2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

GSRender: Deduplicated Occupancy Estimation via Weakly Supervised 3D Gaussian Splatting

Qianpu Sun[1,2], Sifan Zhou[2], Changyong Shu[2], Sirui Han[3*], Yuan Chun[1*]

***Abstract*— Weakly-supervised 3D occupancy perception is crucial for vision-based autonomous driving in outdoor environments. Previous methods based on NeRF often face a challenge in balancing the number of samples used. Too many samples can decrease efficiency, while too few can compromise accuracy, leading to variations in the mean Intersection over Union (mIoU) by 5-10 points. Furthermore, even with surrounding-view image inputs, only a single image is rendered from each viewpoint at any given moment. This limitation leads to duplicated predictions, which significantly impacts the practicality of the approach. However, this issue has largely been overlooked in existing research. To address this, we propose GSRender, which uses 3D Gaussian Splatting for weakly-supervised occupancy estimation, simplifying the sampling process. Additionally, we introduce the Ray Compensation module, which reduces duplicated predictions by compensating for features from adjacent frames. Finally, we redesign the dynamic loss to remove the influence of dynamic objects from adjacent frames. Extensive experiments show that our approach achieves SOTA results in RayIoU (+6.0), while also narrowing the gap with 3D-supervised methods. This work lays a solid foundation for weakly-supervised occupancy perception. The code is available at `https://github.com/Jasper-sudo-Sun/GSRender`.**

I. INTRODUCTION

Autonomous driving has advanced rapidly in recent years. As a key task, occupancy estimation has attracted attention from both academia and industry. This task models the scene as a grid-based collection, where each grid is assigned relevant attributes such as semantics and motion flow. Compared to traditional 3D object detection [1], [2], [3] and BEV(Bird's Eye View) perception [4], [5], 3D occupancy estimation provides a more fine-grained understanding of the environment and compensates for the inability of BEV space to capture height information. Researchers are committed to advancing the accuracy and real-time efficiency of comprehensive scene estimation, aiming to push the boundaries of understanding and interaction with dynamic environments.

Most existing approaches [6], [7], [8], [9], [10], [11], [12], [13], [14] are heavily dependent on 3D ground truth. As noted in RenderOcc [15], generating 30,000 frames of 3D labels requires approximately 4,000 human hours. Moreover, inconsistencies in 3D label generation across benchmarks further complicate real-world applications. Thus, reducing or eliminating reliance on 3D labels is a critical challenge, particularly as the demand for 3D perception in autonomous driving grows.

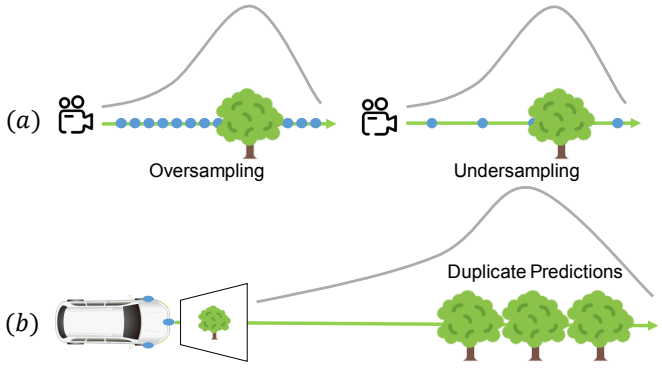

Fig. 1. **The limitations of RenderOcc [15]**. (a) NeRF-based method face a trade-off between efficiency and precision. (b) Duplicate predictions caused by the uncertainty in depth estimation.

RenderOcc [15] introduces a method that uses LiDAR point clouds to directly supervise predictions, removing the need for post-processing. Even though RenderOcc is relatively superior, there are still two issues that require our attention, as shown in Figure 1. **(a) Sampling Trade-off:** NeRF-based methods necessitate sampling 3D points along camera rays, but achieving a balanced sampling remains a significant challenge. This complex trade-off may cause accuracy fluctuations ranging from 5 to 10 percentage points. If sampling is insufficient, some positions might be completely undetectable. **(b) Duplicated Predictions:** Rendering merely relying on single-view images often leads to duplicated predictions along camera rays. Such duplications artificially inflate the mean Intersection over Union (mIoU) metric [16], thereby reducing the practical utility of 3D occupancy estimation when applied to downstream tasks.

To address these challenges, we introduce GSRender, an end-to-end framework that leverages 3D Gaussian Splatting [17], which enables high-quality rendering of photorealistic scenes by utilizing multiple viewpoint images of the same object. But in occupancy estimation tasks, there are no multi-view images; instead, only surrounding-view images are used, with minimal overlap between different viewpoints. Specifically, we achieve multi-view supervision of the same object by incorporating adjacent frames to address the issue of duplicated predictions. Furthermore, to minimize the potential negative impact of dynamic objects on the rendered scenes, we redesign a differentiated loss for static and dynamic objects, which leads to further performance improvements.

[1]Qianpu Sun and Yuan Chun are with Tsinghua Shenzhen International Graduate School, Tsinghua University, China.
[2]Qianpu Sun, Changyong Shu and Sifan Zhou are with Houmo AI, China.
[3]Sirui Han is with The Hong Kong University of Science and Technology, Hong Kong SAR, China.
*Corresponding author: Sirui Han, Yuan Chun.

We conducted all experiments on the nuScenes [18] dataset. It outperforms RenderOcc in the conventional mIoU [19] metric. We also employed RayIoU [16], a novel metric specifically designed to more accurately measure the model's true predictive performance. Impressively, our method achieved **SOTA** results in RayIoU under the category of weakly-supervised methods. Specifically, we summarize our contributions as follows:

1) We present **GSRender** and a detailed explanation of the effectiveness of 3D Gaussian Splatting in the analysis section for the weakly-supervised occupancy estimation task.
2) We propose the **Ray Compensation module** and a refined loss to mitigate duplicated predictions and handle dynamic and static objects in auxiliary frames, enhancing the practicality of weakly supervised methods.
3) We achieved **SOTA** performance in RayIoU metrics among all 2D weakly-supervised methods and substantially reduced the performance gap compared to methods utilizing 3D supervision.

