

Omni-Scene: Omni-Gaussian Representation for Ego-Centric Sparse-View Scene Reconstruction

Supplementary Material

In this document, we first provide implementation details including data preprocessing of nuScenes [1] (Sec.1.1), network architecture and hyperparameters (Sec.1.2). We follow with additional experiment discussions including introduction to our supplementary videos (Sec.2.1), quantitative comparisons with more baseline methods (Sec.2.2), runtime analysis (Sec.2.3), more ablations (Sec.2.4), further discussions on generalizability to larger bins (Sec. 2.5) and effectiveness of our Volume-Pixel Collaboration (Sec. 2.6), more qualitative results on scene-centric reconstruction (Sec.2.7). **We strongly recommend readers to view the accompanying supplementary video (“video.mp4”),** which contains 360-degree exploring videos of both reconstructed and synthetic scenes, as well as comparisons with other methods.

1. Additional Implementation Details

1.1. Data Preprocessing

As described in Sec.4.1 of our main manuscript, we partition each scene of nuScenes dataset [1] into equally spaced bins, with each bin serving as one data sample. For nuScenes dataset, each video is captured in a single scene along with the car trajectory. The length of the trajectory ranges drastically from several meters to hundreds of meters. If we segment the trajectory into bins according to frame indexes, the spatial ranges of bins would exhibit significant variation, which leads to non-IID data distribution for training and evaluation. To circumvent this issue, we segment the bins based on the distance traveled by the car as detailed in Fig.1. Specifically, for videos with a trajectory length exceeding 3.2 meters, we uniformly segment them into N bins, each 3.2 meters in length. For each bin, we use the central frame with 6 surrounding images to derive

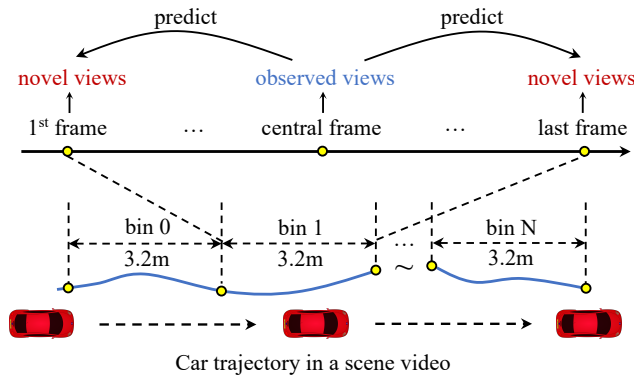


Figure 1. Data preprocessing of nuScenes [1].

the observed input views, and the first and last frames with 12 surrounding images as the novel views. For videos with a trajectory length less than 3.2 meters, we directly use the first and last frames of the video as the novel views.

1.2. Network Architecture and Hyperparameters

In Table 1(a), the order from top to bottom are the parameters of Triplane Transformer (i.e., number of transformer layers, embedding dimensions, number of 2D and 3D reference points used in our cross-image and cross-plane deformable attentions, and number of attention heads), Voxel Decoder (i.e., number of Gaussians decoded for each voxel, number of linear layers used for decoding Gaussian parameters), Multi-View U-Net (i.e., feature dimensions and patch sizes of patchified cross attentions [4] used in U-Net down-sample and upsample blocks), Pixel Decoder (i.e., number of convolution layers used for decoding Gaussian parameters), respectively. In Table 1(b), we specify loss weights for Eq.(4) in our main manuscript, which is followed by parameters used in our training phase.

(a) Network Architecture		
2D Image Encoder	backbone	R50-DINO [2]
	neck	FPN (P2 only) [10]
Triplane Transformer	# layers	3
	# embed dims	128
	# 2D ref points	8, 16, 16
	# 3D ref points	16, 16, 16
Voxel Decoder	# attn heads	8
	# Gaussians per voxel	3
Multi-View U-Net	# linear layers	3
	# downsample feats	128, 256, 512, 512
	# upsample feats	512, 512, 256, 128
	# downsample patches	8, 8, 4, 2
Pixel Decoder	# upsample patches	2, 4, 8, 8
	# conv layers	3
(b) Hyperparameters		
Loss Weights	# $\lambda_1, \lambda_2, \lambda_{V_1}, \lambda_{V_2}$	0.05, 1.0, 0.05, 0.01
	learning rate scheduler	Cosine
Training Details	# iterations	100,000
	# learning rate	1e-4
	optimizer	Adam [9]
	# beta1, beta2	0.9, 0.999
	# weight decay	0.01
	# warm-up	1000
	# gradient clip	1.0

Table 1. Details of network architecture and hyperparameters. In the table, “#” denotes numerical parameters. We present parameters that specify our network architecture, and parameters used in our loss functions and training phase, in (a) and (b), respectively.

