VSRD: Instance-Aware Volumetric Silhouette Rendering for Weakly Supervised 3D Object Detection

Supplementary Material

1. Additional Implementation Details

1.1. Multi-View 3D Auto-Labeling

1.1.1 Cuboid SDF

SDFs for primitives such as spheres and cuboids can be derived theoretically. Here, we introduce the SDF for cuboids. First, we define the cuboid in a local coordinate system with a dimension $\boldsymbol{d} \in \mathbb{R}^3_+$ as the following set of vertices $\mathcal{C}_{\boldsymbol{d}}$:

$$\mathcal{C}_{\boldsymbol{d}} = \{-\frac{\boldsymbol{d}_x}{2}, \frac{\boldsymbol{d}_x}{2}\} \times \{-\frac{\boldsymbol{d}_y}{2}, \frac{\boldsymbol{d}_y}{2}\} \times \{-\frac{\boldsymbol{d}_z}{2}, \frac{\boldsymbol{d}_z}{2}\} \,. \quad (1)$$

Accordingly, the *local* SDF $\bar{\mathcal{B}}(\cdot; \boldsymbol{d})$ for the cuboid $\mathcal{C}_{\boldsymbol{d}}$ can be derived as follows:

$$\begin{aligned} \bar{\mathcal{B}}(\boldsymbol{p}; \boldsymbol{d}) &= \| \max(\boldsymbol{q}, 0) \|_2 + \min(m, 0) \,, \quad (2) \\ \boldsymbol{q} &= |\boldsymbol{p}| - \boldsymbol{d} \,, \\ m &= \max(\boldsymbol{q}_x, \boldsymbol{q}_y, \boldsymbol{q}_z) \,. \end{aligned}$$

Next, we transform the cuboid SDF from the local coordinate system to the global one. In general, given a surface whose SDF is denoted by $\bar{\mathcal{F}}(\cdot)$, the SDF $\mathcal{F}(\cdot; \boldsymbol{R}, \boldsymbol{t})$ for the surface transformed by a rigid transformation $(\boldsymbol{R}, \boldsymbol{t}) \in \mathbf{SE}(3)$ is given by:

$$\mathcal{F}(\boldsymbol{p}; \boldsymbol{R}, \boldsymbol{t}) = \bar{\mathcal{F}}(\boldsymbol{R}^T(\boldsymbol{p} - \boldsymbol{t})) \,. \quad (3)$$

Therefore, the *global* cuboid SDF $\mathcal{B}(\cdot; \boldsymbol{d}, \boldsymbol{\ell}, \boldsymbol{R})$ parameterized by a dimension $\boldsymbol{d} \in \mathbb{R}^3_+$, location $\boldsymbol{\ell} \in \mathbb{R}^3$, and orientation $\boldsymbol{R} \in \mathrm{SO}(3)$ can be derived by transforming the *local* cuboid SDF $\bar{\mathcal{B}}(\cdot; \boldsymbol{d})$ from the local coordinate system to the global one with the rigid transformation $(\boldsymbol{R}, \boldsymbol{\ell})$, as follows:

$$\mathcal{B}(\boldsymbol{p}; \boldsymbol{d}, \boldsymbol{\ell}, \boldsymbol{R}) = \bar{\mathcal{B}}(\boldsymbol{R}^T(\boldsymbol{p} - \boldsymbol{\ell}); \boldsymbol{d}) \,. \quad (4)$$

1.1.2 Symmetric Shape Prior

Since the shapes of vehicles are often horizontally symmetrical in the local coordinate system, we incorporate this shape prior to each instance SDF in our proposed multi-view 3D auto-labeling. Given a *local* SDF $\mathcal{F}(\cdot)$, its horizontally symmetrical version $\overleftrightarrow{\mathcal{F}}(\cdot)$ is given by:

$$\overleftrightarrow{\mathcal{F}}(\boldsymbol{p}) = \mathcal{F}([|\boldsymbol{p}_x|, \boldsymbol{p}_y, \boldsymbol{p}_z]) \,. \quad (5)$$

This symmetric shape prior has the advantage that even if only one of the left or right sides of an instance is visible, the shape of the invisible part can be shared with the other instances via the hypernetwork.

