

# An Instance-Centric Panoptic Occupancy Prediction Benchmark for Autonomous Driving — Supplementary Material —

Yi Feng<sup>1\*</sup>   Junwu E<sup>1\*</sup>   Zizhan Guo<sup>1</sup>   Yu Ma<sup>1</sup>   Hanli Wang<sup>1,2</sup>   Rui Fan<sup>1,2,3✉</sup>

<sup>1</sup>College of Electronic and Information Engineering, Tongji University

<sup>2</sup>Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University

<sup>3</sup>National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, Xi'an Jiaotong University

## 1. Additional Information of CarlaOcc

### 1.1. Data Collection

To construct a comprehensive dataset for evaluating autonomous driving perception models, we collected diverse and realistic driving sequences distributed across eight CARLA towns. Each sequence is recorded under a unique combination of town map and traffic configuration, ensuring wide coverage of road topologies, urban structures, and dynamic agent interactions.

**Traffic Level Configuration.** To simulate realistic and controllable driving environments, we defined four standardized traffic levels: no traffic, light traffic, medium traffic, and heavy traffic. The no-traffic setting contains no surrounding vehicles or pedestrians and is included specifically to support methods that require purely static scenes where only the ego vehicle is in motion. The remaining three levels introduce progressively denser and more complex driving environments by increasing the number of spawned vehicles and pedestrians while tightening their inter-vehicle spacing and increasing speed variability. These traffic configurations collectively shape the density, aggressiveness, and interaction frequency of dynamic agents, producing scenarios that range from free-driving conditions to highly congested urban environments.

**Sequence Distribution.** The collected 104 sequences are evenly distributed across 8 CARLA towns, with each town containing one sequence for no-traffic configuration, three sequences for low and medium traffic configuration, and five sequences for high traffic configuration. To ensure balanced spatial diversity, we use Farthest Point Sampling to initialize each sequence with a distinct starting position and driving route. This strategy remarkably prevents redundant scene coverage across sequences and maximizes the utilization of spatial diversity in each town.

**Sensor Configuration.** Our sensor setup follows the KITTI-360 [8] configuration for cameras 00-03 and the LiDAR, including their viewing directions, intrinsic parameters, relative transformations, and the LiDAR mounting height. However, the original KITTI-360 design presents several limitations for occupancy prediction. First, cameras 00 and 01 are tilted downward by  $5^\circ$ , causing the camera-frame occupancy to intersect the ground prematurely and disrupting the geometric consistency required for single-view occupancy learning. Second, these two cameras use fisheye lenses and require additional rectification to match the perspective cameras, which complicates the training process. Third, KITTI-360 lacks a rear-view camera, making full surround-view occupancy prediction infeasible.

To address these issues, we redesign the sensor suite by removing the camera tilt, using perspective cameras for all views, and adding rear-view cameras to achieve a complete  $360^\circ$  coverage. As illustrated in Fig. 1, cameras 00-05 form a symmetric surround-view setup, each providing RGB, semantic segmentation, and depth maps with a unified field of view of  $103.7^\circ$  and the same resolution of  $1,408 \times 376$  pixels. The 32-channel LiDAR operates with a maximum range of 80 m and a rotation frequency of 20 Hz, producing approximately 250,000 points per second. Its vertical field of view spans from  $-30^\circ$  to  $10^\circ$ , providing dense coverage of both near-ground geometry and elevated structures.

### 1.2. Skeletal Motion Analyzer

To enable precise reconstruction of deforming pedestrian meshes, we develop a *Skeletal Motion Analyzer* that aligns real-time skeletal motion observations with a pre-recorded canonical walking cycle.

**Canonical Gait Database.** We construct a standard gait database  $\mathcal{G} = \{\mathbf{g}_d\}_{d=0}^{D-1}$  containing  $D$  discrete phases of a complete walking cycle. Each entry  $\mathbf{g}_d$  stores the transformation matrices for key articulations and the extracted

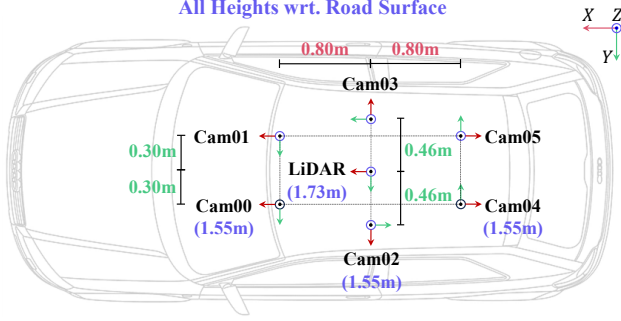


Figure 1. Top-down illustration of the sensor configuration in the CarlaOcc dataset.

kinematic descriptors. For each phase index  $d$ , we compute a motion characteristic descriptor

$$\delta_d = \begin{bmatrix} \Delta x^d \\ \theta_{\text{rel}}^d \end{bmatrix}, \quad (1)$$

where  $\Delta x^d$  encodes the forward displacement differential and  $\theta_{\text{rel}}^d$  denotes the relative articulation angle derived from joint rotations.

