Supplementary Material
NeurOCS: Neural NOCS Supervision for Monocular 3D Object Localization

Zhixiang Min[1] Bingbing Zhuang[2] Samuel Schulter[2] Buyu Liu[2]
Enrique Dunn[1] Manmohan Chandraker[2]
[1]Stevens Institute of Technology [2]NEC Laboratories America

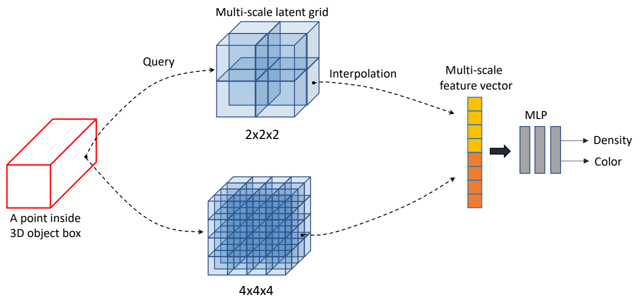

Figure S1. **Illustration of our grid NeRF** that decodes color and density from a feature vector queried from latent grids. We use latent grids with five scales but only two are visualized for clarity.

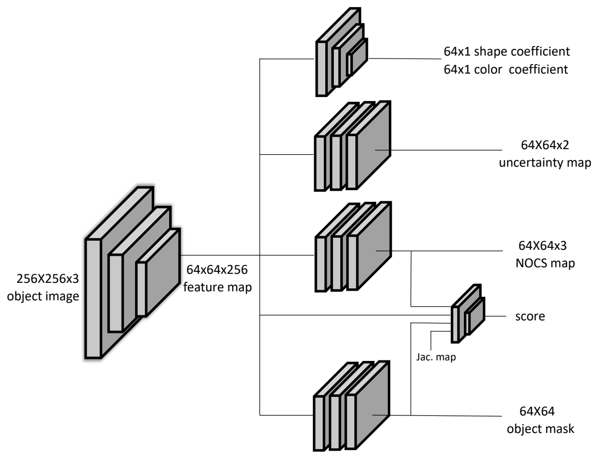

Figure S2. **Structure of our image-conditioned network** consisting of a backbone feature extraction network and a number of regression heads for different predictions.

This supplementary material includes additional technical explanations and experimental results that are not in the main paper due to space limit.

S1. Supportive Explanations

NeRF grid configurations. Our NeRF representation rests on latent feature grids that can be efficiently decoded into color and density with a small MLP. We use the implementation from [10] for latent grids that correspond to the space of a unit cubic. In particular, we adopt the multi-resolution representation consisting of five latent grids, each with size $(2\times2\times2)$, $(4\times4\times4)$, $(8\times8\times8)$, $(16\times16\times16)$, and $(32\times32\times32)$. Each vertex on the grid stores a 4-dimensional learnable feature vector. Given a sampled point inside the 3D object box, we use its size-normalized NOCS to query each latent grid by trilinear interpolation, resulting in a total of $5\times4=20$ dimensional feature vector after concatenation. The feature vector is passed to an MLP containing three 64-channel hidden layers with ReLU activation, yielding color and density. We illustrate this procedure in Fig. S1, with two resolution scales for clarity of visualization. For each batch iteration during training, we randomly sample 768 rays per object to enforce rendering losses.

Base 3D detector. It is worth stressing that the training of NeurOCS is agnostic to the base 3D detector – NeurOCS takes ground truth 2D boxes as input in training, and during inference it is combined with a base 3D detector of choice, taking its 2D boxes as input. Also note the object size prediction is not needed during training. This makes NeurOCS a standalone NOCS-based framework that can be flexibly combined with different base 3D detectors. In addition to DID-M3D, we show in next section that it also works well with another state-of-the-art 3D detector DEVIANT [6]. As future work, we envision that NeurOCS has potential to improve Lidar-based 3D detection as well.

Network architecture. We illustrate our network pathways in Fig. S2. We crop objects using 2D boxes and resize to 256×256 as input to our backbone network, a ResNet50 pretrained on ImageNet. We extract its 8×8 feature map at the fourth layer, upsample to 64×64 by bilinear interpolation, and then pass to a few regression heads. We apply three separate heads to predict the NOCS map, the uncertainty map (detailed later), and the foreground object mask, each using three 1×1 convolutional layers with BachNorm and ReLu

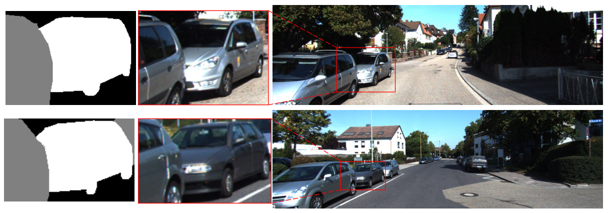

Figure S3. **Illustration of our ground-truth object mask.** The white, black, and gray pixels indicate foreground, background, and unknown regions, respectively.