1.1.3 Frame Sampling

Since our loss functions are based on multi-view 2D supervision, how to sample source frames $\mathcal{S}$ for each target frame t is important. To sample source frames, each of which includes as many instances in the target frame as possible, we sample source frames based on the perspective of *what percentage of instances in the target frame are included in each source frame*. Therefore, we first define a set of candidate source frames $\tilde{\mathcal{S}}(\eta)$ as the set of frames with the maximum number of elements, where the percentage of *target* instances in every source frame is greater than or equal to a certain threshold η as follows:

$$\tilde{\mathcal{S}}(\eta) = \max_{|\cdot|}(\{\mathcal{N} \ni t \mid \forall s \in \mathcal{N} : \frac{|\mathcal{I}_t \cap \mathcal{I}_s|}{|\mathcal{I}_t|} \geq \eta\}) \,, \quad (6)$$

where $\mathcal{I}_t$ and $\mathcal{I}_s$ denote the sets of instance IDs of the target and source frames, respectively. $\max_{|\cdot|}$ denotes the *max* operation that selects the set with the maximum number of elements. In our experiments, we set empirically $\eta = 0.5$, balancing the number of viewpoints and convergence time. In practice, we further sample a fixed number of frames from $\tilde{\mathcal{S}}(\eta)$ as evenly as possible due to implementation considerations to avoid large differences in the number of source frames between scenes. We sample 16 frames from $\tilde{\mathcal{S}}(0.5)$ and use it as the final set of source frames $\mathcal{S}$.

1.1.4 Ray Sampling

For each iteration in stochastic gradient descent, as in NeRF [2], we randomly sample a batch of rays used for volumetric rendering. However, in the case where the instance masks are given, it is inefficient to sample rays far away from any instance in the scene. Therefore, we propose an efficient ray sampling algorithm based on the instance masks. First, for each source frame $i \in \mathcal{S}$, we extract the polygon for each instance by a contour finder. Then, we theoretically derive the 2D SDF $\mathcal{P}_{in}(\cdot)$ for the n-th polygon in frame i. Then, we derive the 2D SDF $\mathcal{P}_i(\cdot)$ for the union of all the polygons in frame i as $\mathcal{P}_i(\boldsymbol{p}) = \min(\mathcal{P}_{i1}(\boldsymbol{p}), \ldots, \mathcal{P}_{iN_i}(\boldsymbol{p}))$, where N_i denotes the number of *target* instances in frame i. Then, we generate a *soft* instance mask M_i via *soft rasterization* [1] as follows:

$$M_i(\boldsymbol{p}) = \Phi(-\mathcal{P}_i(\boldsymbol{p})/\tau) \,, \quad (7)$$

where $\boldsymbol{p} \in \mathbb{R}^2$ denotes a pixel coordinate, $\Phi(\cdot)$ denotes the Sigmoid function, and τ denotes the temperature parameter

that controls the degree of relaxation, indicating that as it becomes higher, rays farther away from each instance are sampled. Then, we normalize the soft instance mask M_i across all the source frames $\mathcal{S}$ as follows:

$$\tilde{M}_i(\boldsymbol{p}) = \frac{M_i(\boldsymbol{p})}{\sum_{j \in \mathcal{S}} \sum_{\boldsymbol{q}} M_j(\boldsymbol{q})} \,. \tag{8}$$

Finally, we sample a batch of rays from the multinomial distribution based on the normalized soft instance mask $\tilde{M}_i(\cdot)$. This mechanism enables us to intensively sample rays that are likely to hit the surface of each instance.

1.2. Monocular 3D Object Detection

To compare our method and Autolabels [5] with WeakM3D [3], we modify the architecture of WeakM3D so that it can be trained in a supervised manner using pseudo labels. More specifically, we train dimension and confidence heads in addition to the existing location and orientation heads using the same supervised loss as MonoDIS [4]. We call this model S-WeakM3D. Note that for the comparison with methods other than WeakM3D, we do not modify the architectures and loss functions.