**Real-time Motion Analysis.** For each pedestrian instance  $k$  at frame  $t$ , we extract a local temporal window of  $W$  frames:

$$\mathcal{X}_k^t = \left\{ \mathbf{x}_k^{t-\frac{W}{2}}, \dots, \mathbf{x}_k^t, \dots, \mathbf{x}_k^{t+\frac{W}{2}} \right\}, \quad (2)$$

where each observation  $\mathbf{x}_k^t$  contains instantaneous kinematic features. To align the observed motion with canonical gait phases, we derive an observation descriptor for the central frame:

$$\hat{\delta}_k^t = \begin{bmatrix} \Delta \hat{x}_k^t \\ \hat{\theta}_{\text{rel},k}^t \end{bmatrix}, \quad (3)$$

where  $\Delta \hat{x}_k^t$  is estimated from the temporal window and  $\hat{\theta}_{\text{rel},k}^t$  is obtained from articulated joint rotations.

We then compute a phase likelihood score for each canonical phase  $d$  as

$$\mathcal{L}(d | \hat{\delta}_k^t) = \exp\left(-\frac{1}{2} \left\| \hat{\delta}_k^t - \delta_d \right\|_{\Sigma^{-1}}^2\right), \quad (4)$$

where  $\Sigma$  is a diagonal covariance matrix calibrated from training data. The matched gait phase is obtained via maximum-likelihood estimation:

$$d_k = \arg \max_{d \in [0, D-1]} \mathcal{L}(d | \hat{\delta}_k^t). \quad (5)$$

**Mesh Retrieval and Reconstruction.** Once the phase  $d_k$  is determined, the corresponding posed mesh  $\mathcal{M}_{d_k}$  is retrieved from the canonical gait database and transformed to the world coordinate system using the recorded global

transformation  $\mathbf{T}_k$ , yielding a geometrically accurate reconstruction of the pedestrian’s instantaneous pose. This phase matching approach achieves robust gait alignment without requiring explicit skeletal retargeting or expensive inverse kinematics, enabling efficient processing of large-scale pedestrian populations in simulation.

### 1.3. Sensor Artifacts Rectification

As discussed in Sec. 3.2 in the main paper, the multi-modal signals rendered in Unreal Engine 5 frequently exhibit systematic artifacts that violate geometric and semantic consistency. These issues mainly arise from the way that rendering pipeline handles transparent (*e.g.*, glass, windows) and non-Nanite<sup>1</sup> (*e.g.*, terrain) materials:

- Translucent materials are excluded from the depth buffer, which causes the rendered depth to correspond to the first opaque surface behind the transparent object.
- The semantic rendering pass, inherited from the legacy CARLA implementation, applies material substitution only to the Nanite rendering path.

As a result, transparent objects inherit the depth and semantics of background structures, and some non-Nanite instances disappear entirely from the semantic maps. These modality-specific failures manifest as broken silhouette alignment, incorrect depth ordering, and incomplete semantic regions, thereby corrupting the physical integrity of the dataset.

To address these issues, we introduce an instance-guided rectification method that reconstructs per-frame, physically consistent depth and semantic maps by using instance-level geometry priors, without the need of modifying the rendering engine.

**Depth Map Rectification.** We reconstruct per-camera, ray-casted geometry and fuse it with the sensor-collected depths in a physically consistent manner. For each frame, we first assemble a compact scene mesh  $\mathcal{M}$  containing only the geometries of transparent or semi-transparent objects, as well as those exhibiting semantic inconsistencies in the rendered maps. The remainder of the static environment is excluded to mitigate ray-casting overhead and to prevent artifacts caused by texture-dependent meshes (*e.g.*, vegetation), whose system-exported geometry is often less detailed than the sensor-collected signals. Subsequently, Then, by using the calibrated camera intrinsics and extrinsics, we create a tensor-accelerated pinhole ray bundle, where each ray originates from the optical center  $\mathbf{o}_{\text{cam}}$  and travels along the viewing direction  $\mathbf{d}(u, v)$ . The ray-casted depth can be obtained from the distance to the first visible surface intersected by each ray:

$$\mathbf{D}_{\text{rc}}(u, v) = \min_{t>0} \{ t \mid \mathbf{o}_{\text{cam}} + t \mathbf{d}(u, v) \in \mathcal{M} \}. \quad (6)$$

<sup>1</sup>Nanite is Unreal Engine 5’s virtualized micropolygon geometry system that enables efficient rendering of extremely high-detail meshes.

