OccGen: Generative Multi-modal 3D Occupancy Prediction for Autonomous Driving — Supplementary Material —

Guoqing Wang[1], Zhongdao Wang[2], Pin Tang[1], Jilai Zheng[1], Xiangxuan Ren[1], Bailan Feng[2], and Chao Ma[1⋆]

[1]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University,
[2]Huawei Noah's Ark Lab
{guoqing.wang,pin.tang,zhengjilai,bunny_renxiangxuan,chaoma}@sjtu.edu.cn
{wangzhongdao,fengbailan}@huawei.com

In the supplementary material, we first present the methodology details of our proposed OccGen, including hard 2D-to-3D view transformation, geometry mask, discriminative vs. generative modeling, and DDPM and DDIM. Then, we provide the details of datasets and implementation. Furthermore, we present additional experimental results to demonstrate the effectiveness of OccGen. Finally, we discuss the broader impact statement and limitations.

A Methodology Details

A.1 Hard 2D-to-3D View Transformation

The previous LSS-based methods [15, 17, 19] associate a set of discrete depths for every pixel, covering the full range of potential depth values. These methods typically choose the softmax operation, which is a smooth approximation of *argmax*, which normalizes the vector, enabling gradient computation while the values can also represent probabilities. Intuitively, this soft approach for depth prediction allows the network to learn depth information that is more conducive to feature optimization, rather than more precise depth information. As per the findings in [13], this soft depth prediction fails to obtain precise depth information, consequently leading to the presence of ambiguous grids in the camera voxel. In light of this, we propose a hard 2D-to-3D view transformation method that utilizes hard Gumbel-Softmax [8] to obtain a deterministic discrete output vector. The key point lies in the fact that the *argmax* operation for obtaining a one-hot vector is non-differentiable, making it impossible to calculate gradients, and consequently, network updates cannot be performed. Hard Gumbel-Softmax is a deterministic version of Gumbel-Softmax, where instead of sampling from the Gumbel-Softmax distribution. The formula for hard Gumbel-Softmax can be expressed as follows:

$$\mathbf{hard\_gumbel\_softmax}(z) = \mathbf{one\_hot}(\mathbf{argmax}((z+g)/\tau)) \tag{1}$$

⋆ Corresponding author.

where z represents the logits. g is sampled from the Gumbel distribution, typically calculated as $-log(-log(u))$ with u being a uniform random variable. τ is the temperature parameter controlling the softness of the distribution. When τ approaches zero, the hard Gumbel-Softmax approaches the one-hot encoding. The introduced hard 2D-to-3D view transformation enables gradient backpropagation during training and yields a definitive assignment of discrete depth during inference.

A.2 Geometry Mask

The camera voxel features F_c obtained through the hard 2D-to-3D view transformation module contain misleading information, leading to a blurred feature distribution. This issue arises because all points along a camera ray in 3D space are projected to the same location on the 2D image plane, resulting in some voxels sharing the same image features. This inaccurate 3D spatial structure hinders subsequent feature fusion and adversely impacts the final occupancy prediction. To overcome this challenge, we propose a straightforward method to exploit the geometry-aware correspondence between camera and LiDAR modalities. We generate a geometry mask by leveraging the LiDAR voxel features and applying it to the camera voxel features. This process effectively bridges the gaps between the image voxel representation and the true spatial distribution, resulting in improved fusion representation.

Due to the limitations of LiDAR sensors, the raw point cloud data only covers a portion of the real scene, leading to incomplete object shapes. When the point cloud is regularized into voxel grids, only a small portion of these voxels contain non-empty information. As a result, the initial sparse voxels fail to adequately represent the 3D geometry-aware correspondence between LiDAR and camera features. Through performing several 3D sparse convolutional operations, non-empty voxels are significantly increased, effectively covering more 3D space. Compared with the original LiDAR features, most of the 3D voxels are completed, and the generated 3D geometry-aware constraints can better reflect the real scene distribution.

Specifically, we can generate the geometry mask as follows,

$$L_{mask} = f_{\text{dense}}(f_{\text{sp}}(F_{\text{p}})) \tag{2}$$

where f_{sp} denotes a 3D sparse convolution block that mixes both regular and submanifold convolution and f_{dense} indicates that sparse camera voxel features are densified by padding zeros in empty positions. However, there is no guarantee that all foreground objects can be adequately represented even after the expansion of sparse features. Directly utilizing the aforementioned constraints on the camera voxel grids can obtain numerous all-zero features, potentially losing meaningful features. Therefore, we employ a softmax operation along the height dimension to maintain the density of camera voxel features. Subsequently, we apply the 3D constraint to the image voxel feature, which can be represented as,

$$F_c = \textbf{Softmax}(L_{mask}) \cdot F_c. \tag{3}$$

This weight assignment reduces the influence of misleading or ambiguous features during the transformation, effectively guaranteeing the robustness of the spatial information.