Method	Time(s)	Param(M)	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PCC \uparrow
AttnRend [6]	9.98	125.1	20.96	0.533	0.467	N/A
MuRF [13]	0.672	5.3	20.34	0.504	0.433	-0.332
pixelSplat [3]	0.508	125.4	21.51	0.616	0.372	0.001
MVSplat [5]	<u>0.174</u>	<u>12.0</u>	<u>21.61</u>	<u>0.658</u>	<u>0.295</u>	<u>0.181</u>
Ours	0.088	81.7	24.27	0.736	0.237	0.800

Table 2. Additional quantitative results on ego-centric reconstruction task performed on nuScenes [1]. We **bold 1st-place** results and underline 2nd-place results. PCC is not available (N/A) for AttnRend which has no interpretable 3D structure for depth rendering.

2. Additional Experiments

2.1. Video Results

To better demonstrate the quality of 3D reconstruction, we provide **exploring video demos in “video.mp4”** along with our supplementary material. Specifically, given six surrounding images of a scene, we conduct inference and obtain 3D Gaussians for reconstructing the scene. Then, we utilize these Gaussians to render a 360-degree rotation video at 30fps with the camera FOV set to 70 degree following [1]. In the video, each frame that falls between the input viewpoints can be considered as a novel view unseen in the inputs. To further demonstrate the model’s performance in the presence of object occlusions and frustum truncations, we move the camera forward and backward by 3 meters in the front and rear view perspectives, respectively, ensuring that there are contents invisible from the input views. It’s also noted that the camera’s movement range has reached 6 meters, exceeding the 3.2-meter range of bin samples seen during training, thereby showcasing the model’s capability to reconstruct scenes at greater distances.

Comparisons with other methods. We first present comparisons with state-of-the-art methods pixelSplat [3] and MVSplat [5] from 00:00 to 01:40 in “video.mp4”. Our approach significantly outperforms other methods in both visual and geometric quality. Notably, due to the minimal cross-view overlap among input views, pixelSplat and MVSplat fail to predict accurate depths based on pixel-level 3D priors (e.g., epipolar lines, cost volumes), which results in artifacts in the rendered videos especially when the camera is substantially moved forward and backward.

Exploring videos of reconstructed scenes. Then, we present more examples to illustrate our functionality on scene reconstruction. Examples with normal conditions are shown from 01:41 to 02:49 in “video.mp4”. Examples with extreme conditions (e.g., low-light, bad weather) are shown from 02:50 to 03:26 in “video.mp4”. We can see that our method achieves high-quality reconstruction and maintains robustness in both normal and hard cases.

Exploring videos of generated scenes. We also present examples to illustrate our functionality on scene generation from 03:27 to 05:07 in “video.mp4”. The left side of the video shows the our generated results given different random seeds. The right side of the video shows examples of MagicDrive3D [7], which are directly adopted from their

official website¹. We can see that our method achieves better visual details than per-scene optimization-based MagicDrive3D in a much more efficient feed-forward manner.

2.2. Comparisons with More Baselines

We also make comparisons with more baseline methods (i.e., MuRF [13] and AttnRend [6]) for ego-centric sparse-view reconstruction task. Specifically, MuRF and AttnRend are feed-forward reconstruction methods based on NeRF [11] and light field [12], respectively. They are both leading and representative methods within their respective lines of works, which constitute the mainstream feed-forward methods together with 3DGS-based approaches such as pixelSplat [3] and MVSplat [5]. As shown in Table 2, our method surpasses MuRF and AttnRend significantly in terms of all metrics. We can also observe that methods with explicit Gaussians as representations (i.e., ours, pixelSplat, MVSplat) outperform methods with implicit NeRF or light field as representations (i.e., AttnRend, MuRF), showing the effectiveness of explicit 3D representation.

2.3. Runtime Analysis

As shown in Table 2, we conduct runtime analysis on the ego-centric reconstruction task to demonstrate the efficiency of our method. It’s noted that the inference speed is reported based on the time cost of six-view reconstruction averaged by 2,048 times. From the table, we can see that our method achieves the shortest inference time (i.e., “Time” in Table 2), which is nearly 2 \times faster than that of the 2nd place method MVSplat [5]. We attribute this advantage to our triplane-based volume feature encoding in Triplane Transformer, and efficient patchified cross-attention module in Multi-View U-Net. Besides, our method is also lightweight with model size (i.e., “Param” in Table 2) comparable to other methods. Furthermore, we observe that, thanks to the efficient rendering of 3DGS [8], 3DGS-based methods (i.e., our method, pixelSplat [3], MVSplat [5]) show significant superiority in speed compared to methods based on implicit representations (i.e., MuRF [13], AttnRend [6]).

2.4. Additional Ablations

We present more ablation results to demonstrate the effectiveness of our components.