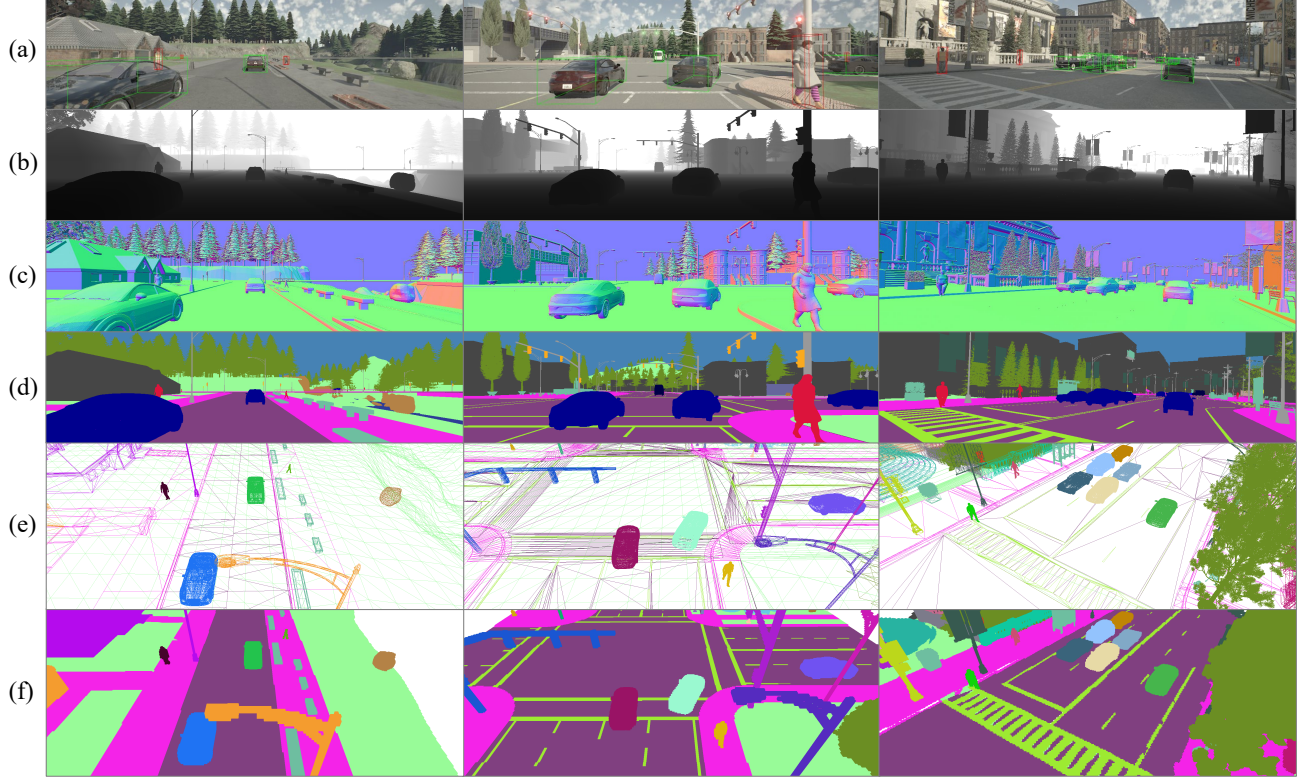


Figure 2. Visualization of multi-modal data in CarlaOcc: (a) RGB images with rich annotations, (b) depth maps, (c) surface normal maps, (d) semantic segmentation maps, (e) panoptic meshes shown in wireframe, and (f) panoptic occupancy ground truth. “Stuff” classes are visualized using the official color map, while “thing” instances are displayed in randomly assigned colors.

Subsequently, the raw depth map  $\mathbf{D}_{\text{raw}}$  is rectified by taking the pointwise minimum with the ray-casted result:

$$\hat{\mathbf{D}}(u, v) = \min(\mathbf{D}_{\text{raw}}(u, v), \mathbf{D}_{\text{rc}}(u, v)), \quad (7)$$

which effectively restores the correct surface geometry for transparent and semi-transparent regions that were incorrectly assigned to the background, while preserving valid measurements elsewhere. This depth fusion strategy ensures that each pixel records the nearest physically consistent surface along its viewing ray, thereby improving geometric fidelity and cross-modal consistency.

**Semantic Map Rectification.** Unlike depth rectification which enforces geometric consistency through the nearest-surface fusion, semantic rectification focuses on restoring categorical correctness and instance completeness. Given the mesh subset  $\mathcal{M}$  described above, we first generate a ray-casted semantic map  $\mathbf{S}_{\text{rc}}$  by querying, for each camera ray, the semantic label of the first intersected triangle within  $\mathcal{M}$ . This produces physically grounded labels for all transparent or inconsistent regions, where the collected semantic map  $\mathbf{S}_{\text{raw}}$  often misattributes the semantics of the background objects.

To identify the degraded regions, we exploit the depth consistency between the previously rectified depth  $\hat{\mathbf{D}}$ , the

collected depth  $\mathbf{D}_{\text{raw}}$ , and the ray-casted depth  $\mathbf{D}_{\text{rc}}$ . Pixels with significant depth discrepancies indicate that the collected semantics are associated with surfaces behind transparent materials rather than the true foreground geometry. Therefore, we define the transparency region  $\Omega_t$  as follows:

$$\Omega_t(u, v) = \mathbb{1}[\mathbf{D}_{\text{raw}}(u, v) - \hat{\mathbf{D}}(u, v) \geq \varepsilon], \quad (8)$$

where  $\mathbb{1}[\cdot]$  denotes the indicator function, and  $\varepsilon$  is a small positive threshold that distinguishes meaningful depth differences from sensor noise. This condition captures pixels where the collected depth erroneously penetrates through transparent surfaces to record background geometry. In addition to transparency-induced mislabels, certain instances disappear completely in  $\mathbf{S}_{\text{raw}}$  due to missing or invalid semantic substitution during rendering. We detect such omission region  $\Omega_s$  by finding unlabeled pixels in the collected map:

$$\Omega_s(u, v) = \mathbb{1}[\mathbf{S}_{\text{raw}}(u, v) = s_{\emptyset}], \quad (9)$$

where  $s_{\emptyset}$  denotes the empty label assigned to unrendered instances. These regions typically correspond to non-Nanite assets that are excluded from the semantic rendering pipeline, creating semantic voids in otherwise geometrically