II. Related Work

1) Voxel-based Scene Representation.: Efficient representations have become a key focus. Voxel-based representations enable fine-grained representations, driving success in LiDAR segmentation [20], [21], [22], [14], [23] and 3D scene completion [24], [25], [26]. Since Tesla AI Day 2022 [27], occupancy perception has gained increasing attention. Mainstream models fall into two paradigms: the first uses the LSS method [28], [7], [11], [13], [29], which predicts depth within the ego-vehicle's coordinate system, while the second employs Transformer-based methods [30] [31], [32], [33], [34] with deformable attention [35] to refine BEV features. However, these methods often rely on dense features, whereas most voxels in the space are empty, leading to computational redundancy in real-world scenarios. Recent works have focused on simplifying the feature space, achieving notable results [16], [36], [37], [38], [14].

2) Gaussian Scene Representation.: 3D Gaussian Splatting [17] has been applied widely [39], [40], [41], [42], [43], [44], [45], [46], offering real-time scene reconstruction. It outperforms NeRF in both performance and rendering quality. Recent work treats large-scale outdoor scenes as Gaussian collections, including GaussianFormer [37], which enhances each Gaussian using a transformer, and GaussianBEV [47], which replaces LSS with Gaussian Splatting for BEV feature extraction. GaussianOcc [48] integrates cross-view information to minimize sensitivity to camera poses. Our research also models outdoor scenes as Gaussians for detailed reconstruction.

3) Weakly-Supervised Estimation.: Due to challenges in obtaining 3D labels, many works focus on weakly-supervised paradigms for semantic scene completion. [49] achieves semantic completion from a single image, and S4C [50] improves completion by aligning objects across multiple frames. For occupancy estimation, several methods aim to reduce reliance on 3D labels. RenderOcc [15] uses a NeRF-based approach to bridge semantic and density fields with depth maps from LiDAR. OccNeRF [51] and SelfOcc [52] use self-supervision for generating 2D semantic and depth maps. However, the issue of duplicated predictions from a single viewpoint persists due to the limited overlap between different viewpoints at the same time. We address this issue by introducing temporal supervision, which helps mitigate the problem of duplicated predictions from a single viewpoint.

III. Preliminary

We first briefly introduce the essential knowledge of 3D Gaussian Splatting, the problem definition, and the evaluation metrics we use.

1) 3D Gaussian Splatting.: 3D Gaussian Splatting [17] is a highly popular method in the field of 3D reconstruction, used for real-time reconstruction of both single objects and large-scale scenes. It conceptualizes the entire scene as a collection of Gaussians. Each Gaussian encompasses attributes such as the mean and covariance of the Gaussian distribution, along with opacity and color. The Gaussian probability for each spatial point is calculated as follows:

$$G(x) = e^{-\frac{1}{2}(X-\mu)^T \Sigma \ ^1 (X-\mu)}, \tag{1}$$

where μ represents the position of each Gaussian, while the covariance matrix Σ to determine the size and orientation of the Gaussian distribution. Specifically, the covariance matrix can be computed using rotation R and scale S matrices by:

$$\Sigma = RSS^T R^T. \tag{2}$$

For independent optimization, we represent the scale as a 3D vector $s \in \mathbb{R}^3$ and the rotation as a quaternion $q \in \mathbb{R}^4$. Then, the 3D Gaussians are projected onto the 2D imaging plane using transformation matrices W and Jacobian matrices J. The covariance matrix of the 2D Gaussian is computed as follows:

$$\Sigma' = JW\Sigma W^T J^T. \tag{3}$$

Finally, standard alpha-blending is utilized to render the rgb colors of all Gaussians onto the image based on their opacity, and the loss is computed against the image. The color of each pixel is calculated as follows:

$$c(x) = \sum_{k=1}^{K} c_k \alpha_k G_k^{2D}(x) \prod_{j=1}^{k-1} \left(1 - \alpha_j G_j^{2D}(x)\right). \tag{4}$$

where α_k represents the opacity of the kth Gaussian, $G_k^{2D}(x)$ is the Gaussian probability at pixel position x, and c_k denotes the color of the k-th Gaussian, represented by spherical harmonic coefficients [53]. K denotes the number of Gaussians contributing to pixel x.

2) Problem Definition.: we aim to construct a comprehensive 3D occupancy representation of the scene using images captured from surrounding viewpoints. Specifically, for the vehicle at timestamp t, we take N surrounding images $I = \{I_1, I_2, \ldots, I_N\}$ as input and predict the 3D occupancy grid

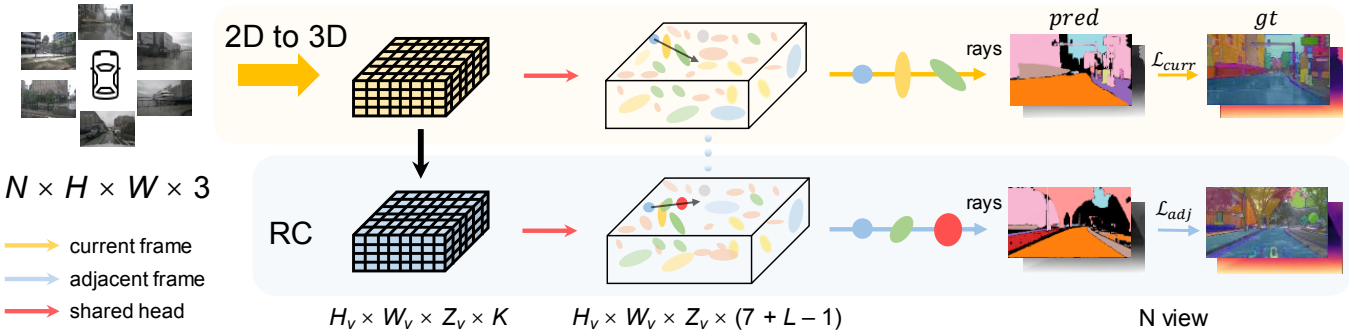

Fig. 2. **Overall Framework of GSRender**. For surround view image input, we employ an arbitrary 2D to 3D module to extract occupancy features. Using a simple Gaussian head, we predict the attributes of each Gaussian, followed by Gaussian rendering. Then, we achieve compensation for different viewpoints of the same object through the Ray Compensation (RC) module, alleviating the issue of duplicate predictions.