A.3 Discriminative vs. Generative Modeling

Discriminative Modeling. Discriminative methods for 3D occupancy semantic prediction aim to predict the occupancy and semantic labels of voxels in a 3D scene. These methods typically focus on learning the conditional distribution $P(Y|X)$, where Y represents the occupancy and semantic labels of voxels, and X represents the observed 3D scene data (e.g., point clouds or multi-view images). However, only learning these mapping between inputs and outputs may result in a limited understanding of the overall scene context. This can lead to incomplete or inaccurate scene completions, especially in complex scenes with intricate spatial relationships between objects. From the perspective of uncertainty, discriminative methods often do not explicitly model uncertainty in predictions, which can be crucial for 3D occupancy prediction where the inputs may be noisy or incomplete. This can lead to overconfident predictions in uncertain regions of the scene. Furthermore, these methods may struggle to incorporate prior knowledge or constraints into the learning process, which can be important for ensuring the coherence and consistency of the completed scene.

Generative Modeling. Generative modeling for 3D occupancy semantic prediction aims to generate complete 3D scenes from partial or incomplete input data. These methods often leverage generative models, such as GAN [5], VAE [10] and diffusion model [7, 20], to predict the semantic occupancy. Inspired by the great success of diffusion models, we adopt the diffusion model as our pipeline. Generative models learn a prior distribution of the scene occupancy, resulting in smoother modeling and the ability to generate results conforming to the ground truth distribution. Generative models outperform discriminative models in filling in blanks in the scene, showing stronger spatial imagination, and achieving better mIoU. More accurate local details of occupancy are essential for fine-grained scene perception and enhance the capability of subsequent planning and control. In addition, generative models inherently capture uncertainty in the predictions. This can be useful in scenarios where the inputs are noisy or ambiguous, as the model can provide a measure of confidence in its predictions.

A.4 DDPM and DDIM

A significant limitation of DDPM [7] is their requirement for numerous iterations to generate high-quality samples. This is due to the generative process, which transforms noise into data, approximating the reverse of the forward diffusion process. The forward diffusion process may involve thousands of steps, necessitating iteration over all these steps to produce a single sample. This process is much slower compared to GANs, which only require one pass through a network.

DDIM [20] enables significantly faster sampling without compromising on the training objective, making generative models using this architecture competitive with GAN of the same model size and sample quality. This is achieved by estimating the cumulative effect of multiple Markov chain steps and incorporating them simultaneously. Since each Markov jump is modeled as a Gaussian distribution, they approximate the combined effect of multiple jumps by using a higher-variance Gaussian distribution with the same mean. It is worth noting that the sum of two Gaussians remains Gaussian. In this paper, we utilize DDPM and DDIM to corrupt the ground truth occupancy and progressively refine the noise map to obtain the final results. The results on DDIM and DDPM are listed in the experimental part.

B Dataset and Implementation

B.1 Details of Dataset.

We provide results on nuScenes-Occupancy [23], Occ3D-nuScenes [22] and SemanticKITTI [1]. Occ3D-nuScenes and nuScenes-Occupancy are extended from the large-scale nuScenes [2] dataset with dense semantic occupancy annotation. SemanticKITTI [1] provides dense semantic annotations for each LiDAR sweep from the KITTI Odometry Benchmark [4]. We will introduce nuScenes-Occupancy [23], Occ3D-nuScnes [22] and SemanticKITTI [1] in sequence.

nuScenes-Occupancy. In the pursuit of establishing a large-scale surrounding occupancy perception dataset, wang [23] et al. introduced the nuScenes-Occupancy based on nuScenes [2]. Notably, the nuScenes-Occupancy dataset boasts approximately 40 times more annotated scenes and about 5 times more annotated frames compared to the work presented in [21]. To efficiently achieve this extensive annotation and densification of occupancy labels, they introduced an Augmenting And Purifying (AAP) pipeline. The pipeline initiates annotation through the superimposition of multi-frame LiDAR points. Acknowledging the sparsity inherent in the initial annotation attributed to occlusion or limitations in LiDAR channels, they employed a strategy to augment it with pseudo-occupancy labels. These pseudo labels are constructed using a pre-trained baseline. To further enhance the quality of the annotations by reducing noise and artifacts, human efforts are enlisted in the purification process.

Occ3D-nuScenes. Occ3D-nuScenes [22] is a comprehensive autonomous driving dataset comprising 700 training scenes and 150 validation scenes. Each frame in this dataset contains a 32-beam LiDAR point cloud and six RGB images captured by six cameras positioned at different angles around the LiDAR. These frames are densely annotated with voxel-wise semantic occupancy labels. The dataset's occupancy scope spans from $-40m$ to $40m$ along the X and Y axes, and from $-1m$ to $5.4m$ along the Z axis in the ego coordinate system. The voxel size for the occupancy labels is $0.4m \times 0.4m \times 0.4m$. Semantic labels in the dataset encompass 17 categories, which include 16 known object classes and an additional "empty" class.

SemanticKITTI. The SemanticKITTI dataset [1] is focused on semantic scene understanding using LiDAR points and front cameras. OccGen is evaluated for semantic scene completion using the monocular left camera as input, following MonoScene [3] and OccFormer [27]. In this evaluation, the ground truth semantic occupancy is represented as $256\times256\times32$ voxel grids. Each voxel is $0.2m\times0.2m\times0.2m$ in size and is annotated with one of 21 semantic classes (19 semantics, 1 free, 1 unknown). Similar to previous work [3, 12, 27], the dataset's 22 sequences are split into 10/1/11 for training/validation/testing.