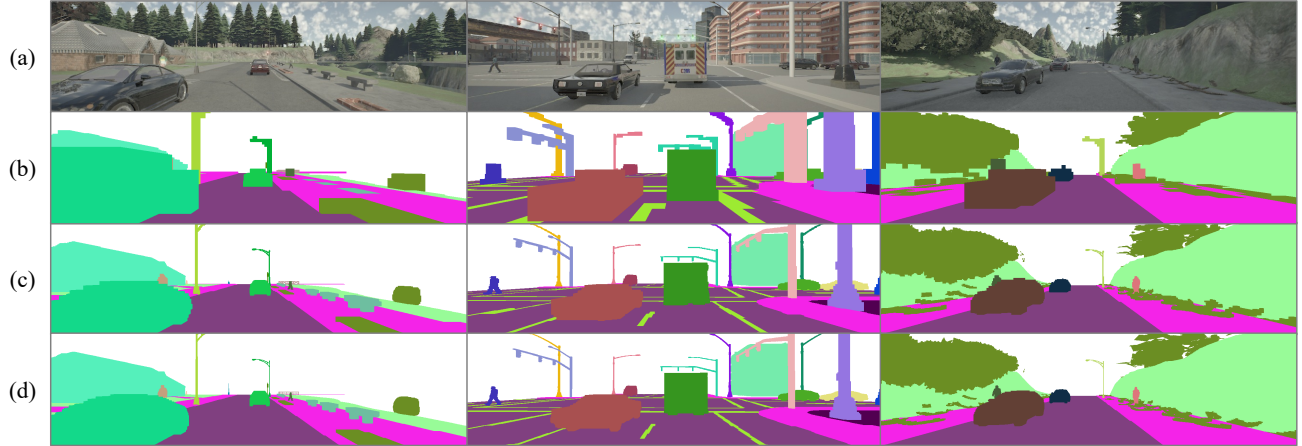


Figure 3. Visualization of occupancy ground truth at different voxel resolutions: (a) RGB images, (b) panoptic occupancy labels at 0.5 m voxel size, (c) 0.1 m voxel size, and (d) 0.05 m voxel size.

complete areas. The complete error region  $\Omega_{\text{err}} = \Omega_t \cup \Omega_s$  for semantic rectification is then defined as the union of transparency-induced mislabels and omission artifacts, and the final rectified semantic map  $\hat{\mathbf{S}}$  is obtained by strategically replacing semantics in these error regions:

$$\hat{\mathbf{S}}(u, v) = \begin{cases} \mathbf{S}_{\text{rc}}(u, v), & (u, v) \in \Omega_{\text{err}}, \\ \mathbf{S}_{\text{raw}}(u, v), & \text{otherwise.} \end{cases} \quad (10)$$

This unified rectification framework simultaneously addresses both transparency-induced semantic misattribution and instance omission artifacts, ensuring that all visible surfaces receive physically consistent and categorically accurate depth measurements and semantic labels. The approach effectively bridges the gap between the detailed geometry captured by the depth sensor and the categorical information provided by the semantic renderer, producing physically consistent depth and semantic outputs suitable for training robust perception systems.

#### 1.4. Data Visualization.

To provide an intuitive understanding of the richness and cross-modal consistency of CarlaOcc, we visualize various samples of the dataset. The dataset offers a diverse set of complementary modalities rendered from the same calibrated multi-camera rig, enabling holistic scene perception. As shown in Fig. 2, each frame includes high-fidelity RGB images, metrically accurate depth maps, dense surface normals, semantic segmentation maps, instance-level panoptic meshes, and the resulting panoptic occupancy ground truth. These visualizations highlight the geometric completeness of our reconstructed scenes and the fine-grained, instance-level annotations preserved across modalities.

In Fig. 3, we further demonstrate the panoptic occupancy ground truth generated at different voxel resolutions,

ranging from coarse (0.5 m) to fine-grained (0.05 m) grids. While coarser voxels provide a compact global overview of the 3D scene layout, reducing memory and computation costs, higher-resolution grids accurately capture object silhouettes, scene boundaries, and small structures such as pedestrians or traffic signs. This multi-resolution supervision enables researchers to adapt model capacity and training objectives to different computational budgets and application scenarios. Together, these visualizations illustrate the high geometric fidelity and scalable annotation quality that CarlaOcc offers for 3D occupancy perception research.

## 2. Additional Experimental Results

### 2.1. Occupancy Dataset Quality Comparison

We compare the quality of occupancy ground truth produced by existing datasets and our proposed CarlaOcc dataset. Traditional pipelines such as KITTI-360-SSCBench [7] and Occ3D-nuScenes [13] construct occupancy labels by directly voxelizing sparse LiDAR point clouds. Since LiDAR measurements are inherently incomplete and noisy, this strategy inevitably leads to broken surfaces, isolated floating points, and large holes in unobserved or distant regions. As shown in Fig. 4(a) and (b), buildings, sidewalks, and road boundaries are clearly fragmented, and many regions that should be occupied are missing, which results in suboptimal and misleading supervisory signals for learning 3D environment layout.