1.2.1 Architecture

Dimension Head WeakM3D utilizes prior knowledge about the typical dimension for category *Car* and freezes it by setting the width, height, and length to 1.8, 1.6, and 4.0, respectively. This is because relying only on the 3D alignment loss utilizing LiDAR point clouds struggles to optimize the location and dimension parameters jointly. In order to make full use of the pseudo labels generated by the proposed auto-labeling, we modify the original network of WeakM3D by adding a simple dimension head, which has the same architecture as the original location head, to estimate the dimension of each instance as follows:

$$\hat{\boldsymbol{d}} = \boldsymbol{d}_{\text{min}} + (\boldsymbol{d}_{\text{max}} - \boldsymbol{d}_{\text{min}}) \odot \Phi(\mathcal{H}_{\text{dim}}(F; \boldsymbol{\psi}_{\text{dim}})) \,, \tag{9}$$

where $\boldsymbol{d}_{\text{min}} = [1.5, 1.5, 3.0]$ and $\boldsymbol{d}_{\text{max}} = [2.0, 2.0, 5.0]$ denote the pre-defined minimum and maximum dimensions, respectively. As we assume we cannot access any 3D bounding boxes in the dataset, they are determined based on not the statistics of the dataset but the dimensions of typical production cars. F denotes the RoI-aligned feature maps for each instance, $\mathcal{H}_{\text{dim}}(\cdot; \boldsymbol{\psi}_{\text{dim}})$ denotes the dimension head parameterized by $\boldsymbol{\psi}_{\text{dim}}$, and $\Phi(\cdot)$ denotes the Sigmoid function. We implement the dimension head as an MLP with two hidden layers, each of which has 256 channels.

Confidence Head Following MonoDIS [4], we train the network to estimate not only the 3D bounding box but also a confidence score that represents the quality of the predicted 3D bounding box in a self-supervised manner. We add a simple confidence head, which has the same architecture as the original location head, to estimate the confidence of the predicted 3D bounding box as follows:

$$\hat{c} = \Phi(\mathcal{H}_{\text{conf}}(F; \boldsymbol{\psi}_{\text{conf}})) \,, \tag{10}$$

where $\mathcal{H}_{\text{conf}}(\cdot; \boldsymbol{\psi}_{\text{conf}})$ denotes the confidence head parameterized by $\boldsymbol{\psi}_{\text{conf}}$. We implement the confidence head as an MLP with two hidden layers, each of which has 256 channels. We train the confidence head with the self-supervised loss explained in Sec. 1.2.2. The output confidence score is further multiplied by the classification score output by the off-the-shelf 2D detector and used as the final score to filter low-quality predictions during inference.

1.2.2 Loss Functions

Distentangled loss For bounding box regression, we employ the same disentangled loss as MonoDIS [4] as follows:

$$\begin{aligned} \mathcal{L}_{\text{box}}(\hat{\boldsymbol{d}}, \hat{\boldsymbol{l}}, \hat{\theta}, \boldsymbol{d}, \boldsymbol{l}, \theta) = &\|B(\hat{\boldsymbol{d}}, \boldsymbol{l}, \theta) - B(\boldsymbol{d}, \boldsymbol{l}, \theta)\|_{\text{H}} + \\ &\|B(\boldsymbol{d}, \hat{\boldsymbol{l}}, \theta) - B(\boldsymbol{d}, \boldsymbol{l}, \theta)\|_{\text{H}} + \\ &\|B(\boldsymbol{d}, \boldsymbol{l}, \hat{\theta}) - B(\boldsymbol{d}, \boldsymbol{l}, \theta)\|_{\text{H}} \,, \end{aligned} \tag{11}$$

where $\hat{\boldsymbol{d}}$, $\hat{\boldsymbol{l}}$, and $\hat{\theta}$ denote the predicted dimension, location, and orientation, respectively, and $\boldsymbol{d}$, $\boldsymbol{l}$, and θ denote the ground truth dimension, location, and orientation, respectively. $B(\boldsymbol{d}, \boldsymbol{l}, \theta)$ denotes the 3D bounding box decoded from dimension $\boldsymbol{d}$, location $\boldsymbol{l}$, and orientation θ. Actually, we minimize the confidence-based weighted regression loss $\tilde{\mathcal{L}}_{\text{box}}$ instead of the original regression loss $\mathcal{L}_{\text{box}}$, as explained in Sec. 3.2.2 in the main paper.