Algorithm 1 Training algorithm

Input: Multi-modal inputs: $\{X_p, X_c\}$; GT occupancy: Y;
Output: Training loss

Extract multi-modal features F_p and F_c. $F_p, F_c \leftarrow$ **Extractor**(X_p, X_c)
Aggregate the camera features with a geometry mask. $F_c \leftarrow$ **Aggregate**(F_c, M_p)
Obtain the multi-modal fusion features F_m. $F_m \leftarrow$ **Fuser**(F_p, F_c);
Encoding the ground truth occupancy. $Y_0 \leftarrow$ **Encoding**(Y)
Construct noise signal and choose step index. $t \leftarrow$ **Randint**$(0, T)$, $\epsilon \leftarrow$ **Randn**(mean=0, std=1)
Signal scaling. $Y_0 \leftarrow$ **Norm**(Y_0)
Corrupt the occupancy input. $Y_t \leftarrow$ **Schedule**$(t) \times Y_0 + (1 -$ **Schedule**$(t)) \times \epsilon$
Obtain the downsampled multi-scale noise maps. $Y_t^i \leftarrow$ **Downsample**(Y_t)
Obtain the refined noise map. $Y_t \leftarrow$ **Refine**(F_m, Y_{t+1}, t)
Predict the occupancy results. $\hat{Y}_t \leftarrow$ **Voxel2Occ**(Y_t)
Calculate the training loss $\mathcal{L}_{\text{total}}$ (Eq. 9).

B.2 Implementation Details

In the camera stream, we adopt the ResNet-50 [6] model as our image backbone and employ the FPN [16] for multi-scale camera feature fusion, generating the image feature maps of size $6 \times 56 \times 100$, with 512 channels. Then, we utilize the proposed hard 2D-to-3D image view transformation to generate the camera voxel feature of size $128 \times 128 \times 10$, with 80 channels. In the LiDAR stream, the point cloud is constrained within the range of [-$51.2m, 51.2m$]for X and Y axis, and [-$5m, 3m$] for the Z axis. Voxelization is performed with a voxel size of $(0.1m, 0.1m, 0.1m)$. We utilize VoxelNet [28] as the backbone and employ sparse convolution [25] to produce the LiDAR voxel features of size $128\times128\times10$, with 80 channels. To fully exploit the implicit geometry-aware cues between camera and LiDAR modalities, we utilize the structure knowledge in LiDAR modality to guide the camera modality to learn geometry mask $80 \times 128 \times 128 \times 10$, which can improve the generalization capability of the fused voxel features significantly. Subsequently, we follow [23] and utilize ResNet3D and FPN-3D [16] to generate multi-scale voxel features as condition input for the progressive refinement module. Finally, we obtain the 3D semantic occupancy by feeding the refined voxel features to the occupancy head [23] for full-scale evaluation.

For the input, we follow the setting in [23] to take the image size as 900×1600 and utilize 10 sweeps to densify the LiDAR point cloud. During training, we adopt similar data augmentation strategies in [23] for both the image and LiDAR data. In our experiments, we utilize the AdamW [9] optimizer with a weight decay of 0.01 and an initial learning rate of $2e^{-4}$. We also utilize the cosine learning rate scheduler with linear warming up in the first 500 iterations. During training, we first construct the diffusion process from ground truth to noisy occupancy and then train the model to reverse this process. Algorithm 1 provides the pseudo-code of OccGen training procedure. The inference procedure of OccGen is a denoising sampling process from noise to 3D semantic occupancy. Starting from 3D voxel grids sampled in Gaussian distribution, the OccGen progressively refines its predictions, as shown in Algorithm 2. All models are trained and inferenced with a batch size of 8 on 8 V100 GPUs.

Algorithm 2 Inference algorithm

Input: Multi-modal inputs: $\{X_p, X_c\}$; Generative steps: T;
Output: Prediction occupancy $\hat{Y}$

Extract multi-modal features. $F_p, F_c \leftarrow \textbf{Extractor}(X_p, X_c)$
Aggregate the camera features. $F_c \leftarrow \textbf{Aggregate}(F_c, M_p)$
Obtain the multi-modal fusion features. $F_m \leftarrow \textbf{Fuser}(F_p, F_c)$;
Initialize the noise map. $Y_T \leftarrow \textbf{Randn}(\text{mean=0, std=1})$
for $i = 1, 2, ..., T$ **do**
    if $i > 1$ **then**
        Update the current 3D noise map. $Y_i \leftarrow Y_{i-1}$
    else
        Obtain the refined noise map. $Y_i \leftarrow \textbf{Refine}(F_m, Y_i, t)$
        Predict the occupancy results. $\hat{Y}_i \leftarrow \textbf{Voxel2Occ}(Y_i)$
        Obtain the noise map for the next evaluation. $Y_i \leftarrow \textbf{DDIM}(Y_i, i)$
    end if
end for

C Additional Experiments

Results on Occ3D-nuScenes. We also compare our OccGen with the state-of-the-art vision-based 3D occupancy prediction methods [12, 14, 24, 27] on Occ3D-nuScenes [22]. For a fair comparison, we removed the LiDAR stream and fusion module from the conditional encoder and followed the same backbone and image size of FB-Occ [14]. As shown in Tab. A, we can see that OccGen achieves the highest mIoU compared with all existing SOTA methods, demonstrating the effectiveness of OccGen for semantic scene completion.