Module	Loss Name	Type	Weight	Region	Note
NeRF	Occupancy Loss	L2	1	FG + BG	
	RGB Loss	L2	3	FG	
	KL Divergence	KL	1	-	
	Lidar Loss	L2	0.5	FG & Valid	Optional
	LiComp Loss	L2	0.2	FG & Valid	Optional
	Dense Prior	L1	0.002	3D Grid	Conditional
Image	FG Loss	L2	1	All	
	IoU Score Loss	L2	1	-	
	Reproj. Loss	L2	2	FG	
Both	NOCS Consistency	L2	1	FG & Occ	

FG = Foreground, BG = Backgroud

Table S1. **A summary of our losses**, including NeRF losses and image-conditioned regression losses. See text for details.

activation. The shape and color coefficients are predicted by a 3-layer MLP with the average-pooled feature map as input. The score prediction head uses two 1×1 convolutional layers followed by average-pooling and a linear layer to regress a confidence score. It takes as input the object feature map, the predicted NOCS map, the predicted foregound object mask, and the Jacobian map.

Foreground object masks. We demonstrate our ground-truth object mask with two example objects in Fig. S3. The foreground and background regions provide shape constraints in our occupancy loss akin to the machinery of shape-from-silhouette. Pixels on other object instances are ignored as they are likely occluding objects that obscure the shape boundary.

Shape Regularization. We demonstrate the impact of the KL divergence loss in Fig. S4(a), illustrate the visual hull ambiguity and that the dense prior improves the shape Fig. S4(b).

Loss. We jointly train the image-conditioned regression network and the NeRF by combining all their losses. In Tab. S1, we detail the training losses composition, including their weight, application scope and condition. The occupancy loss is applied on foreground and background, while skipping unknown regions. The RGB loss is only applied to the foreground region. If a point from Lidar or its completion resides inside the object 3D box and projects to the foreground region, it induces a valid NOCS point for supervision. The dense prior loss is applied to randomly sampled points inside the 3D latent grid; and it is only applied in absence of Lidar/LiComp losses, hence marked as conditional. The NOCS consistency loss is applied to foreground regions and weighted by a detached occupancy map rendered from NeRF. We also add an unsupervised reprojection loss [2], and similarly to [2] we learn a per-pixel aletoric uncertainty map (σ_x, σ_y) for the NOCS projection on image in both horizontal and vertical diction, by optimizing $\frac{|\Delta p_x|^2}{\sigma_x^2} + \log \sigma_x^2$, where $|\Delta p_x|^2$ indicates the reprojection error in x direction; same applies to σ_y. The confidence $(\frac{1}{\sigma_x}, \frac{1}{\sigma_y})$ is multiplied with the foreground probability to weigh each pixel in PnP optimization, which also optimizes the reprojection error loss. The weights combining the losses are chosen empirically based on the validation set, and we note that the performance is not sensitive to these parameters in practice.

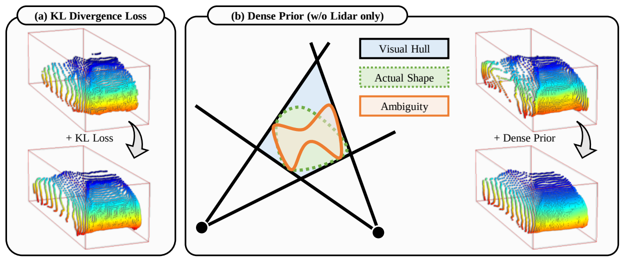

Figure S4. **Shape Regularizations.** (a) An actual example showing the KL loss yields cleaner shape. (b) An illustration on the visual hull ambiguity and the impact of the dense shape prior.

Method	Venue	3D AP_{40} - Val			BEV AP_{40} - Val		
		Easy	Mod	Hard	Easy	Mod	Hard
DEVIANT [6]	ECCV22	25.09	16.99	15.30	33.69	23.10	20.67
NeurOCS-M + [6]	CVPR23	27.70	18.92	16.28	35.31	25.11	21.87
NeurOCS-MLC + [6]	CVPR23	27.83	18.92	16.20	35.14	24.77	21.43

Table S2. Evaluation with DEVIANT [6] as our base 3D detector.