Confidence Loss For confidence learning, we employ the same self-supervised loss as MonoDIS [4] as follows:

$$\mathcal{L}_{\text{conf}}(\hat{c}, \hat{\boldsymbol{d}}, \hat{\boldsymbol{l}}, \hat{\theta}, \boldsymbol{d}, \boldsymbol{l}, \theta) = \text{BCE}(\hat{c}, c) \,, \tag{12}$$

$$c = \exp(-\lfloor \mathcal{L}_{\text{box}}(\hat{\boldsymbol{d}}, \hat{\boldsymbol{l}}, \hat{\theta}, \boldsymbol{d}, \boldsymbol{l}, \theta) \rfloor) \,, \tag{13}$$

where $\hat{c}$ denotes the predicted confidence, $\text{BCE}(\cdot, \cdot)$ denotes the binary cross entropy, and $\lfloor \cdot \rfloor$ denotes the *stop gradient* operation whereby the gradients are not propagated through the box regression loss $\mathcal{L}_{\text{box}}$.

2. Additional Evaluation Results

2.1. Monocular 3D Object Detection

2.1.1 Weakly Supervised Setting

Tab. 1 shows the additional evaluation results of our method compared with the existing weakly supervised and fully

Table 1. Evaluation results of monocular 3D object detection on the KITTI-360 validation set. *Reproduced with the official code. †CAD models are used as extra data. ‡$M(D)$ indicates that detection model D is employed for model-agnostic method M.

Method	Weak Supervision		Full Supervision	$AP_{BEV}/AP_{3D}@0.3$		$AP_{BEV}/AP_{3D}@0.5$	
	LiDAR	Masks	3D Boxes	Easy	Hard	Easy	Hard
WeakM3D* [3]	✓	✓		49.38/44.26	41.53/34.91	17.25/4.64	13.87/3.45
Autolabels*†‡ [5] (S-WeakM3D)	✓	✓		55.55/10.04	51.59/8.50	36.06/1.56	28.12/1.13
VSRD‡ (S-WeakM3D)		✓		62.77/57.28	57.35/51.79	31.84/29.50	30.04/24.93
VSRD‡ (MonoFlex)		✓		**70.04/65.09**	**60.53/55.75**	**50.59/32.52**	**48.83/25.70**
VSRD‡ (MonoDETR)		✓		54.97/50.13	49.81/46.13	38.09/29.52	31.68/24.25
MonoFlex* [7]			✓	80.47/78.15	72.97/68.74	66.81/60.46	57.37/49.38
MonoDETR* [6]			✓	73.21/72.67	68.58/66.27	61.47/58.35	54.53/49.91

Table 2. Evaluation results of semi-supervised monocular 3D object detection on the KITTI validation set.

Method	Ratio	$AP_{BEV}/AP_{3D}@0.7$		
		Easy	Moderate	Hard
MonoFlex [7]	1.00	28.17/23.64	21.92/17.51	19.07/14.83
VSRD (MonoFlex)	0.00	3.65/0.34	2.51/0.23	1.98/0.21
	0.25	24.55/14.62	17.74/10.69	15.67/8.95
	0.50	**29.38**/17.44	21.72/12.40	18.69/10.65
	0.75	**34.32/23.79**	**24.87/17.60**	**21.45/14.97**

supervised methods. As with Tab. 4 in the main paper, our method demonstrates a significant superiority over WeakM3D [3] across all the metrics while eliminating the need for LiDAR points for 3D supervision. Moreover, the detector trained on the pseudo labels generated by the proposed auto-labeling outperforms that trained on the pseudo labels generated by Autolabels [5].

2.1.2 Semi-Supervised Setting

In addition to MonoDETR [6], we also conduct the same experiments as Sec. 4.4.2 in the main paper employing MonoFlex [7]. Tab. 2 shows the performance of the detector pre-trained on the KITTI-360 dataset in a weakly supervised manner with the proposed auto-labeling and then fine-tuned on a subset of the KITTI dataset in a supervised manner. As with Tab. 5 in the main paper, the zero-shot performance is quite low due to the characteristic that monocular depth estimation is greatly affected by the differences in camera parameters, but the performance of the detector fine-tuned on only 75% of the labeled data significantly outperforms that trained on the whole data from scratch, highlighting the broad applicability of our method.

