$
PointTrack++	20	DGL-MOTS	KITTI	**67.42**$_{\uparrow 0.14}$	**82.99**$_{\uparrow 0.6}$	**92.67**$_{\uparrow 0.6}$	**36**$_{\downarrow 0}$	**57.89**$_{\uparrow 1.22}$	**68.20**$_{\uparrow 0.07}$	**83.76**$_{\uparrow 0.09}$	**35**$_{\downarrow 1}$
TrackRCNN	40	KITTI	KITTI	56.92	77.20	87.92	92	42.08	47.43	67.12	78
TrackRCNN	40	DGL-MOTS	KITTI	58.02$_{\uparrow 1.1}$	78.80$_{\uparrow 1.60}$	88.90$_{\uparrow 1.00}$	80$_{\downarrow 12}$	43.11$_{\uparrow 1.03}$	48.61$_{\uparrow 1.18}$	68.33$_{\uparrow 1.21}$	62$_{\downarrow 16}$
TrackRCNN	20	DGL-MOTS	KITTI	57.12$_{\uparrow 0.20}$	77.31$_{\uparrow 0.11}$	88.15$_{\uparrow 0.23}$	90$_{\downarrow 2}$	42.21$_{\uparrow 0.13}$	47.50$_{\uparrow 0.07}$	68.46$_{\uparrow 1.34}$	76$_{\downarrow 2}$
STEm-Seg	40	KITTI	KITTI	56.36	76.30	86.63	76	43.10	51.02	66.60	74
STEm-Seg	40	DGL-MOTS	KITTI	57.50$_{\uparrow 1.14}$	77.35$_{\uparrow 1.05}$	87.92$_{\uparrow 1.29}$	56$_{\downarrow 20}$	45.10$_{\uparrow 2.00}$	52.70$_{\uparrow 1.68}$	68.00$_{\uparrow 1.4}$	60$_{\downarrow 14}$
STEm-Seg	20	DGL-MOTS	KITTI	56.70$_{\uparrow 0.34}$	76.36$_{\uparrow 0.06}$	86.70$_{\uparrow 0.07}$	66$_{\downarrow 10}$	43.45$_{\uparrow 0.35}$	51.42$_{\uparrow 0.4}$	66.99$_{\uparrow 0.39}$	70$_{\downarrow 4}$
PointTrack++	40	BDD100K	BDD100K	**68.33**	**84.60**	93.20	**49**	55.42	**64.56**	**80.29**	45
PointTrack++	40	DGL-MOTS	BDD100K	**69.28**$_{\uparrow 0.95}$	**85.59**$_{\uparrow 0.99}$	**94.32**$_{\uparrow 1.12}$	**38**$_{\downarrow 11}$	**56.89**$_{\uparrow 1.47}$	**65.28**$_{\uparrow 0.72}$	**81.05**$_{\uparrow 0.76}$	**34**$_{\downarrow 11}$
PointTrack++	20	DGL-MOTS	BDD100K	68.26$_{\downarrow 0.07}$	84.43$_{\downarrow 0.17}$	**93.27**$_{\uparrow 0.07}$	52$_{\uparrow 3}$	**55.37**$_{\downarrow 0.05}$	64.23$_{\downarrow 0.33}$	80.26$_{\downarrow 0.03}$	50$_{\uparrow 5}$
TrackRCNN	40	BDD100K	BDD100K	57.91	78.10	88.62	85	46.37	55.93	70.18	88
TrackRCNN	40	DGL-MOTS	BDD100K	59.22$_{\uparrow 1.31}$	79.82$_{\uparrow 1.72}$	89.90$_{\uparrow 1.08}$	68$_{\downarrow 17}$	47.49$_{\uparrow 1.42}$	56.61$_{\uparrow 0.68}$	71.80$_{\uparrow 1.62}$	78$_{\downarrow 10}$
TrackRCNN	20	DGL-MOTS	BDD100K	58.09$_{\uparrow 0.18}$	78.20$_{\uparrow 0.10}$	88.69$_{\uparrow 0.07}$	80$_{\downarrow 5}$	46.52$_{\uparrow 0.15}$	56.07$_{\uparrow 0.14}$	70.32$_{\uparrow 0.14}$	84$_{\downarrow 4}$
STEm-Seg	40	BDD100K	BDD100K	57.39	77.24	87.65	66	47.65	56.30	71.03	48
STEm-Seg	40	DGL-MOTS	BDD100K	58.62$_{\uparrow 1.23}$	78.50$_{\uparrow 1.20}$	88.96$_{\uparrow 1.30}$	56$_{\downarrow 10}$	49.00$_{\uparrow 1.35}$	57.72$_{\uparrow 1.42}$	72.20$_{\uparrow 1.17}$	**38**$_{\downarrow 10}$
STEm-Seg	20	DGL-MOTS	BDD100K	57.70$_{\uparrow 0.31}$	77.78$_{\uparrow 0.54}$	88.04$_{\uparrow 0.39}$	64$_{\downarrow 2}$	47.95$_{\uparrow 0.30}$	56.98$_{\uparrow 0.68}$	71.50$_{\uparrow 0.47}$	44$_{\downarrow 4}$