Synthetic datasets like CarlaSC [16] adopt a similar paradigm even though they are built in simulation. They still rely on multiple LiDAR sensors to scan the environment and then discretize the resulting point clouds into voxels. Consequently, the final occupancy labels are still limited by LiDAR sampling density, occlusions, and sensor noise, rather than reflecting the complete underlying scene



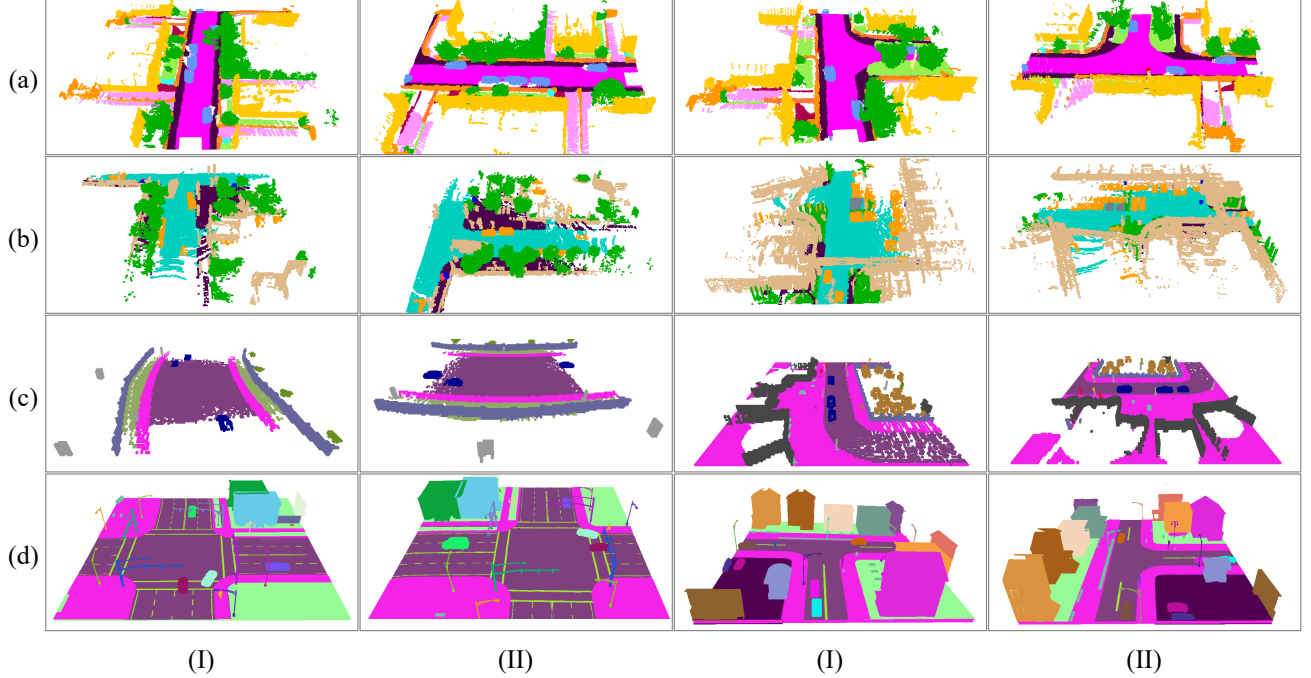


Figure 4. Ground truth quality comparison between existing public occupancy datasets and CarlaOcc: (a) KITTI-360-SSCBench [7], (b) Occ3D-nuScenes [13], (c) CarlaSC [16], and (d) our proposed CarlaOcc dataset. For each dataset, we present the (I) forward view and the (II) left-side view for better comparison.

geometry. This can be observed in Fig. 4(c), where road surfaces and curbs remain discontinuous and the geometry of surrounding structures is coarse and irregular.

In contrast, our CarlaOcc dataset bypasses LiDAR sampling and directly accesses the high-fidelity 3D mesh assets within the simulator. We retrieve the full polygonal meshes of all relevant scene elements, and then perform voxelization and precise cropping in the world coordinate system to obtain dense and consistent occupancy labels. As illustrated in Fig. 4(d), this mesh-based pipeline produces smooth surfaces, clean object boundaries, and almost no noisy or missing regions in both the forward and left-side views. Overall, CarlaOcc delivers significantly higher-quality occupancy ground truth than prior LiDAR-based pipelines, providing a much stronger supervisory signal for training 3D occupancy prediction models.

## 2.2. City-Level Scene Reconstruction

We merge all exported static meshes of each town into a single unified mesh with their corresponding transformation matrices, yielding a full city-level reconstruction of the CARLA maps. As shown in Fig. 5, the resulting reconstructed towns accurately preserve roads, buildings, terrain, and large-scale layout. The left column visualizes the unified static mesh produced by our pipeline, while the right column shows the corresponding RGB rendering in Un-

real Engine 5, confirming that our reconstruction precisely matches the original simulator scenes.

## 2.3. Parametric Scene Generation

We further demonstrate that the proposed ADMesh representation naturally supports parametric generation of high-quality dynamic scenes. Starting from the static mesh library in ADMesh (buildings, roads, vehicles, *etc.*), we describe each scene by a compact set of actor-level parameters: (i) the index of the mesh asset, (ii) a rigid transform in the world coordinate system, and (iii) an optional time-dependent trajectory function that specifies how the actor moves over time. Given a set of trajectories and initial placements that respect the road layout, we can instantiate all actors at each simulation step  $t_k$ , assign the corresponding transform to their meshes, and render the resulting frame from a fixed virtual camera. By advancing  $t_k$  we obtain a temporally coherent, physically plausible dynamic sequence, while the underlying geometry always comes from high-quality meshes in ADMesh.