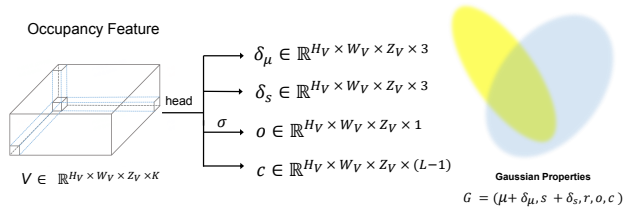

Fig. 3. **Gaussian Properties Field**. The occupancy feature from the 2D to 3D module is fed into the Gaussian head, which outputs the shift of Gaussian's mean δ_μ, scales δ_s, opacity o, and semantic logits c, representing the Gaussian's location, scale, visibility, and semantic category.

$V \in \mathbb{R}^{H_V \times W_V \times Z_V \times L}$, where H_V, W_V, and Z_V denote the resolution of the voxel space, and L represents the number of categories, including free category for unoccupied space. The formulation for this 3D prediction task is articulated as follows:

$$P = \mathbb{H}(I_1, I_2, ..., I_N), V = \mathbb{F}(P), \tag{5}$$

Specifically, $\mathbb{H}$ generates an occupancy feature $P \in \mathbb{R}^{H_V \times W_V \times Z_V \times C}$, where C denote the feature dimension, capturing spatial structure and temporal relationships within the scene. Following this, the function $\mathbb{F}$ serves as a head that takes the scene feature representation P as input and outputs the final prediction V. RenderOcc [15] employs a NeRF-based rendering head and achieves multi-view alignment through Auxiliary Rays. In GSRender, we replace the head with a Gaussian-based head and refine the auxiliary rays module, implementing a more efficient approach.

3) Evaluation metrics.: We utilized both the mIoU metric and the newly proposed RayIoU [16] metric to evaluate our method. The RayIoU metric simulates LiDAR by projecting query rays into the predicted 3D occupancy. A query ray is considered a true positive (TP) if its class matches and the $L1$ depth error between the ground-truth and predicted depths is below a threshold (e.g., 2m). The query ray can originate from the LiDAR position at the current, past, or future moments along the ego path. Temporal casting allows us to evaluate scene completion performance while maintaining a well-posed task. Given C classes, RayIoU is computed as follows:

$$\text{RayIoU} = \frac{1}{C} \sum_{c=1}^{C} \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c}, \tag{6}$$

where TP_c, FP_c, and FN_c correspond to the number of true positive, false positive, and false negative predictions for class c_i. We believe this evaluation method better reflects the model's true performance, so we focus on using it as our primary evaluation metric.

IV. Method

A. Overall Framework.

Figure 2 illustrates our overall framework. We conceptualize the entire scene as an assembly of numerous Gaussian. In the feature extraction stage, we adopt the same backbone to ensure a fair comparison with RenderOcc. First, we treat each voxel center as the center of a Gaussian and predict the offset of their properties within the Gaussian. Next, we render the semantics logits and depth and optimize the network with LiDAR rendering supervision. Finally, we introduce a multi-frame calibration module aimed at reducing false positive detections caused by duplicate predictions.

B. Gaussian Properties Field.

We treat each voxel center as the initial point for a gaussian distribution, as shown in Figure 3. For a given occupancy feature V, in addition to predicting the semantic and opacity attributes, we initialize the Gaussian's mean and scale, and predict offsets for each property, excluding rotation. In structured voxel spaces, we contend that rotating the Gaussian is unnecessary. Instead, adjusting scale suffices to meet the spatial requirements of the entire voxel space. The regular arrangement of voxels in a uniformly distributed and structured environment already provides sufficient geometric and spatial information, rendering rotational adjustments redundant. Our ablation studies have confirmed this approach. The resulting Gaussian properties $G \in \mathbb{R}^{N \times (7+(L-1))}$, where N denotes the total number of Gaussians. For each Gaussian, we have

$$G(i) = (\mu + \delta_\mu, s + \delta_s, r, o, c). \tag{7}$$

During the testing phase, we treat Gaussians with opacity greater than a predefined threshold as the final predictions.

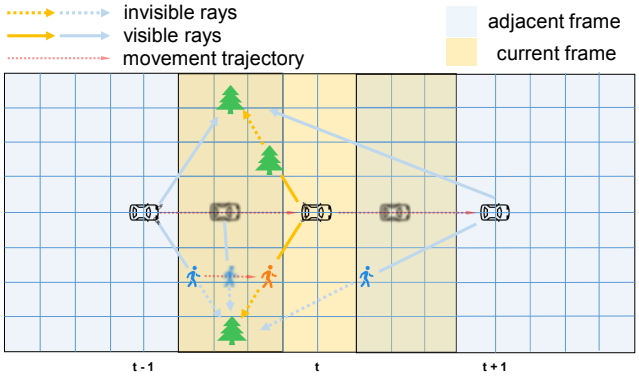

Fig. 4. **Ray Compensation**. In the upper part of the feature map, it indicates that adjacent frame compensation is used to address occlusion in the current frame. In the lower part of the feature map, it shows that dynamic objects may occlude the view of the compensated frame again, so it's necessary to reduce the contribution of dynamic objects in adjacent frame.

Specifically

$$V(\mu) = \begin{cases} \arg\max\left(C(\mu)\right) & \text{if } o(\mu) \geq \tau \\ \varnothing & \text{if } o(\mu) < \tau \end{cases} \tag{8}$$

where τ is the opacity threshold for determining whether a Gaussian is considered empty. Since empty voxels are filtered out based on threshold, the actual number of predicted semantic categories is $L-1$, excluding the 'free' category. Visualization analysis shows that the majority of effective Gaussians closely align with their voxel centers. Thus, we apply the Gaussian semantics directly to the voxels.

1) Gaussian Rasterizer.: We interpret the semantic logits associated with each Gaussian as colors, and apply alpha-blending to render the semantic logits and depth for each pixel. This approach culminates in the generation of a semantic map S and a depth map D.