Additional Results on nuScenes-Occupancy. We use IoU and mIoU for accuracy and Params and FPS for model efficiency. The results are shown in

Table A: Semantic occupancy prediction results on Occ3D-nuScenes validation set.

Method	Backbone	Image Size	mIoU
OccFormer [27]	ResNet-50	900 × 1600	36.5
SurroundOcc [24]	InternImage-B	900 × 1600	40.7
VoxFormer [12]	ResNet-101	900 × 1600	40.7
FB-Occ [14]	ResNet-50	900 × 1600	41.8
Ours	ResNet-50	900 × 1600	**42.6**

Table B: Performance on nuScenes-Occupancy (validation set). We report geometric metric IoU, semantic metric mIoU, parameters and FPS for each method. The C, L, M denotes *camera, LiDAR* and *multi-modal*. Best camera-only, LiDAR-only, and multi-modal results are marked as **red**, **blue**, and **black**, respectively.

Method	Input	IoU	mIoU	Params	FPS	barrier	bicycle	bus	car	const. veh.	motorcycle	pedestrian	traffic cone	trailer	truck	drive. suf.	other flat	sidewalk	terrain	manmade	vegetation
C-Baseline [23]	C	19.3	10.3	97.5M	5.8	9.9	6.8	11.2	11.5	6.3	8.4	8.6	4.3	4.2	9.9	22.0	15.8	14.1	13.5	7.3	10.2
L-Baseline [23]	L	30.8	11.7	66.1M	6.9	12.2	4.2	11.0	12.2	8.3	4.4	8.7	4.0	8.4	10.3	23.5	16.0	14.9	15.7	15.0	17.9
M-Baseline [23]	M	29.1	15.1	122.7M	4.1	14.3	12.0	15.2	14.9	13.7	15.0	13.1	9.0	10.0	14.5	23.2	17.5	16.1	17.2	15.3	19.5
C-CONet [23]	C	20.1	12.8	116.4M	3.5	13.2	8.1	**15.4**	17.2	6.3	11.2	10.0	8.3	4.7	12.1	31.4	18.8	18.7	16.3	4.8	8.2
L-CONet [23]	L	30.9	15.8	66.1M	4.0	17.5	**5.2**	13.3	18.1	7.8	5.4	9.6	5.6	13.2	13.6	34.9	21.5	22.4	21.7	19.2	23.5
M-CONet [23]	M	29.5	20.1	143.7M	2.9	23.3	13.3	21.2	24.3	**15.3**	15.9	18.0	13.3	15.3	20.7	33.2	21.0	22.5	21.5	19.6	23.2
C-OccGen (step1)	C	23.0	14.2	115.3M	3.4	15.5	9.1	15.0	18.9	6.6	11.6	11.4	8.8	5.4	13.1	34.4	21.4	21.6	18.8	5.6	9.6
C-OccGen (step2)	C	23.3	14.4	115.3M	3.2	14.8	8.5	15.2	19.0	7.3	11.4	11.9	8.3	6.0	13.9	34.6	22.0	21.6	19.5	5.7	9.8
C-OccGen (step3)	C	**23.4**	**14.5**	115.3M	3.0	**15.5**	**9.1**	15.3	**19.2**	**7.3**	**11.3**	**11.8**	**8.9**	**5.9**	**13.7**	**34.8**	**22.0**	**21.8**	**19.5**	**6.0**	**9.9**
L-OccGen (step1)	L	31.1	16.1	65.0M	4.0	17.6	4.1	14.3	19.1	6.6	7.1	11.0	6.2	13.2	14.3	35.8	21.3	22.2	20.9	20.1	24.2
L-OccGen (step2)	L	31.4	16.6	65.0M	3.9	18.7	5.1	15.0	19.3	7.3	7.8	11.2	6.3	13.7	14.3	36.3	21.9	22.7	21.9	20.2	24.1
L-OccGen (step3)	L	**31.6**	**16.8**	65.0M	3.7	**18.8**	5.1	**14.8**	**19.6**	**7.0**	**7.7**	**11.5**	**6.7**	**13.9**	**14.6**	**36.4**	**22.1**	**22.8**	**22.3**	**20.6**	24.5
OccGen (step1)	M	29.3	21.7	143.6M	2.8	25.4	16.6	22.2	26.0	13.4	19.9	21.8	14.6	17.3	22.1	35.4	24.1	24.1	22.8	19.5	22.3
OccGen (step2)	M	29.7	21.8	143.6M	2.5	24.8	16.8	22.4	25.9	13.8	20.3	21.7	14.6	17.5	21.9	35.2	24.5	24.3	23.5	19.5	22.5
OccGen (step3)	M	**30.3**	**22.0**	143.6M	2.3	**24.9**	**16.4**	**22.5**	**26.1**	14.0	**20.1**	**21.6**	**14.6**	**17.4**	**21.9**	**35.8**	**24.5**	**24.7**	**24.0**	**20.5**	**23.5**

Tab. B. Compared with the representative discriminative methods, OccGen achieves better results when using only one sampling step, with fewer parameters and comparable FPS on the camera-only, LiDAR-only, or multi-modal methods. When adopting three sampling steps, the performance is further boosted to 22.0%, 16.8%, and 14.5% on the multi-modal, camera-only, and LiDAR-only benchmarks, at a loss of 0.3 $\sim$ 0.5 FPS. These results show that OccGen can progressively refine the output occupancy multiple times with reasonable time cost. We note that OccGen consistently delivers the best IoU results across almost all categories in the third step, which indicates that our method can better complete the scenes due to our coarse-to-fine generation property. We also observe that camera-only methods are more time-consuming compared to LiDAR-only methods due to the 2D-to-3D view transformation. This indicates that a more efficient LSS method is urgent.