Method	Venue	3D AP_{40} - Test			BEV AP_{40} - Test		
		Easy	Mod	Hard	Easy	Mod	Hard
LPCG [11]	ECCV22	25.56	17.80	15.38	35.96	24.81	21.86
CMKD [4]	ECCV22	28.55	18.69	16.77	38.98	25.82	22.80
NeurOCS	CVPR23	29.89	18.94	15.90	37.27	24.49	20.89

Table S3. Additional comparison with LPCG [11] and CMKD [4] on KITTI test set.

Data augmentation. During the training, we apply data augmentation on the ground truth 2D boxes, including flipping and bounding box perturbation similar to [8].

S2. Additional Experimental Results

S2.1. Additional Results on KITTI

Base 3D detector. We demonstrated good performance with DID-M3D [12] as our base 3D detector in the main paper. Here, we evaluate performance by combining NeurOCS with

	PnP Only			+ Fusion		
	Easy	Moderate	Hard	Easy	Moderate	Hard
RLC	27.55	18.27	15.75	30.79	20.78	17.37
RL	27.50	18.36	15.77	30.45	20.71	17.45
RC	27.08	18.10	15.28	30.74	20.67	17.41
R	22.59	15.81	13.07	27.26	19.10	16.48

Table S4. AP_{3D} from directly training with raw depth supervision from Lidar and its completion w/o NeRF.

another base 3D detector, namely DEVIANT [6]. As can be seen in Tab. S2, NeurOCS improves over DEVIANT with the scale fusion, whether the shape is trained with mask only or with additional Lidar supervision. This further consolidates the value of NeurOCS as a standalone shape-based 3D localization framework.

Additional comparison on KITTI benchmark. In the main paper, we primarily compare with existing methods that rely on the original 3D box annotations from KITTI to train the 3D detector. However, it is also worth discussing recent works LPCG [11] and CMKD [4] that improve performance by leveraging additional pseudo ground truth from extra large-scale unlabeled sequences with Lidar. While this line of work is orthogonal to ours, we provide performance comparison in Tab. S3. As shown, the accuracy from NeurOCS is superior to LPCG and comparable to CMKD.

w/o NeRF results. In the main paper, we have compared the PnP results from our NeRF-based method against directly training with raw depth supervision from Lidar and its completion w/o NeRF. Here, we report in Tab. S4 the full AP_{3D} for the "w/o NeRF" setting including the results after scale fusion. Note that we always include the reprojection error loss as it was shown helpful [2] for NOCS learning. As can be seen, the direct training yields good performance thanks to the efficacy of our framework, but lags behind the results when NeRF is applied to serve as a bridge between the raw Lidar data and the NOCS network.

Network architectures. We study the performance with different backbone network architectures and different NeRF configurations. Our default setup uses ResNet50 and the grid NeRF with five resolution scales as described in Sec. S1. We first change ResNet50 to ResNet18 and ResNet34, only observing slightly degraded performance. This indicates NeurOCS does not heavily rely on powerful backbone, and a lighter backbone is viable in latency-sensitive applications. Next, we reduce the number of NeRF scales to 4 (finest scale with $16 \times 16 \times 16$ grids) and 3 (finest scale with $8 \times 8 \times 8$ grids), where the performance remains good. We further substitute the grid-based NeRF to a MLP-based NeRF with positional encoding same as [5]. In practice, we observe the MLP-based NeRF leads to nearly two times slower training process and higher memory occupancy due to its large MLP network, while the performance drops slightly.

Module	Model	PnP Only			+ Fusion		
		Easy	Moderate	Hard	Easy	Moderate	Hard
Default	ResNet50+5 Grid Scales	27.92	18.49	15.78	31.24	21.01	17.70
Backbone	ResNet18	27.04	18.21	15.39	30.18	20.80	17.44
	ResNet34	27.00	18.19	15.47	30.39	20.71	17.38
NeRF	4 Grid Scales	27.80	18.49	15.81	30.93	20.90	17.56
	3 Grid Scales	28.02	18.62	15.89	31.16	21.02	17.64
	CodeNeRF	27.41	18.23	15.10	30.53	20.77	17.45

Table S5. Evaluation with AP_{3D} on different backbone network architectures and NeRF configurations.