C. Ray Compensation Module.

RenderOcc [15] introduces Auxiliary Rays to ensure spatial consistency across different viewpoints by aligning labeled rays from multiple frames to a single frame. Nevertheless, it fails to resolve the duplicated predictions along these rays. We introduce the Ray Compensation module, which provides multiple perspectives for each object at the feature level. Specifically as shown in Figure 4. The upper part of figure illustrates the principle of our method. For the current frame t, trees that are obscured might be revealed in adjacent frames from the different viewpoints. To obtain features from adjacent time steps, we first assume that the current Occupancy Feature is denoted as $P_{\text{curr}} \in \mathbb{R}^{H_V \times W_V \times Z_V \times C}$. We treat the centers of voxels as homogeneous coordinates $O_{\text{curr}}^{\text{curr\_ego}} \in \mathbb{R}^{H_V \times W_V \times Z_V \times 4}$, representing the voxel centers in the ego-car coordinate system at the current time. The position of the adjacent time step in the current time step's coordinate system is computed as follows:

$$O_{\text{adj}}^{\text{curr\_ego}} = T_{\text{adj\_ego}}^{\text{curr\_ego}} O_{\text{adj}}^{\text{adj\_ego}}, \tag{9}$$

where $T_{\text{adj\_ego}}^{\text{curr\_ego}} \in \mathbb{R}^{4\times4}$ is the transformation matrix from the ego-car coordinate system of the adjacent frame to that of the current frame. The features of adjacent frames are mathematically represented by the following formula:

$$\begin{aligned} P_{adj}^{t+k} &= \mathbb{S}(P_{curr}^{t}, O_{\text{adj}}^{\text{curr\_ego}}), \\ \text{for} \quad k &= -1, 1, \ldots, -\frac{T_m}{2}, \frac{T_m}{2}. \end{aligned} \tag{10}$$

Here, $\mathbb{S}$ denotes the sampling function, where we apply bilinear interpolation, and T_m represents the total number of adjacent frames selected. Finally, the shared gaussian head predicts the Gaussian properties for the adjacent frame.

1) Dynamic Object.: It is noteworthy that this approach may still encounter issues, as illustrated in the lower part of Figure 4. For trees obscured in the current frame, they may still be blocked by other moving objects in adjacent frames. In extreme cases, these obstructions might occur at every time step. Therefore, we need to minimize the influence of dynamic objects in adjacent frames. This is achieved by applying a weighting factor, alpha, when calculating the loss for adjacent frames, as detailed in the following subsection.

D. Loss Function

To maintain consistency with RenderOcc, we also optimize the Gaussian properties using a class-balanced cross-entropy loss $\mathcal{L}^{\text{seg}}$ for occupancy prediction and the SILog loss $\mathcal{L}^{\text{depth}}$ [55] for depth estimation. For the current frame, we compute the loss similarly to RenderOcc as follows:

$$\mathcal{L}_{curr} = \mathcal{L}_{curr}^{seg} + \mathcal{L}_{curr}^{depth}. \tag{11}$$

For adjacent frames, the inability to estimate the motion trajectories of moving objects can negatively impact network optimization. Therefore, a lower weight is assigned to dynamic objects. The adjacent loss can be formulated as follows:

$$\mathcal{L}_{adj} = \mathcal{L}_{adj}^{seg} + \mathcal{L}_{adj}^{depth}. \tag{12}$$

The segmentation loss $\mathcal{L}_{adj}^{seg}$ and depth loss $\mathcal{L}_{adj}^{depth}$ are defined as follows:

$$\mathcal{L}_{adj}^{seg} = -\sum_{i=1}^{n} \alpha_{s(i)} \beta_{s(i)} \mathbf{S}_{(i)} \log(\hat{\mathbf{S}}_{pix}(i)), \tag{13}$$

$$\mathcal{L}_{adj}^{depth} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \alpha_{s(i)} d_i^2 - \frac{1}{n^2}\left(\sum_{i=1}^{n} \alpha_{s(i)} d_i\right)^2}, \tag{14}$$

where $d_i = \log \mathbf{D}_i - \log \hat{\mathbf{D}}_i$ denotes the difference between predicted and ground truth depth at pixel i, while $s(i)$ is the ground truth semantic category. $\alpha_{s(i)}$ and $\beta_{s(i)}$ represent the dynamic weight and category balance weight for $s(i)$, respectively. Consequently, our final loss function can be expressed as:

$$\mathcal{L} = \mathcal{L}_{\text{curr}} + \sum_{k} \omega_k \mathcal{L}_{adj,k} \tag{15}$$

where ω represents the weight assigned to adjacent frames. Additionally, although we also considered using the distortion loss proposed in 2D Gaussian Splatting [56], we found it to be less effective and decided not to include it.

method	GT	mIoU	others	barrier	bicycle	bus	car	cons. veh	motorcycle	pedestrian	traffic cone	trailer	truck	dri. sur	other flat	sidewalk	terrain	manmade	vegetation
BEVFormer [32]	3D	23.67	5.03	38.79	9.98	34.41	41.09	13.24	16.50	18.15	17.83	18.66	27.70	48.95	27.73	29.08	25.38	15.41	14.46
BEVStereo [7]	3D	24.51	5.73	38.41	7.88	38.70	41.20	17.56	17.33	14.69	10.31	16.84	29.62	54.08	28.92	32.68	26.54	18.74	17.49
OccFormer [33]	3D	21.93	5.94	30.29	12.32	34.40	39.17	14.44	16.45	17.22	9.27	13.90	26.36	50.99	30.96	34.66	22.73	6.76	6.97
SelfOcc [52]	2D*	9.30	-	0.15	0.66	5.46	12.54	0.00	0.80	2.10	0.00	0.00	8.25	55.49	-	26.30	26.54	14.22	5.60
OccNeRF [51]	2D*	9.54	-	0.83	0.82	5.13	12.49	3.50	0.23	3.10	1.84	0.52	3.90	52.62	-	20.81	24.75	18.45	13.19
GaussianOcc [48]	2D*	9.94	-	1.79	5.82	14.58	13.55	1.30	2.82	7.95	9.76	0.56	9.61	44.59	-	20.10	17.58	8.61	10.29
RenderOcc(1f) [15]	LiDAR-2D	19.33	1.82	19.54	12.6	16.08	12.85	**8.76**	12.14	9.33	12.94	17.60	12.13	56.25	28.83	**36.89**	38.82	15.08	16.98
GSRender(1f)	LiDAR-2D	**21.36**	**2.44**	**20.13**	**13.56**	**19.54**	**18.38**	8.35	**18.77**	**10.67**	**14.81**	**16.63**	**14.92**	**61.74**	**29.77**	35.95	**40.56**	**19.83**	**17.06**
RenderOcc [15]	LiDAR-2D+3D	26.11	4.84	31.72	10.72	27.67	26.45	13.87	18.20	**17.67**	**17.84**	21.19	23.25	63.20	36.42	46.21	44.26	19.58	20.72
GSRender	LiDAR-2D+3D	**29.56**	**8.42**	**34.93**	**15.12**	**30.71**	**29.61**	**16.70**	**9.48**	17.61	16.58	**23.92**	**27.24**	**77.94**	**39.21**	**51.69**	**54.61**	**23.82**	**24.92**