Scaling factor. The performance of different scaling factors is shown in Tab. Ca. We found the lower scaling factor 0.001 has achieved a bit lower performance than 0.01. A larger scaling factor means more noise is added to the estimate,

Table C: The diffusion settings of progressive refinement layer on nuScenes-Occupancy. We report Iou and mIoU. Default settings are marked in gray.

(a) Scaling factor. The best scaling factor is 0.01.

scale	IoU	mIoU
0.001	30.0	21.8
0.01	30.3	22.0

(b) Noise schedule. Consine works best.

type	IoU	mIoU
cosine	30.3	22.0
linear	29.9	21.4

(c) Sampling strategy. Using DDIM works best.

type	IoU	mIoU
DDIM	30.3	22.0
DDPM	29.2	21.7

Table D: Ablations on hard 2D-to-3D view transformation and depth supervision under the multi-modal setting. "*Hard LSS*" and "*Depth Supervision*" denote hard 2D-to-3D view transformation and the generated depth ground truth following [13, 23] , respectively.

	Hard LSS	Depth Supervision	IoU	mIoU
(a)	-	-	25.1	19.4
(b)	✓	-	28.6	20.6
(c)	-	✓	29.5	20.1
(d)	✓	✓	29.4	20.8

typically resulting in faster convergence but potentially reducing the quality of sampling. Conversely, a smaller scaling factor reduces the noise level but may require more iteration steps to converge, thereby increasing sampling time.

Noise schedule. As shown in Tab. Cb, we compare the effectiveness of the cosine schedule [18] and linear schedule [7] in OccGen for occupancy prediction. We observe that the model using a cosine schedule achieves better performance (22.0% vs. 21.4%). The possible reason is that the cosine schedule allows for a smooth reduction in noise, promoting more stable learning dynamics and the linear schedule may sometimes exhibit a more abrupt transition, and its impact on model convergence and sample quality can differ from the cosine schedule.

Sampling strategy. As shown in Tab. Cc, we compare the effectiveness of the DDIM [20] and DDPM [7] sampling strategies in OccGen, and find that the model using DDIM is better than DDPM. DDIM uses a non-Markovian diffusion process to accelerate sampling and DDPM is defined as the reverse of a Markovian diffusion process.

The effectiveness of hard 2D-to-3D view transformation. We also conduct experiments to fully exploit the effectiveness of hard 2D-to-3D view transformation based on CONet [23] under the multi-modal setting. The results are shown in Tab. D We observe that "Depth supervision" proposed in BEVDepth [13] can boost the performance of occupancy prediction significantly. This indicates that the accurately predicted depth can lead to more complete occupancy. We also note that our proposed hard 2D-to-3D view transformation can achieve comparable results without adopting depth supervision, which demonstrates the effectiveness of the hard Gumbel softmax on depth prediction.

Table E: Ablation study of backbone selection, input size of different modality, and number of progressive refinement layers. *C,L* denotes camera and LiDAR.

	Method	2D Backbone	Input Size	Layers	IoU	mIoU
(a)	C-OccGen	R-50	704×256	six	21.8	13.0
	C-OccGen	R-50	1600×900	six	23.4	14.5
	C-OccGen	R-101	1600×900	six	23.3	15.0
(b)	L-OccGen	-	1 sweep	six	30.4	15.9
	L-OccGen	-	10 sweeps	six	31.6	16.2
(c)	OccGen	R-50	1600×900 10 sweeps	one	29.4	21.6
	OccGen	R-50	1600×900 10 sweeps	three	29.9	21.7
	OccGen	R-50	1600×900 10 sweeps	six	30.4	22.0

Table F: The Expected Calibration Error (ECE) metric of OpenOcc, CONet, and OccGen (with different steps).

	OpenOcc	CONet	OccGen(1)	OccGen(2)	OccGen(3)	OccGen(4)	OccGen(5)
ECE(%)	3.72	4.03	3.52	3.25	3.23	3.23	3.25

Different Experiment Setting. In this subsection, we ablate the different experiment settings (*e.g.*, input size, backbone selection, number of progressive refinement layers) in Tab. E. For the camera-based OccGen, using a larger input size (1600×900) relatively improves IoU and mIoU by 7.3% and 11.5%. Besides, replacing ResNet-50 with ResNet-101 can further improve the performance of mIoU. For the LiDAR-based OccGen, it is observed that utilizing multi-sweeps as input (following [11,25,26], 10 sweeps are used) perform well the single-sweep counterpart on IoU and mIoU. For the multi-modal OccGen, we observe that the number of refinement layers has a discernible impact on the performance. The performance tends to increase with a greater number of layers.