Table 2. The results for cross-dataset evaluation on KITTI, BDD100K, and our DGL-MOTS. ↑ and ↓ indicate the change of performance on the metrics. The best and the second-best methods on KITTI and BDD100K are highlighted.

Track++ [45], TrackRCNN [39], and STEm-Seg [3], which only require an instance-level label for training [3]. All the methods are trained on the KITTI MOTS, BDD100K, and DGL-MOTS train sets separately and cross-validated on each dataset. In training, we designate no fixed number of total iterations and allow each method to be trained until performance asymptotically. The evaluation metrics are sMOTSA, MOTSA, IDS and HOTA from [39]. All experiments are conducted on one TITAN RTX GPU.

5.2. Dataset Statistics

Annotation volume is summarized in Table 1. We compare DGL-MOTS with BDD100K and KITTI MOTS in terms of the number of video clips, video frames, unique identities, instances, and instances per frame. In comparison, DGL-MOTS outperforms KITTI MOTS in evaluation metrics. Particularly, our instances per frame are around 1.5% higher than that of KITTI MOTS, which indicates that our dataset has a higher portion of scene complexity. BDD100K has the largest data volume among the three datasets, but its data is intensively selected from sequential video frames, which include redundant learning examples.

Instance variations are represented by the instance appearance change as well as the temporal description (as shown in Figure 4). The left figure illuminates the distribution of squared bounding-box size $\sqrt{wh}$ (where width w and height h); while the right figure shows the distribution of tracking length per instance. Figure 4 demonstrates that our dataset is not only more diverse in visual scale, but also longer in the temporal range for tracking.

Scene diversity is well-represented in our DGL-MOTS dataset, which includes more diverse driving scenes (Figure 1). Since DGL-MOTS provides recordings from four different states, it covers significantly more areas than KITTI MOTS that contains driving footage from a single city (Karlsruhe, German). Compared to BDD100K, our dataset include more road settings such as parking, residential, local, and high-way, while BDD100K only collects data of inner-city from the populous areas in the US [47].

5.3. Cross-Dataset Evaluations

Table 2 reports the results for the cross-dataset evaluations to assess our DGL-MOTS dataset. For the same method trained on different datasets, their performance gaps stem from the quality of the dataset (*i.e.*, annotation quality, data diversity, and temporal representation). Essentially, we observe two benefits of using our dataset in training over its counterparts. First, methods trained on DGL-MOTS all outperform their counterparts (with the same

[3]There are more progressive methods [35, 17, 28]. However, these methods needs extra information (i.e., flow field and LiDAR measurement) for supervision.