TABLE I

QUANTITATIVE COMPARISON ON THE OCC3D-NUSCENES DATASET IN MIOU METRIC. 2D* DENOTES THE SEMANTIC AND DEPTH MAPS GENERATED USING A SEMI-SUPERVISED METHOD, RATHER THAN BY PROJECTING LIDAR POINT CLOUDS ONTO THE IMAGING PLANE. THE BEST RESULTS OBTAINED USING ONLY 2D SUPERVISION ARE HIGHLIGHTED IN BOLD.

Method	GT	Backbone	Input Size	Epoch	RayIoU	RayIoU_{1m}	RayIoU_{2m}	RayIoU_{4m}
FB-Occ (16f) [12]	3D	R50	704×256	90	33.5	26.7	34.1	39.7
SparseOcc(8f) [16]	3D	R50	704×256	24	34.0	28.0	34.7	39.4
Panoptic-FlashOcc (1f) [11]	3D	R50	704×256	24	35.2	29.4	36.0	40.1
SimpleOcc [54]	3D	R101	672×336	12	22.5	17.0	22.7	27.9
OccNeRF [51]	2D*	R101	640×384	24	10.5	6.9	10.3	14.3
GaussianOcc [48]	2D*	R101	640×384	12	11.9	8.7	11.9	15.0
RenderOcc(2f)	LiDAR-2D	Swin-B	1408×512	12	19.3	12.7	19.3	25.9
RenderOcc(7f)	LiDAR-2D	Swin-B	1408×512	12	19.5	13.4	19.6	25.5
GSRender(2f)	LiDAR-2D	Swin-B	1408×512	12	**25.5**(+6.0)	**18.7**(+5.3)	**25.8**(+6.2)	**31.8**(+6.3)

TABLE II

QUANTITATIVE COMPARISON ON THE OCC3D-NUSCENES DATASET IN RAYIOU METRIC. THE BEST RESULTS OBTAINED USING ONLY LIDAR-2D SUPERVISION ARE HIGHLIGHTED IN BOLD.

V. EXPERIMENTS

We trained and evaluated our method on the nuscenes dataset, leveraging the surround-view camera setup for better feature interaction. Additionally, we introduced RayIoU [16] to highlight the effectiveness of multi-frame compensation and conducted ablation studies for deeper insights into GSRender.

A. Experimental Setup

1) Dataset.: All of our experiments were conducted on the nuscenes dataset [18] and the corresponding occupancy dataset, OCC3D-NuScenes [57]. The occupancy dataset contains 600 scenes for training and 150 scenes for validation. It covers a region centered around the ego-vehicle with dimensions $[-40\text{m}, -40\text{m}, -1\text{m}, 40\text{m}, 40\text{m}, 5.4\text{m}]$ and uses a voxel size of $[0.4\text{m}, 0.4\text{m}, 0.4\text{m}]$. This dataset contains 18 classes, where classes 0 to 16 follow the same definitions as the nuScenes-lidarseg dataset, and class 17 represents free space. We followed the approach used in RenderOcc [15] to project the semantic and depth labels of 3D LiDAR points onto the 2D maps.

2) Implementation details.: We implemented our model using PyTorch. We used off-the-shelf architecture Swin Transformer [58] as the image backbone with an input resolution of 512×1408 and BEVStereo [7] for extracting 3D features. For the gaussian head, we simply employed a two-layer fully connected network. For the RC module, we used only one future frame as the basic experimental configuration for all experiments. For balance weights, we follow the original RenderOcc configuration. For dynamic weights, we simply set α to 0.1 and ω to 0.8. During the training phase, we utilized the Adam optimizer with a linear learning rate scheduling strategy, set the batch size to 16, and trained for 12 epochs with a learning rate of 2e-4. All experiments were conducted on NVIDIA A6000 GPUs.

B. Performance

We report the performance of GSRender in this part, demonstrating its superior results compared to the existing SOTA method. We achieved significant performance improvements in RayIoU. Building on these enhancements, we now turn to a detailed comparison, both qualitative and quantitative, to further validate the efficacy of our approach.

1) Quantitative Comparison.: As illustrated in Table I, The objective was to explore the performance of both methods when limited to a single-frame scenario, using mIoU as the evaluation metric. GSRender outperformed RenderOcc, achieving a higher mIoU, and enhanced performance when used as auxiliary supervision alongside 3D supervision, improving from **26.11 to 29.56**. When supervised only by the current frame, the inherent limitations of single view supervision cause both RenderOcc and GSRender to produce a thick surface. Consequently, relying solely on mIoU does not accurately reflect the model's performance. To truly assess the capabilities of our model, we adopted RayIoU [16] as our primary evaluation metric to better evaluate the model's true capability in scene reconstruction. As shown in Table 2. it is surprising that by utilizing only one frame for auxiliary