More visualization. We visualize the predicted results of OccFormer [27] and OccGen on SemanticKITTI [1] in Fig A. We can observe that OccGen produces more reasonable results than OccFormer [27]. In Fig. B, we visualize the additional predicted results of 3D semantic occupancy on nuScenes-Occupancy from CONet [23] and our proposed OccGen. The "drivable surface" and "sidewalk" regions predicted by our OccGen exhibit superior continuity and integrity. This results in a significant reduction in the number of void areas compared to the previous SOTA CONet [23]. In Fig. C, we visualize the additional predicted results of 3D occupancy of different sampling steps. We observe that the results of the third step have more complete geometric structure and semantic information compared with the generated results of the first step. In Fig. D, we note that

the uncertainty maps of different steps clearly show that the proposed OccGen can iteratively refine the occupancy in a coarse-to-fine manner.

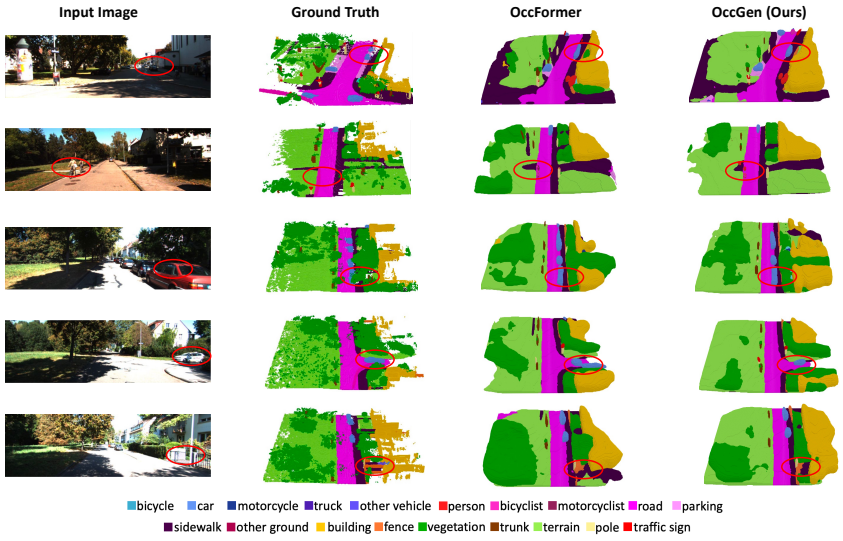

Fig. A: Qualitative results of semantic scene completion on SemanticKITTI [1] validation set. The leftmost column shows the input image and the following three columns visualize the results from the ground truth, OccFormer [27], and Our OccGen.

Quantitative uncertainty evaluation. We compare the Expected Calibration Error (ECE) metric of OpenOcc, CONet, and OccGen (with different steps) and show results in Tab. F. OccGen demonstrates superior calibration compared to OpenOcc and CONet, highlighting its suitability for making more reliable predictions. We also noticed that the calibration gradually reached saturation in the third step.

D Broader Impact Statement and Limitations

This paper studies a generative model for 3D occupancy semantic prediction and does not see potential privacy-related issues. Nevertheless, deploying a model biased toward the training data may introduce significant safety concerns and potential risks when utilized in real-world applications. This research is simple yet effective, which may inspire the community to produce follow-up generative studies for 3D occupancy.