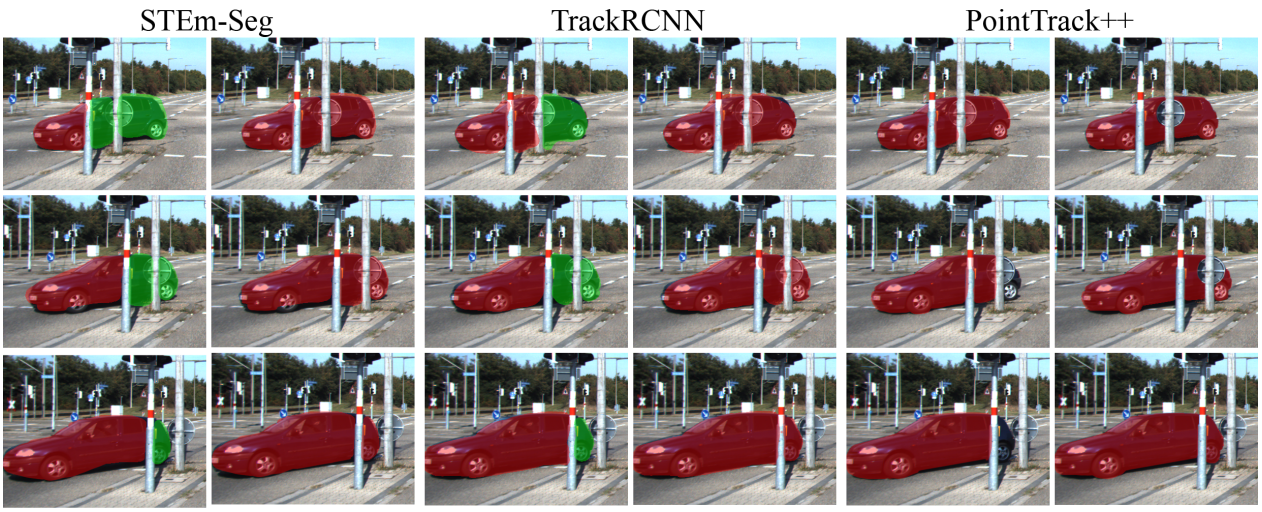

Figure 5. Qualitative examples of different methods tested on the KITTI test set. Results on the left column are methods trained on the KITTI while results on the right column are methods trained on our dataset. We can see the improvement of segmentation and tracking using our dataset. Masks of the same color indicate the tracking of the same instance.

Method	Dataset	Cars				Pedestrians			
		HOTA↑	sMOTSA↑	MOTSA↑	IDS↓	HOTA↑	sMOTSA↑	MOTSA↑	IDS↓
PointTrack++	KITTI	**67.28**	**82.82**	**92.61**	**36**	**56.67**	**68.13**	**83.67**	**36**
TrackRCNN	KITTI	56.92	77.20	87.92	92	42.08	47.43	67.12	78
STEm-Seg	KITTI	56.36	76.30	86.63	76	43.10	51.02	66.60	74
DG-Labler (Ours)	KITTI	**69.72**	**83.68**	**90.72**	**35**	**55.90**	**69.36**	**83.40**	**50**
PointTrack++	BDD100K	**68.33**	**84.60**	**93.20**	**49**	**55.42**	**64.56**	**80.29**	**45**
TrackRCNN	BDD100K	57.91	78.10	88.62	85	46.37	55.93	70.18	88
STEm-Seg	BDD100K	57.39	77.24	87.65	60	47.65	56.30	71.03	48
DG-Labler (Ours)	BDD100K	**67.89**	**85.30**	**91.70**	**58**	**56.23**	**65.43**	**81.40**	**48**
PointTrack++	DGL-MOTS	**68.10**	**83.62**	**92.39**	**42**	**59.10**	**71.90**	**86.60**	**32**
TrackRCNN	DGL-MOTS	58.63	78.8	88.9	88	48.23	60.29	76.10	77
STEm-Seg	DGL-MOTS	57.90	77.99	87.9	78	47.88	59.82	67.70	58
DG-Labler (Ours)	DGL-MOTS	**69.35**	**84.10**	**91.43**	**40**	**61.20**	**73.17**	**87.14**	**28**

Table 3. Comparison with the state-of-the-art methods on the KITTI MOTS, BDD100K, and DGL-MOTS. Each method is trained on KITTI MOTS, BDD100K, and DGL-MOTS separately. The best and the second-best methods are highlighted.

network architecture) on all metrics (Table 2). The improved performance indicates that, compared to KITTI and BDD100K, our dataset captures more general road settings and driving scenarios in training. Second, the DGL-MOTS dataset can train methods to achieve improved performance with a shorter schedule than methods trained on KITTI and BDD100K. For instance, TrackRCNN [39] and STEm-Seg [3] trained on DGL-MOTS with 20 epochs outperform its counterpart trained on KITTI and BDD100K with 40 epochs respectively.