To evaluate the flexibility of this parametric representation, we construct two types of sequences. As shown in Fig. 6, we fix a configuration of buildings and vehicles and simulate their trajectories over several time steps, yielding a dynamic urban scene in which cars move along the road while the camera remains static. Second, we keep all tra-

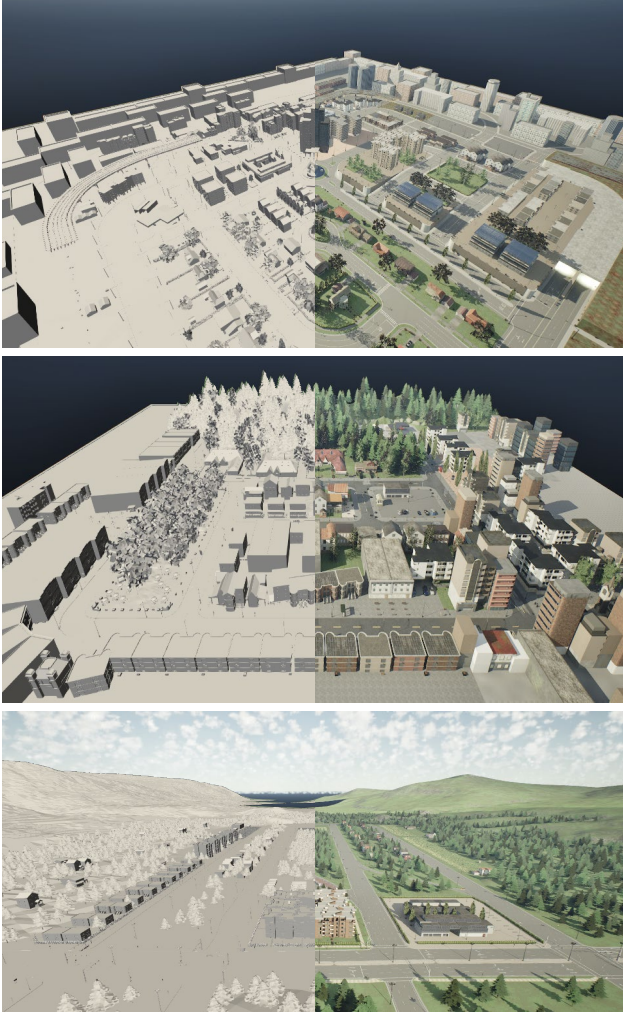


Figure 5. Visualization of the reconstructed CARLA maps. Left: unified static scene meshes obtained through our reconstruction pipeline. Right: corresponding RGB images captured in Unreal Engine 5.

jectories and camera parameters unchanged, but replace the mesh asset assigned to each actor (*e.g.*, replacing building or vehicle models for another with a compatible footprint). This simple mesh substitution produces a new dynamic scene with different appearance and layout, while preserving temporal consistency and avoiding any manual re-authoring of trajectories. The experiment indicates that ADMesh enables efficient scene editing and instance replacement at the level of individual actors, making it suitable for large-scale generation of diverse driving scenarios.

## 2.4. 3D Occupancy Prediction

**Ablation on Voxel Size.** To analyze the impact of voxel resolution on 3D semantic occupancy prediction, we conduct an ablation study on the CarlaOcc dataset by vary-

Table 1. Ablation study on voxel size for semantic occupancy prediction on the CarlaOcc dataset.

Method	Voxel size (m)	IoU	mIoU
Symphonies [5]	0.05	15.2	10.1
	0.10	21.5	12.1
	0.50	32.6	15.9
GaussianFormer2 [4]	0.05	18.4	10.9
	0.10	23.1	14.8
	0.50	38.3	20.7

ing the voxel size from 0.05 m and 0.10 m to 0.50 m for Symphonies [5] and GaussianFormer2 [4]. As shown in Table 1, the coarsest voxel size (0.50 m) achieves the best IoU and mIoU for both methods, while the performance decreases when using finer voxels. This is mainly because CarlaOcc provides high-resolution ground-truth occupancy, where even small localization errors lead to noticeable misalignment with the dense labels at finer voxel resolutions, making IoU and mIoU much more sensitive and the task more challenging.

**Per-Class Results on CarlaOcc.** In the main paper, we report only the overall IoU and mIoU on the CarlaOcc benchmark. For completeness, Table 3 further provides the per-class IoU for all semantic categories. The distribution of scores is consistent with the overall metrics: road, building and vegetation related classes are generally easier, while small and thin objects such as poles, traffic signs and traffic lights remain more challenging. These results offer a detailed per-class benchmark for CarlaOcc and can serve as a reference for future monocular 3D semantic occupancy methods evaluated on this dataset.

## 2.5. Sim-to-Real Experiments

To validate the practical utility of CarlaOcc, we conduct *Sim-to-Real Transfer* experiments by pretraining models on CarlaOcc and finetuning on real-world datasets. To bridge the domain gap, we map semantic labels and resample CarlaOcc occupancy GT to match the target dataset’s semantic classes, voxel resolution, and spatial origin. As shown in Table 4, **pretraining on CarlaOcc consistently improves mIoU by 0.8%-1.5% compared to training from scratch**. While seemingly modest, these gains are mainly constrained by the sparsity and noise of real-world GT (Fig. 7, left), and by the configuration discrepancy, where KITTI-360 benefits more due to its closer setup to CarlaOcc. Qualitatively (Fig. 7), CarlaOcc pretraining yields more complete instance contours and improved occlusion handling, proving that CarlaOcc helps the model learn robust 3D spatial reasoning.