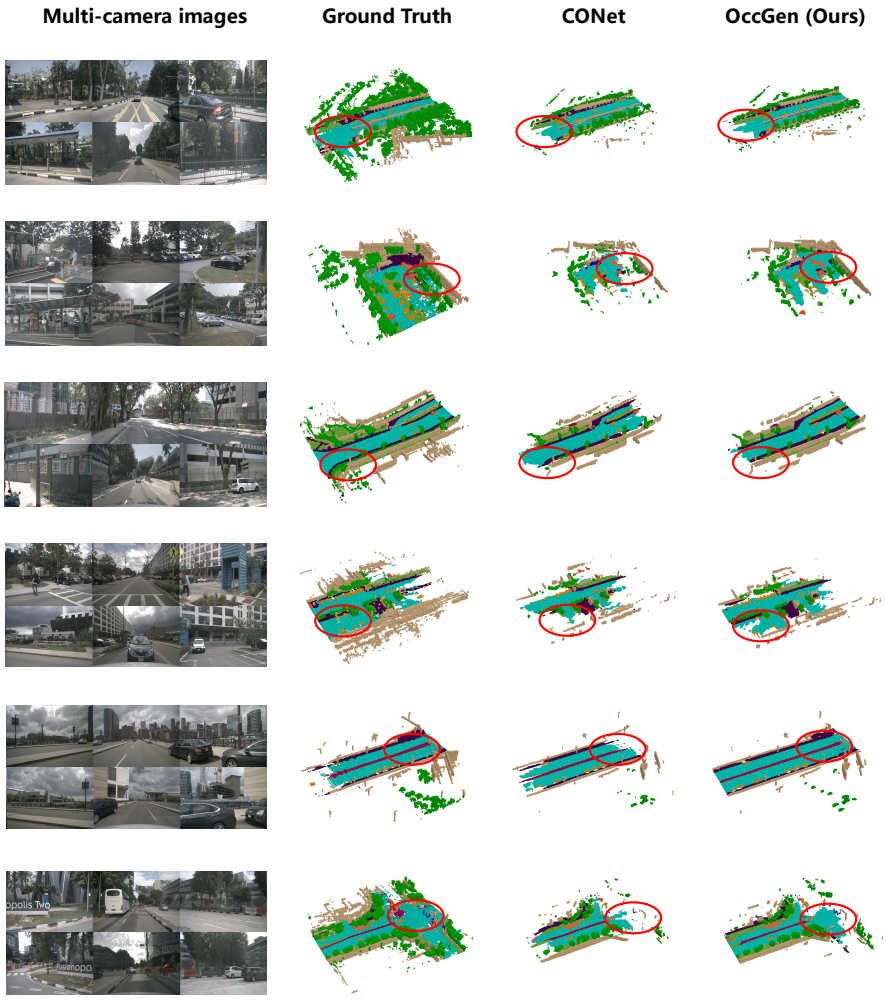

Fig. B: Additional qualitative results of the 3D semantic occupancy predictions on nuScenes-Occupancy. The leftmost column shows the input surrounding images, the following three columns visualize the 3D semantic occupancy results from the ground truth, CONet [23], and OccGen. The regions highlighted by red circles indicate that these areas have obvious differences (better viewed when zoomed in).

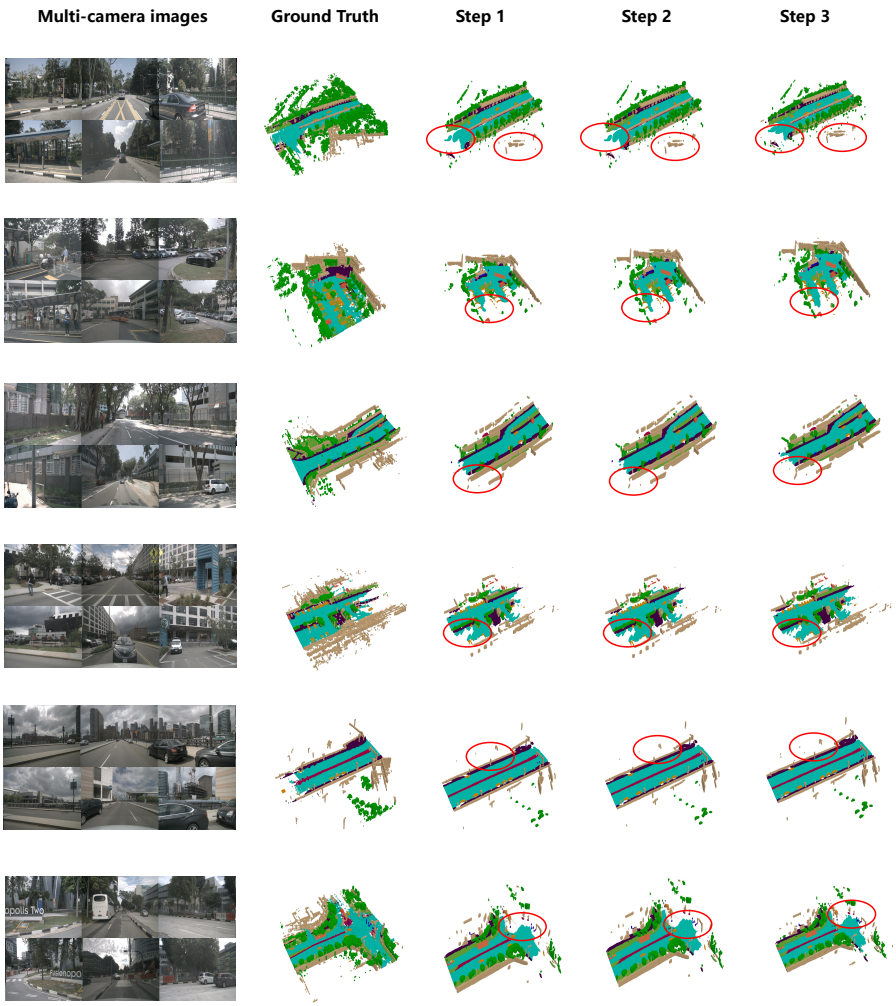

Fig. C: Additional predicted occupancy results on the different steps of OccGen on nuScenes-Occupancy. The leftmost column shows the input surrounding images, and the following four columns visualize the 3D semantic occupancy results from the ground truth, first step, second step, and third step. The regions highlighted by red circles indicate that these areas have obvious differences (better viewed when zoomed in).