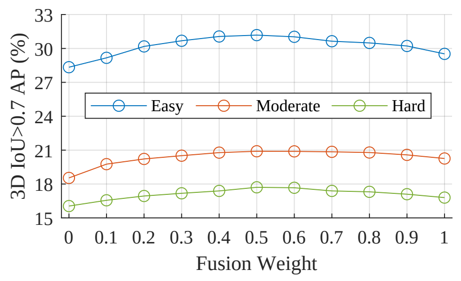

Figure S5. **Effect of Fusion Weight.** Lower weight value indicates lower impact from direct depth estimation and higher impact from scale in PnP solution.

Scale fusion. In the main paper, our scale fusion is described as an averaging between scales in PnP and network-predicted object depth. In fact, we have studied the generic linear combination of the two, *i.e.*,

$$\begin{aligned} \mathbf{s}' &= \frac{w \cdot d_{pred} + (1-w) \cdot [\mathbf{t}]_z}{2 \cdot [\mathbf{t}]_z} \mathbf{s}, \\ \mathbf{t}' &= \frac{w \cdot d_{pred} + (1-w) \cdot [\mathbf{t}]_z}{2 \cdot [\mathbf{t}]_z} \mathbf{t}. \end{aligned} \tag{S1}$$

where w is a balancing weight. We study different weight values in Fig. S5, and empirically found 0.5 (*i.e.* averaging) gives the overall best result, which indicates similar reliability for direct depth prediction and object size prediction that gives the metric scale in PnP solutions. We have also attempted to learn the fusion using networks akin to [7], but found this scheme to be superior in practice.

Visual scope. In the main paper we have studied the behavior of object-centric and scene-centric training in NeurOCS with NeRF. Here, we study the performance when training NOCS with Lidar directly without using NeRF-rendered supervision. This allows us to focus on the image-conditioned regression branch, although we observe the NeRF branch behaves similarly in the two schemes. The results are shown in Tab. S6. In addition to the vanilla scene-centric scheme, we replace its NOCS map prediction with the one from object-centric scheme, while keeping its object mask and score prediction; this setting is denoted as "scene-centric++". As can be seen, while a performance gap exists between the object-centric and scene-centric training, scene-centric++

	PnP Only			+ Fusion		
	Easy	Moderate	Hard	Easy	Moderate	Hard
object-centric	27.92	18.49	15.78	31.24	21.01	17.70
scene-centric (resnet)	26.39	17.62	14.61	29.40	20.20	16.94
scene-centric++ (resnet)	25.30	18.02	15.64	28.54	20.60	17.57
scene-centric-L (resnet)	25.95	17.35	14.51	29.57	20.39	17.06
scene-centric (dla)	25.04	16.89	14.03	28.48	19.34	16.26
scene-centric++ (dla)	26.00	18.05	15.81	29.25	20.60	17.55

Table S6. Additional study on the benefits of object-centric training.

	PnP only			+ Fusion		
	Easy	Moderate	Hard	Easy	Moderate	Hard
NeurOCS-MLC	28.06	18.52	15.36	31.20	20.99	17.55
NeurOCS-ML	28.23	18.58	15.49	30.79	20.80	17.48
NeurOCS-MC	28.07	18.69	15.82	30.93	20.87	17.46
NeurOCS-M	27.66	18.42	15.54	30.66	20.77	17.49

Table S7. AP_{3D} from NeurOCS without using the uncertainty map.

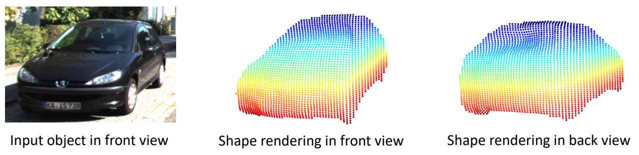
Input object in front view
Shape rendering in front view
Shape rendering in back view

Figure S6. Shape visualization for an example object instance from the validation set.

largely reduces the gap especially for the Moderate and Hard cases. This demonstrates the benefits of object-centric training towards more accurate NOCS map predictions. We also increase the network capacity in the scene-centric NOCS regression head by doubling its number of layers, but do not observe consistent improvements, as shown in “scene-centric-L”. Furthermore, we evaluate scene-centric scheme with DLA as the backbone, and similar results are observed.

Uncertainty map. As aforementioned, we learn a uncertain map for NOCS predictions using the reprojection loss, for both our w/ NeRF settings (NeurOCS) and those w/o NeRF baselines. We found that including the uncertainty map in the weight of PnP optimization improves the final performance. Here, we report in Tab. S7 the accuracy of NeurOCS in the case of without using it.