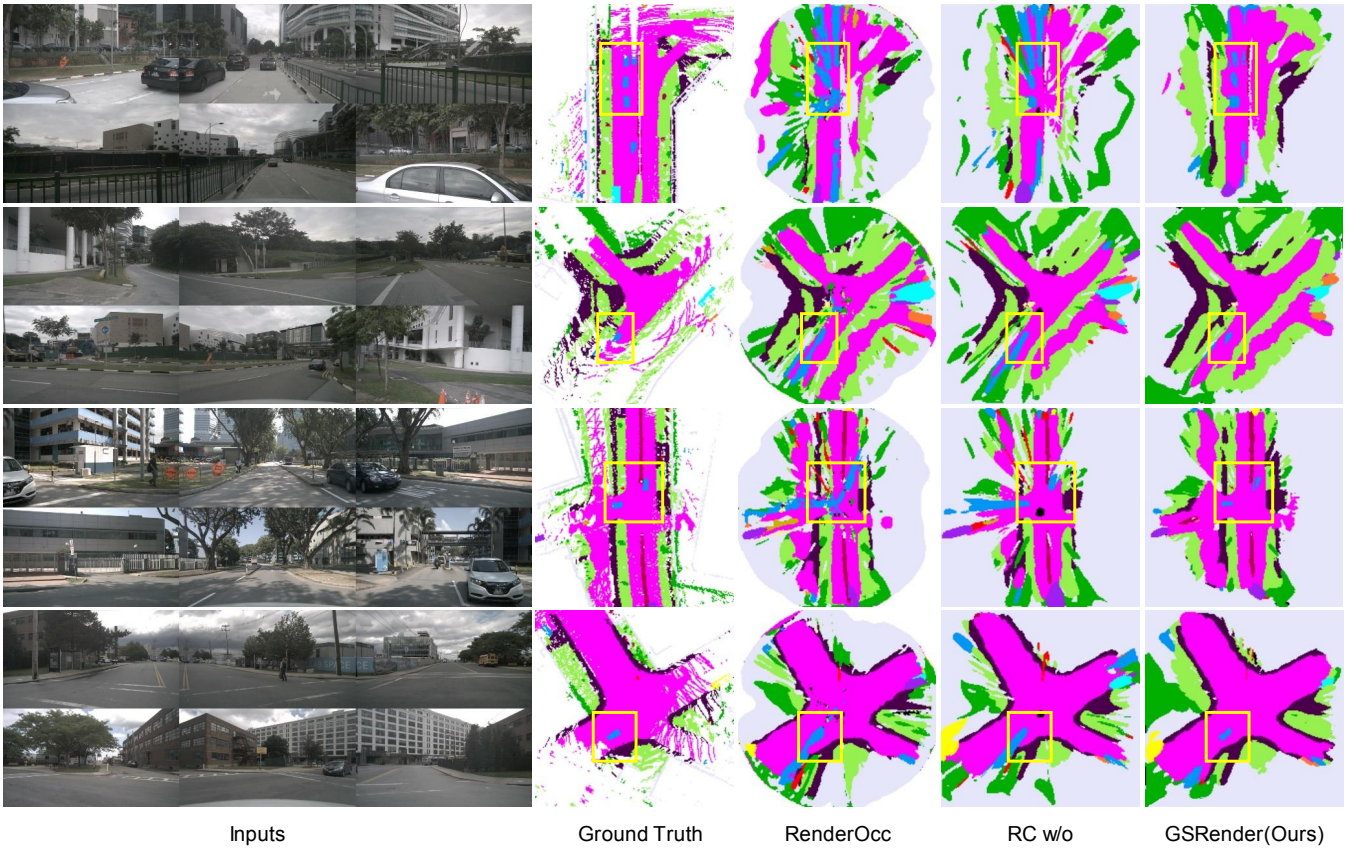

Fig. 5. **Qualitative results of GSRender**. Top view on Occ3D-nuScenes

supervision. Our results achieved a significant improvement over RenderOcc which uses six auxiliary frames. In fact our approach even surpasses certain methods that rely on 3D occupancy ground truth such as SimpleOcc.

2) Qualitative Comparison.: As illustrated in Figure 5, our experimental results demonstrate that the RC module significantly enhances the accuracy in positioning and shaping of specific categories, such as vehicles. In the third column, we visualized the results without using the RC module. This comparison clearly shows that reliance on single-frame ground truths tends to result in duplicated predictions. However, the integration of the RC module effectively addresses this problem, leading to a more unified and seamless reconstruction of the entire scene and enhancing the practicality of the prediction results.

3) Ablations and Analysis.: In our initial experiments, we postulated that using fixed Gaussian properties could also yield satisfactory rendering results. Nonetheless, this does not hold true. Therefore, we conducted ablation studies to verify the necessity of shifts in each Gaussian attribute, as shown in Table IV, indicate that Gaussian trained without mean and scale shift δ_μ, δ_s and with rotation shift δ_r exhibit decreased accuracy. Furthermore, in Table IV, we also validated the effectiveness of the dynamic weight ω and the adjacent weight α. Also presents the ablation experiment showing only the current frame as supervision, without the RC module. It can be seen that the RC module brings a significant improvement in performance. Notably, the mIoU for using only a single frame is comparable to that with the RC module, which is attributed to the hacked mIoU caused by duplicated predictions.

	RayIoU(%)	mIoU(%)
Ours	**25.5**	**22.2**
Ours w δ_r	24.0	22.1
Ours w/o δ_s	8.5	6.2
Ours w/o δ_μ	19.5	16.1
Ours $\lvert\delta_\mu\rvert \leq 0.2$	21.0	18.7

TABLE III

ABLATION OF EACH GAUSSIAN PROPERTIES

RC	ω	α	RayIoU(%)	mIoU(%)
✓	✓	✓	**25.5**	**22.2**
✓	✓	×	24.1	21.9
✓	×	✓	24.5	22.1
✓	×	×	23.5	21.8
×	×	×	19.7	21.4

TABLE IV

ABLATION OF RC MODULE AND LOSS WEIGHTS.

To better investigate the benefits of the Gaussian representation, we excluded Gaussians that do not correspond to any occupied space, then analyzed the distribution of mean shifts among the remaining Gaussians, as illustrated in Figure 6. Notably, the majority of these Gaussian means remained within their original positions, with shifts not exceeding half voxel. This stability enabled us during inference to directly associate the semantic labels of the Gaussians with their corresponding voxels, leading to robust results.

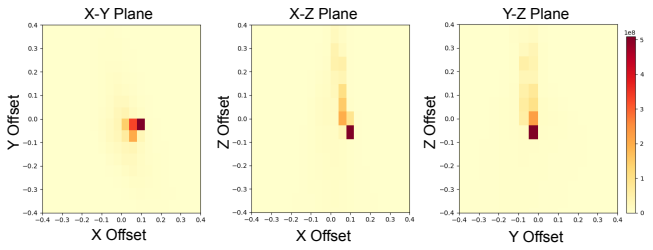

Fig. 6. Statistics of δ_μ distribution. Most Gaussians move within the voxel, with a notable deviation in the z-direction.