In addition, we display the qualitative examples of each method from KITTI MOTS in Figure 5. The selected results also resonate with our quantitative analysis that methods trained on our DGL-MOTS dataset generally achieve improved performance in instance mask generation and tracking than their counterparts (with the same architecture) trained on KITTI MOTS. Both quantitative and qualitative results prove the advantages of the proposed DGL-MOTS dataset over the concurrent datasets.

5.4. Comparison to The State-Of-The-Art

This section presents the state-of-the-art comparison of our DG-Labeler on KITTI, BDD100K, and DGL-MOTS.

Quantitative results illuminates that DG-Labeler achieves the appealing performance on all metrics (on HOTA, sMOTSA, MOTSA, and IDS) among all methods. (Table 3). Also, DG-Labeler is on-par with PointTrack++ [45], the concurrent method. For instance, our margins over the strong methods (TrackRCNN [39] and STEm-Seg [3]) are around 3.53-13.36% for the car class and 9.5-21.93% for the pedestrian class on HOTA, sMOTSA, and MOTSA. The improvements suggest that DG-Labeler has a superior segmentation behavior to other recent methods. Meanwhile, DG-Labeler performs on par with the top-performing method, PointTrack++ [45] on all metrics. The reported results indicate that our DG-Labeler

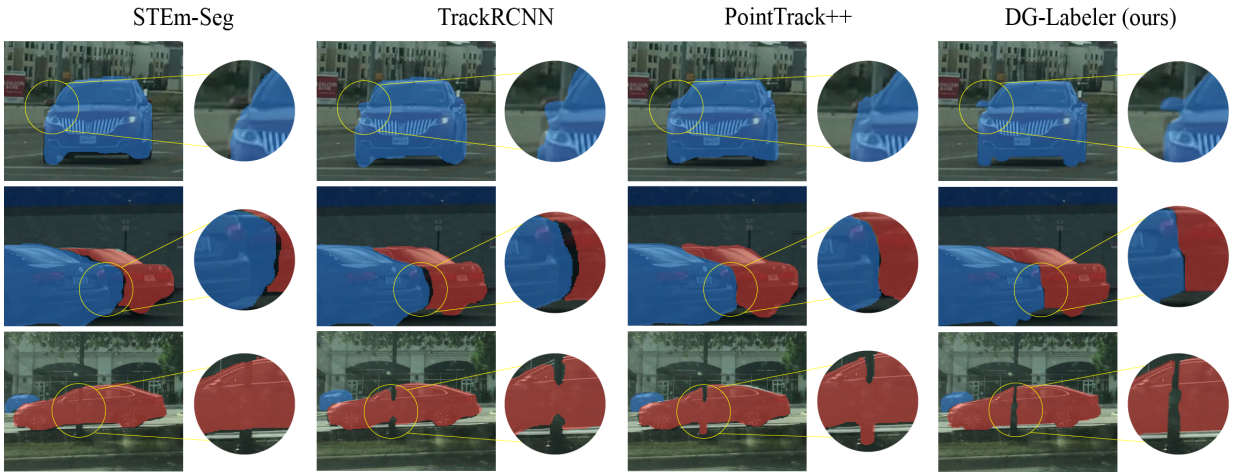

Figure 6. Qualitative examples of different methods on DGL-MOTS dataset. Compared to other methods, our DG-Labeler offers fine-grained instance masks. All methods are trained and tested on our DGL-MOTS dataset.