3. Additional Visualization Results

Figs. 1 to 4 show the additional visualization results of the proposed multi-view 3D auto-labeling, weakly supervised monocular 3D object detection, and semi-supervised monocular 3D object detection employing MonoDETR [6] and MonoFlex [7], respectively. In particular, as can be seen from Fig. 2, it is worth noting that WeakM3D [3] struggles to estimate the orientations of laterally moving objects accurately as it assumes that most objects are facing forward, whereas our method leverages the pseudo labels generated by the proposed auto-labeling as 3D supervision without any priors, leading to more accurate orientation estimation.

References

[1] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7708–7717, 2019. 1

[2] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1

[3] Liang Peng, Senbo Yan, Boxi Wu, Zheng Yang, Xiaofei He, and Deng Cai. Weakm3d: Towards weakly supervised monocular 3d object detection. *arXiv preprint arXiv:2203.08332*, 2022. 2, 3, 4

[4] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1991–1999, 2019. 2

[5] Sergey Zakharov, Wadim Kehl, Arjun Bhargava, and Adrien Gaidon. Autolabeling 3d objects with differentiable rendering of sdf shape priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12224–12233, 2020. 2, 3, 4

[6] Renrui Zhang, Han Qiu, Tai Wang, Ziyu Guo, Ziteng Cui, Yu Qiao, Hongsheng Li, and Peng Gao. Monodetr: Depth-guided transformer for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9155–9166, 2023. 3, 5

[7] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3289–3298, 2021. 3, 5

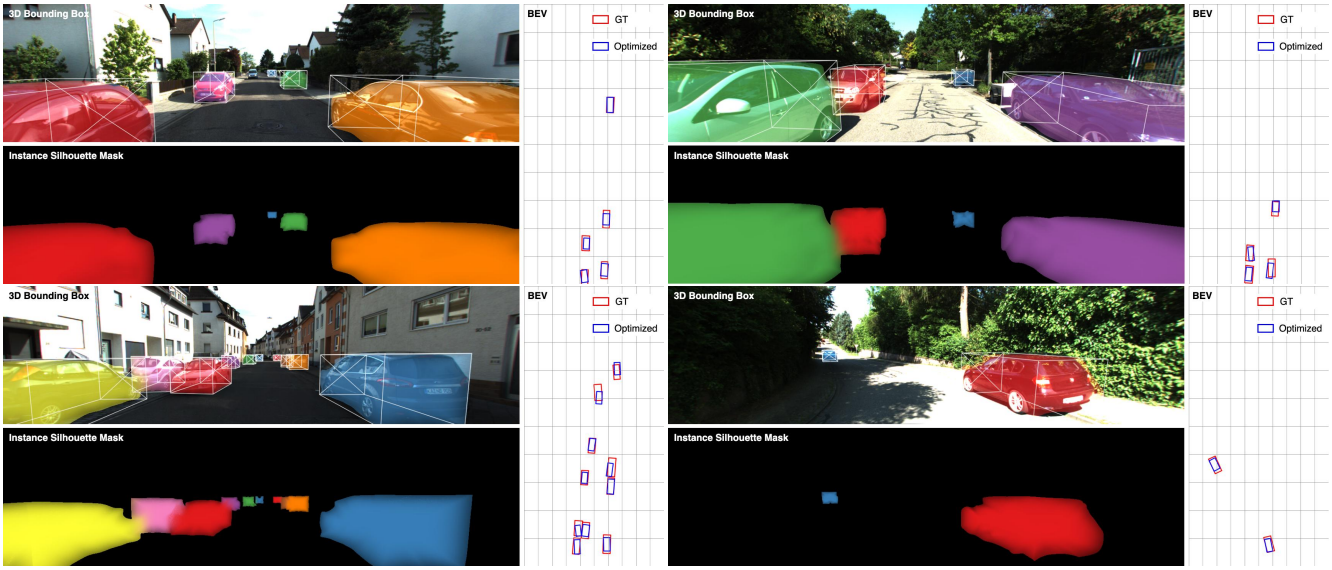

Figure 1. Visualization results of the optimized 3D bounding boxes (1st row) and rendered instance masks (2nd row). We assign a unique color to each instance, and each pixel is colored as the weighted summation based on the rendered soft instance label.