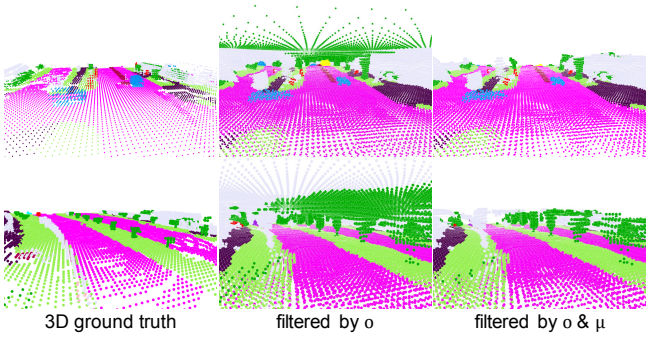

Fig. 7. Comparison after filtering Gaussians.

Furthermore, we visualized more intuitive outcomes, as demonstrated in the Figure 7, where the positions of Gaussian distributions were depicted as a point cloud. The second column displays results filtered by opacity alone. Many Gaussians in the sky lack supervision, which results in Gaussians near object surfaces being categorized as adjacent objects and exhibiting a tendency to shift closer to these objects. This also confirms the common z-offset in many Gaussians exceeds half a voxel and shows, as in Table III, that overly constraining the Gaussian mean μ degrades performance. Building on this, as shown in the third column of Figure 7, we have additionally filtered out Gaussians that have shifted beyond the confines of their voxels. The results indicates that most Gaussians near real object surfaces adhered to their voxel positions. Additionally, since the 2D ground truth is directly derived from LiDAR point clouds, it more accurately reflects the distribution of objects in the real world compared to 3D ground truth. Consequently, our method achieves object shapes , like trees, that are more realistic than the 3D ground truth.

VI. Limitation and Future Work

Qualitative results indicate that our method still has several limitations. We treat each voxel as a Gaussian, but in practice, such a large number of redundant Gaussians is unnecessary, leading to increased computational burden and memory usage. Furthermore, the sparsity of LiDAR supervision could also impede the optimization of Gaussians, thereby affecting the overall performance of the model. GSRender serves as a foundation for our future work, where we will focus on reducing the number of Gaussians to further optimize performance.

VII. Conclusion

In this work, we explored the application of 3D Gaussian Splatting in 3D occupancy prediction tasks. We implemented a method to mitigate the issue of duplicate predictions along the same camera ray when using 2D weakly supervision. Our experiments validated the effectiveness of our approach, achieving state-of-the-art performance among current 2D-supervised methods. And provided a detailed analysis for the performance improvement. This technique has the potential to become a powerful tool in both industry and academia.

References

[1] R. Qian, X. Lai, and X. Li, "3d object detection for autonomous driving: A survey," *Pattern Recognition*, vol. 130, p. 108796, 2022.

[2] S. Zhou, Z. Tian, X. Chu, X. Zhang, B. Zhang, X. Lu, C. Feng, Z. Jie, P. Y. Chiang, and L. Ma, "Fastpillars: A deployment-friendly pillar-based 3d detector," *arXiv preprint arXiv:2302.02367*, 2023.

[3] S.-h. Kim and Y. Hwang, "A survey on deep learning based methods and datasets for monocular 3d object detection," *Electronics*, vol. 10, no. 4, p. 517, 2021.

[4] Y. Ma, T. Wang, X. Bai, H. Yang, Y. Hou, Y. Wang, Y. Qiao, R. Yang, and X. Zhu, "Vision-centric bev perception: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[5] L. He, Q. Wang, H. Sun, Q. Xu, B. Gao, S. E. Li, J. Wang, and K. Li, "Vision-driven 2d supervised fine-tuning framework for bird's eye view perception," *arXiv preprint arXiv:2409.05834*, 2024.

[6] H. Liu, Y. Teng, T. Lu, H. Wang, and L. Wang, "Sparsebev: High-performance sparse 3d object detection from multi-camera videos," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18 580–18 590.

[7] Y. Li, H. Bao, Z. Ge, J. Yang, J. Sun, and Z. Li, "Bevstereo: Enhancing depth estimation in multi-view 3d object detection with temporal stereo," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 2, 2023, pp. 1486–1494.

[8] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3d semantic occupancy prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 9223–9232.

[9] S. Xu, P. Li, Q. Sun, X. Liu, Y. Li, S. Guo, Z. Wang, B. Jiang, R. Wang, K. Sheng *et al.*, "Lion: Learning point-wise abstaining penalty for lidar outlier detection using diverse synthetic data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 9, 2025, pp. 8951–8959.

[10] J. Huang and G. Huang, "Bevdet4d: Exploit temporal cues in multi-camera 3d object detection," *arXiv preprint arXiv:2203.17054*, 2022.

[11] Z. Yu, C. Shu, Q. Sun, J. Linghu, X. Wei, J. Yu, Z. Liu, D. Yang, H. Li, and Y. Chen, "Panoptic-flashocc: An efficient baseline to marry semantic occupancy with panoptic via instance center," *arXiv preprint arXiv:2406.10527*, 2024.

[12] Z. Li, Z. Yu, D. Austin, M. Fang, S. Lan, J. Kautz, and J. M. Alvarez, "Fb-occ: 3d occupancy prediction based on forward-backward view transformation," *arXiv preprint arXiv:2307.01492*, 2023.

[13] Y. Li, Z. Ge, G. Yu, J. Yang, Z. Wang, Y. Shi, J. Sun, and Z. Li, "Bevdepth: Acquisition of reliable depth for multi-view 3d object detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 2, 2023, pp. 1477–1485.

[14] P. Tang, Z. Wang, G. Wang, J. Zheng, X. Ren, B. Feng, and C. Ma, "Sparseocc: Rethinking sparse latent representation for vision-based semantic occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 15 035–15 044.

[15] M. Pan, J. Liu, R. Zhang, P. Huang, X. Li, H. Xie, B. Wang, L. Liu, and S. Zhang, "Renderocc: Vision-centric 3d occupancy prediction with 2d rendering supervision," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 404–12 411.