Method	Cars				Pedestrians			
	HOTA↑	sMOTSA↑	MOTSA↑	IDS↓	HOTA↑	sMOTSA↑	MOTSA↑	IDS↓
TrackRCNN	57.91	78.10	88.62	85	46.37	55.93	70.18	88
TrackRCNN+CC	$63.51_{\uparrow 5.6}$	$79.9_{\uparrow 1.8}$	$89.42_{\uparrow 0.8}$	$67_{\downarrow 18}$	$48.77_{\uparrow 2.4}$	$58.73_{\uparrow 2.8}$	$72.18_{\uparrow 2.0}$	$62_{\downarrow 26}$
TrackRCNN+DGM	$66.81_{\uparrow 8.9}$	$82.7_{\uparrow 4.6}$	$90.92_{\uparrow 2.3}$	$74_{\downarrow 11}$	$53.07_{\uparrow 6.7}$	$62.03_{\uparrow 6.1}$	$77.48_{\uparrow 7.3}$	$74_{\downarrow 14}$
TrackRCNN+CC+GL	$64.51_{\uparrow 6.6}$	$80.90_{\uparrow 2.8}$	$90.92_{\uparrow 2.3}$	$65_{\downarrow 20}$	$50.27_{\uparrow 3.9}$	$59.93_{\uparrow 4.0}$	$73.78_{\uparrow 3.6}$	**$55_{\downarrow 33}$**
TrackRCNN+CC+DGM	**$68.51_{\uparrow 10.6}$**	**$84.68_{\uparrow 6.4}$**	**$91.42_{\uparrow 2.8}$**	**$63_{\downarrow 22}$**	**$54.77_{\uparrow 8.4}$**	**$64.83_{\uparrow 8.9}$**	**$80.08_{\uparrow 9.9}$**	$60_{\downarrow 28}$
DG-Labeler (Ours)	**$69.35_{\uparrow 11.44}$**	**$85.30_{\uparrow 7.2}$**	**$91.70_{\uparrow 3.08}$**	**$58_{\downarrow 27}$**	**$56.23_{\uparrow 9.86}$**	**$65.43_{\uparrow 9.5}$**	**$81.40_{\uparrow 11.22}$**	**$48_{\downarrow 40}$**

Table 4. Ablation study results on the BDD100K. All methods are trained on the BDD100K training set. CC, DGM, and GL stand for collaborative connections, depth-granularity module, and GIoU loss respectively. We use the best models in training for testing. ↓ and ↑ indicate the performance gain to the baseline. The best and the second-best methods are highlighted.

is competitive with the existing best approaches.

Qualitative examples demonstrate the improved instance mask quality of our DG-Labeler over the counterpart methods (as shown in Figure 6). To demonstrate our advantage, we select some samples where other methods have trouble dealing with. Those cases include 1) objects with complex shapes ($i.e.$, wing mirrors or pedestrians), which is hard to depict sharp borders; 2) same class objects with overlapping. Other methods often get confused with the borders and fail to segment accurate boundaries; 3) Objects in separated parts ($i.e.$, occluded or truncated objects). Other methods may segment targets into separate objects or include occlusions as false positives. Based on the results, our DG-Labeler achieves an improved segmentation behavior in these cases because our depth-granularity module models the object spatial relations which offer more accurate descriptions of instance details and boundaries. Besides, our collaborative connections allow our segmentation and tracking head to accurately fire on the pixel of the instance instead of using the candidate proposals.

5.5. Ablation Study

We perform an ablation study on the BDD100K test set. Note our method is crafted on TrackRCNN [39], thus our baseline. By progressively integrating different contributing components: collaborative connections (CC), depth-granularity module (DGM), and GIoU loss (GL) (Sec. 3.4), to the baseline, we assess the contribution of each new component in DG-Labeler to TrackRCNN [39].

We present the results in Table 4. All of our components (CC, DGM, GL) assist in achieving improved performance. Particularly for a single module, the baseline with CC avoids inefficient proposal-based operations and performs predictions on the accurate RoIs, thus achieving improved performance in accuracy; DGM contributes the largest improvements in dense prediction (HOTA, sMOTSA, and MOTSA). Compared to the strong baseline TrackRCNN, our full model integrating all contributions obtains absolute gains of 11.44%, 7.2%, 3.08%, and 27 in terms of HOTA, sMOTSA, MOTSA, and IDS for car class and 9.86%, 9.5%, 11.22%, and 40 for pedestrian class respectively. More results are displayed in the supplementary materials.

6. Conclusion

In this work, we offer the DGL-MOTS Dataset for training MOTS algorithm as well as DG-Labeler for data annotation. We believe that our work holds valuable potentials to facilitate the progress of the MOTS studies.

References

[1] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. Posetrack: A benchmark for human pose estimation and tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5167–5176, 2018.

[2] Aditya Arun, CV Jawahar, and M Pawan Kumar. Weakly supervised instance segmentation by learning annotation consistent instances. In *European Conference on Computer Vision*, pages 254–270. Springer, 2020.