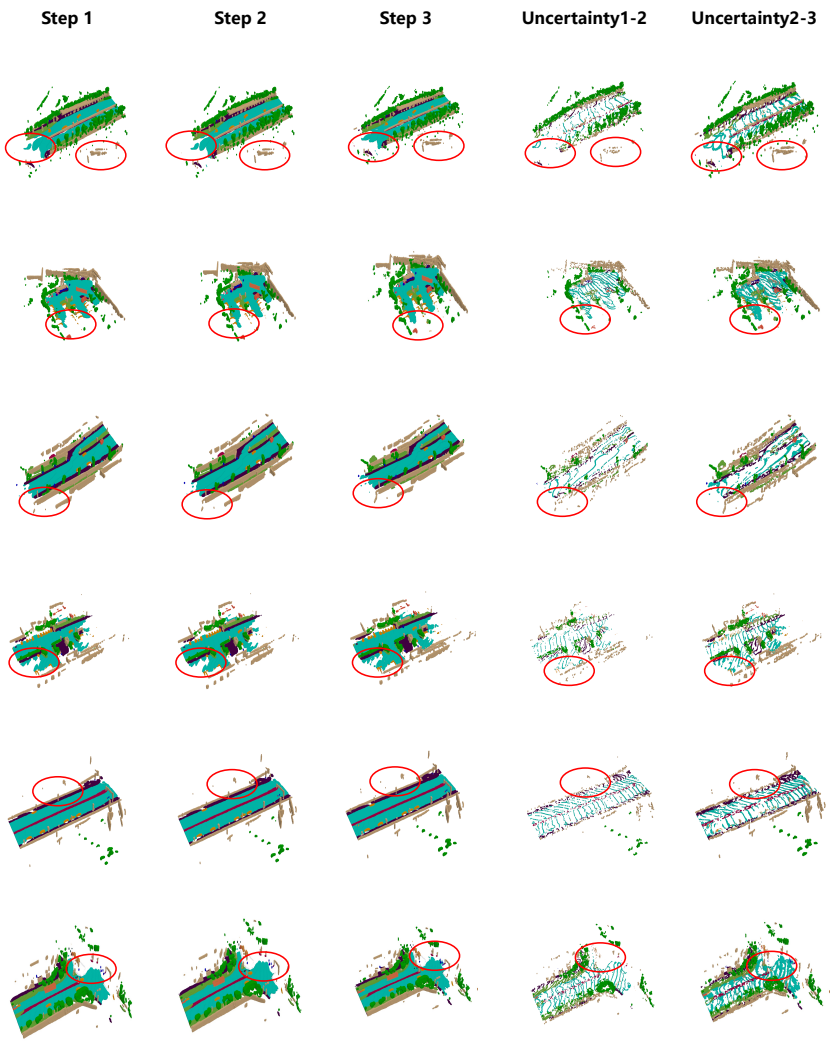

Fig. D: Additional uncertainty estimates between different steps of OccGen on nuScenes-Occupancy. The three left columns show the predicted 3D semantic occupancy results from the first, second, and third steps. The “Uncertainty 1-2" and “Uncertainty 2-3" represent the high estimated uncertainty voxels from step one to step two and from step two to step three, respectively. The regions highlighted by red circles indicate that these areas have obvious differences (better viewed when zoomed in).

References

1. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: Semantickitti: A dataset for semantic scene understanding of lidar sequences. In: ICCV. pp. 9297–9307 (2019)
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR. pp. 11621–11631 (2020)
3. Cao, A.Q., de Charette, R.: Monoscene: Monocular 3d semantic scene completion. In: CVPR. pp. 3991–4001 (2022)
4. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR. pp. 3354–3361 (2012)
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS. vol. 27 (2014)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
7. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS. vol. 33, pp. 6840–6851 (2020)
8. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
10. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
11. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR. pp. 12697–12705 (2019)
12. Li, Y., Yu, Z., Choy, C., Xiao, C., Alvarez, J.M., Fidler, S., Feng, C., Anandkumar, A.: Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In: CVPR. pp. 9087–9098 (2023)
13. Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., Li, Z.: Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. arXiv preprint arXiv:2206.10092 (2022)
14. Li, Z., Yu, Z., Austin, D., Fang, M., Lan, S., Kautz, J., Alvarez, J.M.: Fb-occ: 3d occupancy prediction based on forward-backward view transformation. arXiv preprint arXiv:2307.01492 (2023)
15. Liang, T., Xie, H., Yu, K., Xia, Z., Lin, Z., Wang, Y., Tang, T., Wang, B., Tang, Z.: Bevfusion: A simple and robust lidar-camera fusion framework. In: NeurIPS (2022)
16. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR. pp. 2117–2125 (2017)
17. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D., Han, S.: Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In: ICRA (2023)
18. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: ICML. pp. 8162–8171. PMLR (2021)
19. Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: ECCV. pp. 194–210. Springer (2020)
20. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020)

21. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: CVPR. pp. 1746–1754 (2017)
22. Tian, X., Jiang, T., Yun, L., Wang, Y., Wang, Y., Zhao, H.: Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. arXiv preprint arXiv:2304.14365 (2023)
23. Wang, X., Zhu, Z., Xu, W., Zhang, Y., Wei, Y., Chi, X., Ye, Y., Du, D., Lu, J., Wang, X.: Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. In: ICCV. pp. 17850–17859 (2023)
24. Wei, Y., Zhao, L., Zheng, W., Zhu, Z., Zhou, J., Lu, J.: Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In: ICCV. pp. 21729–21740 (2023)
25. Yan, Y., Mao, Y., Li, B.: SECOND: Sparsely embedded convolutional detection. Sensors (2018)
26. Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: CVPR. pp. 11784–11793 (2021)
27. Zhang, Y., Zhu, Z., Du, D.: Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction. In: ICCV. pp. 9433–9443 (2023)
28. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: CVPR. pp. 4490–4499 (2018)