Note on NeRF. While the shape rendering from NeRF is only required during training, we observe that it also works well for the validation set. We demonstrate this in Fig. S6 with an example object instance from the validation set.

Qualitative results. We demonstrate the output of NeurOCS by running on KITTI raw sequences that have no overlap with the training set. In particular, the sequences are from the KITTI-Tracking subset where ground truth 3D boxes are provided for every frame. These results are shown in the attached supplementary video.

Pedestraian	DID-M3D			NeurOCS-MLC		
	Easy	Moderate	Hard	Easy	Moderate	Hard
	11.27	8.75	7.15	12.13	9.21	7.40

Table S8. AP_{3D} on the pedestrian class in KITTI validation set.

Difficulty	Method	AP_{3D}				APH_{3D}			
		All	0-30m	30-50m	50m-∞	All	0-30m	30-50m	50m-∞
Level_1	PCT [15]	0.89	3.18	0.27	0.07	0.88	3.15	0.27	0.07
	MonoJSG [9]	0.97	4.65	0.55	0.10	0.95	4.59	0.53	0.09
	DEVIANT [6]	2.69	6.95	0.99	0.02	2.67	6.90	0.98	0.02
	NeurOCS (PnP)	1.67	4.28	0.74	0.02	1.66	4.26	0.73	0.02
	NeurOCS (Fusion)	2.44	6.35	0.97	0.04	2.43	6.31	0.97	0.04
Level_2	PCT [15]	0.66	3.18	0.27	0.07	0.66	3.15	0.26	0.07
	MonoJSG [9]	0.91	4.64	0.55	0.09	0.89	4.65	0.53	0.09
	DEVIANT [6]	2.52	6.93	0.95	0.02	2.50	6.87	0.94	0.02
	NeurOCS (PnP)	1.56	4.26	0.71	0.02	1.55	4.24	0.70	0.02
	NeurOCS (Fusion)	2.29	6.32	0.94	0.03	2.28	6.29	0.93	0.03

Table S9. Evaluation on Waymo dataset with $IoU \geq 0.7$.

Other object classes. We report in Tab. S8 the results on the pedestrian class; we omit cyclists as their instance masks are not provided by [3]. As shown, NeurOCS improves upon its base detector DID-M3D despite the non-rigid object topology, further indicating its robustness.

Computation efficiency. The runtime of our framework without TTA costs 50ms on average for each frame in KITTI object dataset on a single RTX2080. The ResNet50 backbone takes about 15ms and the pose solver costs 35ms, while the TTA doubles the backbone runtime.

S2.2. Evaluation on Waymo

Here, we follow [6, 9, 15] to train on the Waymo [14] dataset, only using its front camera for the monocular 3D detection task. The Waymo dataset contains 798 training sequences and 202 validation sequences, each with around 200 images. The objects are split into two difficulty level: “Level_1” and “Level_2”, depending on the number of Lidar points within the object bounding box. For evaluation, we adopt the official metrics - 3D average precision AP_{3D} and 3D average precision weighted by heading APH_{3D}. Besides evaluating all objects together, we also evaluate for objects separately in different distance ranges, including 0-30m, 30-50m, and 50m-∞. We use the panoptic annotations in Waymo to extract instance masks, which are however provided only for a subset of images - as a result we train with 12128 images instead of 52386 as in [6, 9, 15]. Since DID-M3D [12] does not release the code for Waymo, we instead use DEVIANT [6] as our base 3D detector. As shown in Tab. S9, despite slightly lagging behind [6], our method overall outperforms other recent methods [9, 15] even without fusion. This indicates the promising potential of our method towards 3D detection in more diverse and complex environment. Empirically, we observe that the 3D object bounding box annotations in Waymo are often not tight. This may affect the learning on object size and further on the localization accuracy of NeurOCS, since the object size directly

	Mean Depth MAE			Median Depth MAE		
range (# of obj.)	0-20m (596)	20-40m (719)	>40m (16)	0-20m (596)	20-40m (719)	>40m (16)
DEVIANT [6]	0.757	**1.603**	**4.499**	0.660	1.270	**4.590**
NeurOCS-PnP	0.746	2.160	6.719	0.575	1.780	6.725
NeurOCS-Fusion	**0.602**	1.714	5.624	**0.460**	**1.180**	5.415
range (# of obj.)	0-20m (1273)	20-40m (2173)	>40m (858)	0-20m (1273)	20-40m (2173)	>40m (858)
DID-M3D [12]	0.665	**1.713**	**3.373**	0.510	1.400	**2.740**
NeurOCS-PnP	0.801	2.396	5.671	0.650	1.950	5.275
NeurOCS-Fusion	**0.637**	1.721	3.711	**0.480**	**1.380**	3.150

Table S10. **Cross-dataset evaluation on NuScenes.** The depth MAEs are reported for recalled objects in different ranges, each containing # objects as shown.

impacts the object distance in the PnP optimization. Further, the rolling shutter effect [16] in the camera used by Waymo may impact our geometric optimization which assumes the image is taken from a global shutter camera. We leave the handling of these challenges for performance improvements as the future work.