[3] Ali Athar, Sabarinath Mahadevan, Aljoša Ošep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. *arXiv preprint arXiv:2003.08429*, 2020.

[4] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9739–9748, 2020.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset. In *CVPR Workshop on the Future of Datasets in Vision*, volume 2, 2015.

[6] Yiming Cui, Liqi Yan, Zhiwen Cao, and Dongfang Liu. Tf-blender: Temporal feature blender for video object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8138–8147, October 2021.

[7] 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 *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[8] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 682–691, 2019.

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

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[11] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3838, 2019.

[12] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5356–5364, 2019.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 954–960, 2018.

[16] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 876–885, 2017.

[17] Aleksandr Kim, Aljoša Ošep, and Laura Leal-Taixé. Eagermot: Real-time 3d multi-object tracking and segmentation via sensor fusion.

[18] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9859–9868, 2020.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014.

[20] Chung-Ching Lin, Ying Hung, Rogerio Feris, and Linglin He. Video instance segmentation tracking with a modified vae architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13147–13157, 2020.

[21] Huaijia Lin, Xiaojuan Qi, and Jiaya Jia. Agss-vos: Attention guided single-shot video object segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3949–3957, 2019.

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] Dongfang Liu, Yiming Cui, Zhiwen Cao, and Yingjie Chen. A large-scale simulation dataset: Boost the detection accuracy for special weather conditions. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.

[25] Dongfang Liu, Yiming Cui, Yingjie Chen, Jiyong Zhang, and Bin Fan. Video object detection for autonomous driving: Motion-aid feature calibration. *Neurocomputing*, 409:1–11, 2020.

[26] Dongfang Liu, Yiming Cui, Xiaolei Guo, Wei Ding, Baijian Yang, and Yingjie Chen. Visual localization for autonomous driving: Mapping the accurate location in the city maze. In

2020 25th International Conference on Pattern Recognition (ICPR), pages 3170–3177, 2021.

[27] Dongfang Liu, Yiming Cui, Wenbo Tan, and Yingjie Chen. Sg-net: Spatial granularity network for one-stage video instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9816–9825, 2021.

[28] Jonathon Luiten, Tobias Fischer, and Bastian Leibe. Track to reconstruct and reconstruct to track. *IEEE Robotics and Automation Letters*, 5(2):1803–1810, 2020.

[29] Jonathon Luiten, Idil Esen Zulfikar, and Bastian Leibe. Unovost: Unsupervised offline video object segmentation and tracking. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2000–2009, 2020.

[30] Santiago Manen, Michael Gygli, Dengxin Dai, and Luc Van Gool. Pathtrack: Fast trajectory annotation with path supervision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 290–299, 2017.

[31] 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.

[32] Eslam Mohamed, Mahmoud Ewaisha, Mennatullah Siam, Hazem Rashed, Senthil Yogamani, and Ahmad El-Sallab. Instancemotseg: Real-time instance motion segmentation for autonomous driving. *arXiv preprint arXiv:2008.07008*, 2020.

[33] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the International Conference on Computer Vision*, 2017.

[34] Lorenzo Porzi, Markus Hofinger, Idoia Ruiz, Joan Serrat, Samuel Rota Bulo, and Peter Kontschieder. Learning multi-object tracking and segmentation from automatic annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6846–6855, 2020.

[35] Siyuan Qiao, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. *arXiv preprint arXiv:2012.05258*, 2020.

[36] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019.

[37] 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.

[38] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 international conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017.

[39] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7942–7951, 2019.

[40] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International journal of computer vision*, 101(1):184–204, 2013.

[41] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 107–122. Springer, 2020.

[42] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking. *arXiv preprint arXiv:1511.04136*, 2015.

[43] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016.

[44] Wenqiang Xu, Yonglu Li, and Cewu Lu. Srda: Generating instance segmentation annotation via scanning, reasoning and domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 120–136, 2018.

[45] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Liusheng Huang. Segment as points for efficient online multi-object tracking and segmentation. In *European Conference on Computer Vision*, pages 264–281. Springer, 2020.

[46] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5188–5197, 2019.

[47] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020.

[48] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020.