[16] P. Tang, Z. Wang, G. Wang, J. Zheng, X. Ren, B. Feng, and C. Ma, "Sparseocc: Rethinking sparse latent representation for vision-based semantic occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 035–15 044.

[17] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[18] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[19] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[20] V. E. Liong, T. N. T. Nguyen, S. Widjaja, D. Sharma, and Z. J. Chong, "Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation," *arXiv preprint arXiv:2012.04934*, 2020.

[21] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, "Searching efficient 3d architectures with sparse point-voxel convolution," in *European conference on computer vision*. Springer, 2020, pp. 685–702.

[22] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12 547–12 556.

[23] M. Ye, R. Wan, S. Xu, T. Cao, and Q. Chen, "Drinet++: Efficient voxel-as-point point cloud segmentation," *arXiv preprint arXiv:2111.08318*, 2021.

[24] A.-Q. Cao and R. De Charette, "Monoscene: Monocular 3d semantic scene completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3991–4001.

[25] L. Roldao, R. De Charette, and A. Verroust-Blondet, "Lmscnet: Lightweight multiscale 3d semantic completion," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 111–119.

[26] X. Chen, K.-Y. Lin, C. Qian, G. Zeng, and H. Li, "3d sketch-aware semantic scene completion via semi-supervised structure prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4193–4202.

[27] "Tesla ai day," 2022, https://www.youtube.com/watch?v=j0z4FweCy4M.

[28] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer, 2020, pp. 194–210.

[29] J. Huang, G. Huang, Z. Zhu, Y. Ye, and D. Du, "Bevdet: High-performance multi-camera 3d object detection in bird-eye-view," *arXiv preprint arXiv:2112.11790*, 2021.

[30] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[31] Y. Li, Z. Yu, C. Choy, C. Xiao, J. M. Alvarez, S. Fidler, C. Feng, and A. Anandkumar, "Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 9087–9098.

[32] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *European conference on computer vision*. Springer, 2022, pp. 1–18.

[33] Y. Zhang, Z. Zhu, and D. Du, "Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9433–9443.

[34] Y. Wei, L. Zhao, W. Zheng, Z. Zhu, J. Zhou, and J. Lu, "Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 729–21 740.

[35] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[36] Y. Shi, T. Cheng, Q. Zhang, W. Liu, and X. Wang, "Occupancy as set of points," *arXiv preprint arXiv:2407.04049*, 2024.

[37] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Gaussianformer: Scene as gaussians for vision-based 3d semantic occupancy prediction," *arXiv preprint arXiv:2405.17429*, 2024.

[38] J. Wang, Z. Liu, Q. Meng, L. Yan, K. Wang, J. Yang, W. Liu, Q. Hou, and M.-M. Cheng, "Opus: Occupancy prediction using a sparse set," *arXiv preprint arXiv:2409.09350*, 2024.

[39] S. Szymanowicz, C. Rupprecht, and A. Vedaldi, "Splatter image: Ultra-fast single-view 3d reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 208–10 217.

[40] D. Charatan, S. L. Li, A. Tagliasacchi, and V. Sitzmann, "pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 457–19 467.

[41] Y. Xu, Z. Shi, W. Yifan, H. Chen, C. Yang, S. Peng, Y. Shen, and G. Wetzstein, "Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation," *arXiv preprint arXiv:2403.14621*, 2024.

[42] Y. Yan, H. Lin, C. Zhou, W. Wang, H. Sun, K. Zhan, X. Lang, X. Zhou, and S. Peng, "Street gaussians for modeling dynamic urban scenes," *arXiv preprint arXiv:2401.01339*, 2024.

[43] Q. Shen, X. Yi, Z. Wu, P. Zhou, H. Zhang, S. Yan, and X. Wang, "Gamba: Marry gaussian splatting with mamba for single view 3d reconstruction," *arXiv preprint arXiv:2403.18795*, 2024.

[44] Q. Gao, Q. Xu, Z. Cao, B. Mildenhall, W. Ma, L. Chen, D. Tang, and U. Neumann, "Gaussianflow: Splatting gaussian dynamics for 4d content creation," *arXiv preprint arXiv:2403.12365*, 2024.

[45] Z. Lu, X. Guo, L. Hui, T. Chen, M. Yang, X. Tang, F. Zhu, and Y. Dai, "3d geometry-aware deformable gaussian splatting for dynamic view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 8900–8910.

[46] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang, "4d gaussian splatting for real-time dynamic scene rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 310–20 320.

[47] F. Chabot, N. Granger, and G. Lapouge, "Gaussianbev: 3d gaussian representation meets perception models for bev segmentation," *arXiv preprint arXiv:2407.14108*, 2024.

[48] W. Gan, F. Liu, H. Xu, N. Mo, and N. Yokoya, "Gaussianocc: Fully self-supervised and efficient 3d occupancy estimation with gaussian splatting," *arXiv preprint arXiv:2408.11447*, 2024.

[49] F. Wimbauer, N. Yang, C. Rupprecht, and D. Cremers, "Behind the scenes: Density fields for single view reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 9076–9086.

[50] A. Hayler, F. Wimbauer, D. Muhle, C. Rupprecht, and D. Cremers, "S4c: Self-supervised semantic scene completion with neural fields," in *2024 International Conference on 3D Vision (3DV)*. IEEE, 2024, pp. 409–420.

[51] C. Zhang, J. Yan, Y. Wei, J. Li, L. Liu, Y. Tang, Y. Duan, and J. Lu, "Occnerf: Self-supervised multi-camera occupancy prediction with neural radiance fields," *arXiv preprint arXiv:2312.09243*, 2023.

[52] Y. Huang, W. Zheng, B. Zhang, J. Zhou, and J. Lu, "Selfocc: Self-supervised vision-based 3d occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 946–19 956.

[53] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 5501–5510.

[54] W. Gan, N. Mo, H. Xu, and N. Yokoya, "A simple attempt for 3d occupancy estimation in autonomous driving," *arXiv preprint arXiv:2303.10076*, 2023.

[55] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[56] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2d gaussian splatting for geometrically accurate radiance fields," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.

[57] X. Tian, T. Jiang, L. Yun, Y. Mao, H. Yang, Y. Wang, Y. Wang, and H. Zhao, "Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[58] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.