S2.3. Evaluation on NuScenes

While cross-dataset generalization is not the main goal of our work, we follow DEVIANT [6] to perform evaluation in NuScenes dataset [1] to understand its generalization capability. Specifically, we train on the training split of the KITTI training set and evaluate on all 6019 frontal images in the NuScenes validation set, using the mean absolute error (MAE) [6, 13] of the depth of the boxes. This is computed only on recalled objects (2D IoU>0.7) where the ground truth depth can be retrieved; strictly speaking, the 3D error measured by depth MAE from two methods are comparable only when using the same set of recalled objects. We first compare with DEVIANT as shown in Tab. S10, where we use DEVIANT as our base detector to have the same set of recalled objects. As can be seen, while DEVIANT achieves good performance due to its special design in depth-equivariant architecture, NeurOCS yields reasonable performance with PnP alone, and its fusion with DEVIANT leads to improvements in some cases especially for nearby objects. Similar observations are obtained when using DID-M3D as the base detector. These results indicate the promising potential of NeurOCS in generalization due to its underlying geometric principles, especially for the near field where the PnP solution is less sensitive to the error in object size prediction. Further explorations towards improving its generalization capability remain our future work.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 5

[2] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *CVPR*, 2021. 2, 3

[3] Jonas Heylen, Mark De Wolf, Bruno Dawagne, Marc Proesmans, Luc Van Gool, Wim Abbeloos, Hazem Abdelkawy, and Daniel Olmeda Reino. Monocinis: Camera independent monocular 3d object detection using instance segmentation. In *ICCV*, 2021. 4

[4] Yu Hong, Hang Dai, and Yong Ding. Cross-modality knowledge distillation network for monocular 3d object detection. In *ECCV*, 2022. 2, 3

[5] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *ICCV*, 2021. 3

[6] Abhinav Kumar, Garrick Brazil, Enrique Corona, Armin Parchami, and Xiaoming Liu. DEVIANT: Depth EquiVarIAnt NeTwork for Monocular 3D Object Detection. In *ECCV*, 2022. 1, 2, 3, 4, 5

[7] Zhuoling Li, Zhan Qu, Yang Zhou, Jianzhuang Liu, Haoqian Wang, and Lihui Jiang. Diversity matters: Fully exploiting depth clues for reliable monocular 3d object detection. In *CVPR*, 2022. 3

[8] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *ICCV*, 2019. 2

[9] Qing Lian, Peiliang Li, and Xiaozhi Chen. Monojsg: Joint semantic and geometric cost volume for monocular 3d object detection. In *CVPR*, 2022. 4

[10] Thomas Müller. tiny-cuda-nn, 4 2021. 1

[11] Liang Peng, Fei Liu, Zhengxu Yu, Senbo Yan, Dan Deng, Zheng Yang, Haifeng Liu, and Deng Cai. Lidar point cloud guided monocular 3d object detection. In *ECCV*, 2022. 2, 3

[12] Liang Peng, Xiaopei Wu, Zheng Yang, Haifeng Liu, and Deng Cai. Did-m3d: Decoupling instance depth for monocular 3d object detection. *arXiv preprint arXiv:2207.08531*, 2022. 2, 4, 5

[13] Xuepeng Shi, Qi Ye, Xiaozhi Chen, Chuangrong Chen, Zhixiang Chen, and Tae-Kyun Kim. Geometry-based distance decomposition for monocular 3d object detection. In *ICCV*, 2021. 5

[14] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 4

[15] Li Wang, Li Zhang, Yi Zhu, Zhi Zhang, Tong He, Mu Li, and Xiangyang Xue. Progressive coordinate transforms for monocular 3d object detection. *NeurIPS*, 2021. 4

[16] Bingbing Zhuang, Loong-Fah Cheong, and Gim Hee Lee. Rolling-shutter-aware differential sfm and image rectification. In *ICCV*, 2017. 5