(a) WeakM3D

(b) Autolabels

(c) VSRD

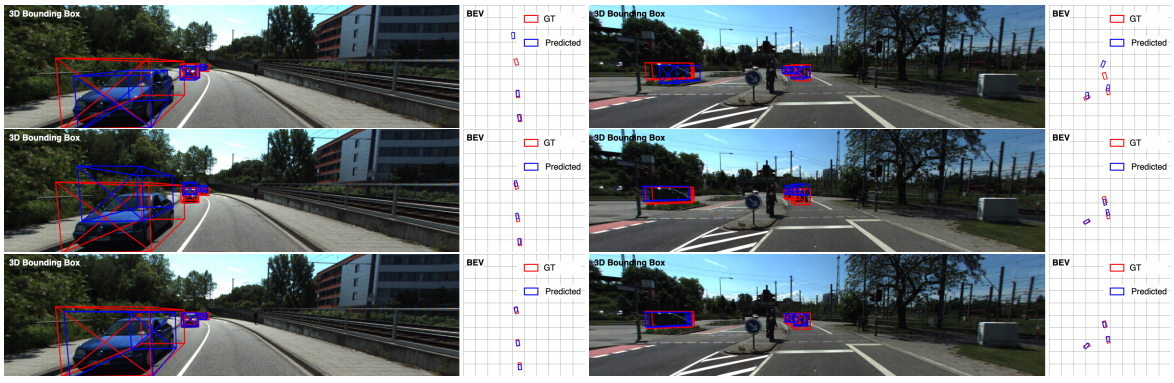

Figure 2. Visualization results of weakly supervised monocular 3D object detection compared with Autolabels [5] and WeakM3D [3]. The ground truth and predicted bounding boxes are drawn in red and blue, respectively.

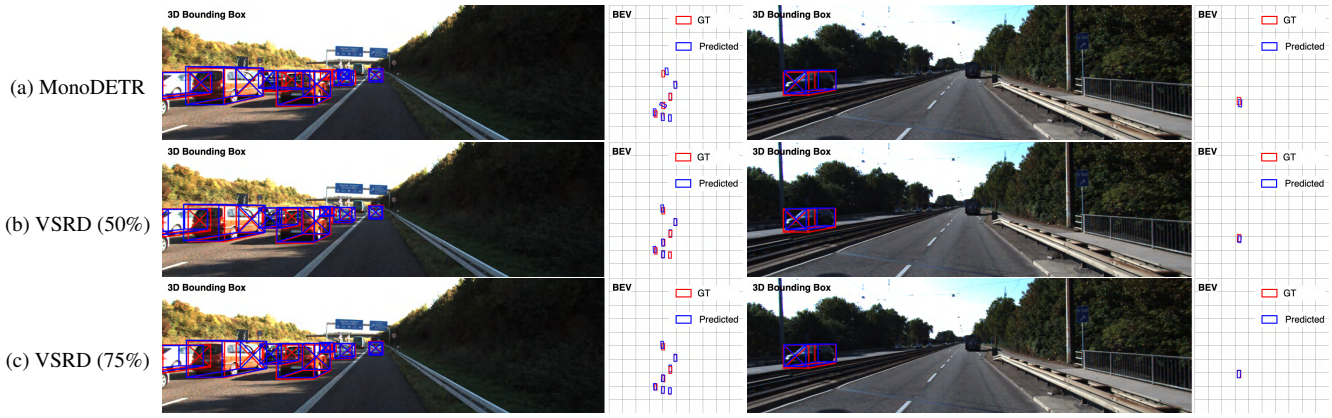

Figure 3. Visualization results of semi-supervised monocular 3D object detection compared with MonoDETR [6]. The ground truth and predicted bounding boxes are drawn in red and blue, respectively.

Figure 4. Visualization results of semi-supervised monocular 3D object detection compared with MonoFlex [7]. The ground truth and predicted bounding boxes are drawn in red and blue, respectively.