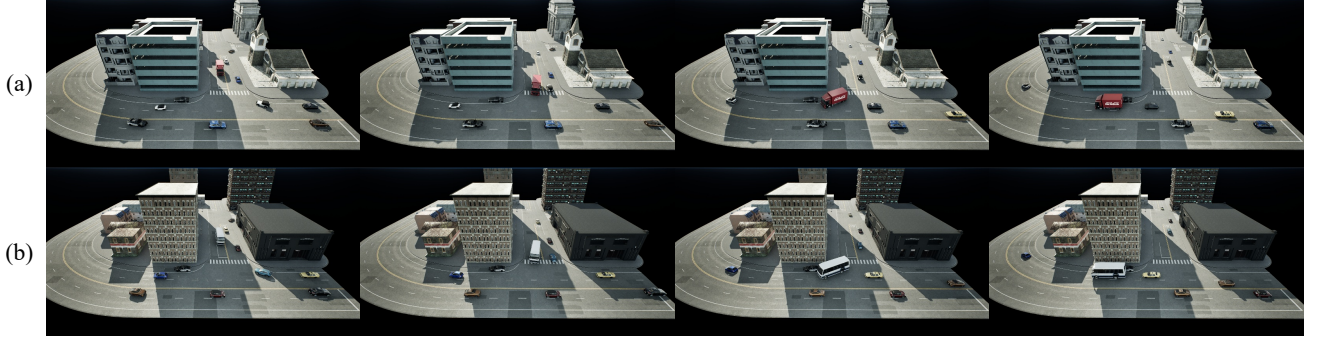


Figure 6. Parametric scene generation and editing on ADMesh. Each row shows a dynamic sequence rendered at four successive time steps. (a) Original scene generated by assigning trajectories and poses to the static mesh actors. (b) Edited scene obtained by replacing the mesh asset of each actor while keeping all trajectories, camera parameters, and illumination fixed. The example illustrates that our parametric representation can generate diverse, high-quality dynamic urban scenes with simple actor-level edits.

Table 2. Depth estimation results on the CarlaOcc dataset.

Category	Method	AbsRel ↓	SqRel ↓	RMSE ↓	RMSElog ↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
Discriminative	MiDaS [1]	0.128	1.642	7.215	0.335	0.842	0.948	0.977
	LeReS [18]	0.112	1.392	6.932	0.314	0.868	0.957	0.980
	DPT [12]	0.101	1.248	6.621	0.302	0.887	0.964	0.982
	DepthAnythingV2 [17]	0.093	1.035	6.159	0.297	0.899	0.965	0.983
Generative	Marigold [6]	0.089	1.012	6.045	0.284	0.904	0.968	0.985
	LOTUS [2]	0.085	0.985	5.918	0.276	0.911	0.972	0.987
Metric	Metric3Dv2 [3]	0.087	1.111	6.427	0.199	0.899	0.959	0.975
	UniDepthV2 [11]	0.082	0.942	5.731	0.219	0.928	0.975	0.989

Table 3. Semantic occupancy prediction results on the CarlaOcc dataset.

Method																	
	IoU	mIoU	Road	Sidewalk	Building	Wall	Fence	Pole	TrafficLight	TrafficSign	Vegetation	Ground	Person	Car	Truck	OtherVehicle	Other
Symphonies [5]	32.6	15.9	36.74	20.04	21.38	6.68	10.02	5.34	6.68	6.01	26.72	16.70	5.34	30.06	23.38	16.70	6.71
OccFormer [19]	30.1	14.4	33.28	18.15	19.36	6.05	9.08	4.84	6.05	5.45	24.20	15.13	4.84	27.23	21.18	15.13	6.03
OPUS [14]	37.0	19.3	44.60	24.33	25.95	8.11	12.16	6.49	8.11	7.30	32.44	20.27	6.49	36.49	28.38	20.27	8.11
GaussianFormer2 [4]	38.3	20.7	47.84	26.09	27.83	8.70	13.05	6.96	8.70	7.83	34.79	21.74	6.96	39.14	30.44	21.74	8.69

## 2.6. Other 3D Perception tasks on CarlaOcc

**Monocular Depth Estimation.** Table 2 reports monocular depth estimation results on the CarlaOcc benchmark. We evaluate a representative set of state-of-the-art methods that cover three major paradigms: *discriminative* models, *generative* models, and *metric* depth estimators. All methods take a single RGB image as input and are evaluated using standard depth metrics: absolute relative error (AbsRel), squared relative error (SqRel), root mean squared error (RMSE), log RMSE (RMSElog), and threshold accu-

racies [1]. Among discriminative approaches, DepthAnythingV2 [17] achieves the best overall performance, substantially improving over earlier networks such as MiDaS [1] and LeReS [18] in both error and accuracy metrics. Generative methods such as Marigold [6] and LOTUS [2] further reduce the AbsRel error, and the metric depth models (e.g., Metric3Dv2 [3], UniDepthV2 [11]) produce competitive or superior results while predicting depth in an absolute scale. These results indicate that CarlaOcc constitutes a challenging yet well-structured benchmark for advanced monocular depth estimators and can serve as a



Table 4. Sim-to-real evaluation results of representative baselines pretrained on CarlaOcc and finetuned on real-world datasets.

Pretrain	Target	Method	Scratch mIoU	mIoU
CarlaOcc	KITTI-360	Symphonies	15.9	<b>17.4</b>
CarlaOcc	nuScenes	SparseOcc	16.5	<b>17.3</b>

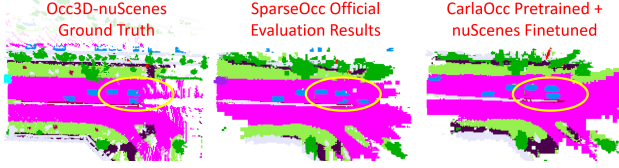


Figure 7. Visualizations of sim-to-real experiment. As shown in yellow circles, SparseOcc exhibits more complete instance contours after pretraining on CarlaOcc compared to official results.

Table 5. Representative 3D object detection baselines on the CarlaOcc dataset.

Method	mAP $\uparrow$	mATE $\downarrow$	mASE $\downarrow$	MAOE $\downarrow$
DETR3D [15]	0.39	0.63	0.26	0.37
PETR [10]	0.45	0.57	0.27	0.39
Sparse4D [9]	0.48	0.66	0.24	0.33

strong basis for downstream 3D perception tasks.

**3D Object Detection.** Representative camera-only 3D object detection baselines are evaluated on our CarlaOcc benchmark. We follow the nuScenes evaluation protocol and report mean Average Precision (mAP) together with the standard error metrics: translation error (mATE), scale error (mASE), and orientation error (MAOE), where higher mAP and lower errors indicate better performance. We include three widely used multi-view detectors: DETR3D [15], PETR [10], and Sparse4D [9]. As shown in Table 5, performance steadily improves from DETR3D to PETR and further to Sparse4D, with Sparse4D achieving the highest mAP among the tested methods. These baselines demonstrate that our dataset is compatible with state-of-the-art 3D detectors and can serve as a standard testbed for future research on camera-based 3D object detection.

## References

- [1] Reiner Birkel et al. MiDaS v3.1 – A Model Zoo for Robust Monocular Relative Depth Estimation. *arXiv preprint arXiv:2307.14460*, 2023. 7
- [2] Jing He et al. Lotus: Diffusion-Based Visual Foundation Model for High-Quality Dense Prediction. *arXiv preprint arXiv:2409.18124*, 2024. 7
- [3] Mu Hu et al. Metric3D v2: A Versatile Monocular Geometric Foundation Model for Zero-Shot Metric Depth and Surface Normal Estimation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024. 7
- [4] Yuanhui Huang et al. GaussianFormer-2: Probabilistic Gaussian Superposition for Efficient 3D Occupancy Prediction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 27477–27486, 2025. 6, 7
- [5] Haoyi Jiang et al. Symphonize 3D Semantic Scene Completion with Contextual Instance Queries. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 20258–20267, 2024. 6, 7
- [6] Bingxin Ke et al. Repurposing Diffusion-Based Image Generators for Monocular Depth Estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9492–9502, 2024. 7
- [7] Yiming Li et al. Ssbench: A Large-Scale 3D Semantic Scene Completion Benchmark for Autonomous Driving. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13333–13340, 2024. 4, 5
- [8] Yiyi Liao et al. KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(3):3292–3310, 2023. 1
- [9] Shaoyu Lin et al. Sparse4D: Multi-view 3D Object Detection with Sparse Spatial-Temporal Fusion. In *Eur. Conf. Comput. Vis.*, pages 87–104, 2024. 8
- [10] Yingfei Liu et al. PETR: Position Embedding Transformation for Multi-View 3D Object Detection. In *Eur. Conf. Comput. Vis.*, pages 531–548, 2022. 8
- [11] Luigi Piccinelli et al. UniDepthV2: Universal Monocular Metric Depth Estimation Made Simpler. *arXiv preprint arXiv:2502.20110*, 2025. 7
- [12] René Ranftl et al. Vision Transformers for Dense Prediction. In *Int. Conf. Comput. Vis.*, pages 12179–12188, 2021. 7
- [13] Xiaoyu Tian et al. Occ3D: A Large-Scale 3D Occupancy Prediction Benchmark for Autonomous Driving. In *Adv. Neural Inform. Process. Syst.*, pages 64318–64330, 2023. 4, 5
- [14] Jiabao Wang et al. OPUS: Occupancy Prediction Using a Sparse Set. *arXiv preprint arXiv:2409.09350*, 2024. 7
- [15] Yue Wang et al. DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In *Conference on Robot Learning*, pages 180–191, 2021. 8
- [16] Joey Wilson et al. MotionSC: Data Set and Network for Real-Time Semantic Mapping in Dynamic Environments. *IEEE Robotics and Automation Letters*, 7(3):8439–8446, 2022. 4, 5
- [17] Lihe Yang et al. Depth Anything v2. *Adv. Neural Inform. Process. Syst.*, 37:21875–21911, 2024. 7
- [18] Wei Yin et al. Learning To Recover 3D Scene Shape From a Single Image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 204–213, 2021. 7
- [19] Yunpeng Zhang et al. OccFormer: Dual-path Transformer for Vision-based 3D Semantic Occupancy Prediction. In *Int. Conf. Comput. Vis.*, pages 9433–9443, 2023. 7