Qualitative Ablations on Volume-Pixel Collaboration. In

¹<https://gaoruiyuan.com/magicdrive3d/>

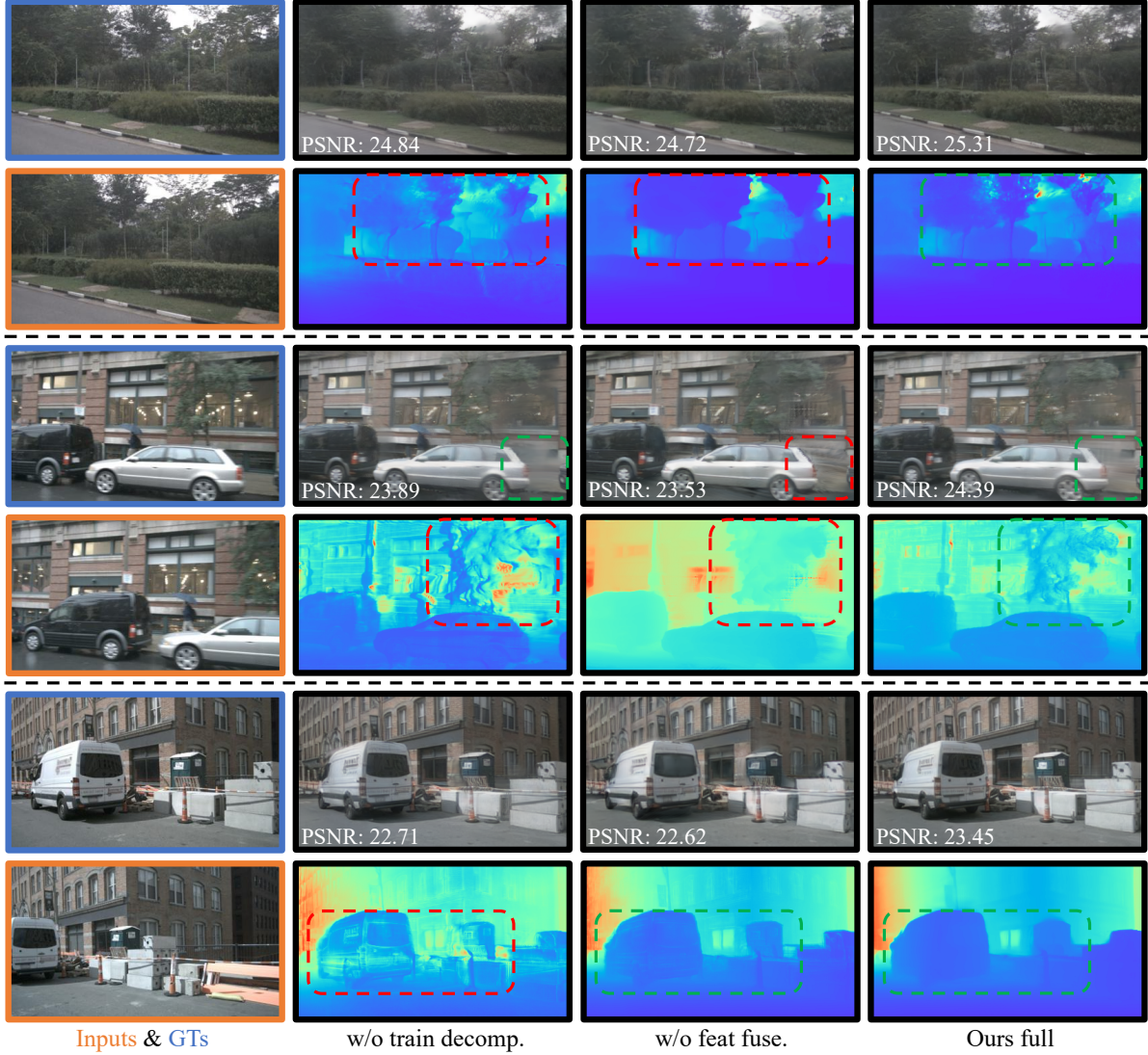


Figure 2. Qualitative ablations on Volume-Pixel Collaboration. Images of input views (Inputs) and ground-truth novel views (GTs) are outlined by orange and blue rectangles, respectively. The remaining are novel-view images and depths generated by our variant models and full model. From left to right, the order is the variant without Depth-Guided Training Decomposition, the variant without Projection-Based Feature Fusion, and our full method. The *red dashed lines* highlight undesirable artifacts (e.g., noise, over-smooth), while the *green ones* denote plausibly-rendered areas (e.g., better and sharper details). We also show PSNR values of the generated images for better comparison.

our main manuscript, we have quantitatively compared our full method with the two variants without the Volume-Pixel Collaboration designs (i.e., Projection-based Feature Fusion and Depth-Guided Training Decomposition). Here, we show additional qualitative results in Figure 2 for visual comparisons. It can be observed that our full method can generate images with higher quality and depths with sharper details, which demonstrate that our collaboration designs can effectively encourage the complementarity between pixel-based and volume-based Gaussian representations, and further improve the performance.

Qualitative Ablations on Depth Initialization. In our main manuscript, we have quantitatively demonstrate the

effectiveness of depth initialization for our pixel-based Gaussian representation. Here, we show additional qualitative results in Figure 3 for visual comparisons. From the figure, we can see that, although the depth initialization has no significant impact on visual quality, it is beneficial for improving geometric quality. The main reason is that the depth initialization can ease the learning of complex scene geometries for our Pixel Decorator that built upon pixel-based representation. Besides, with the collaboration of volume-based representation, our full method significantly surpasses the two variants with only pixel-based representations both visually and geometrically, further demonstrating the advantage of our Omni-Gaussian representation.

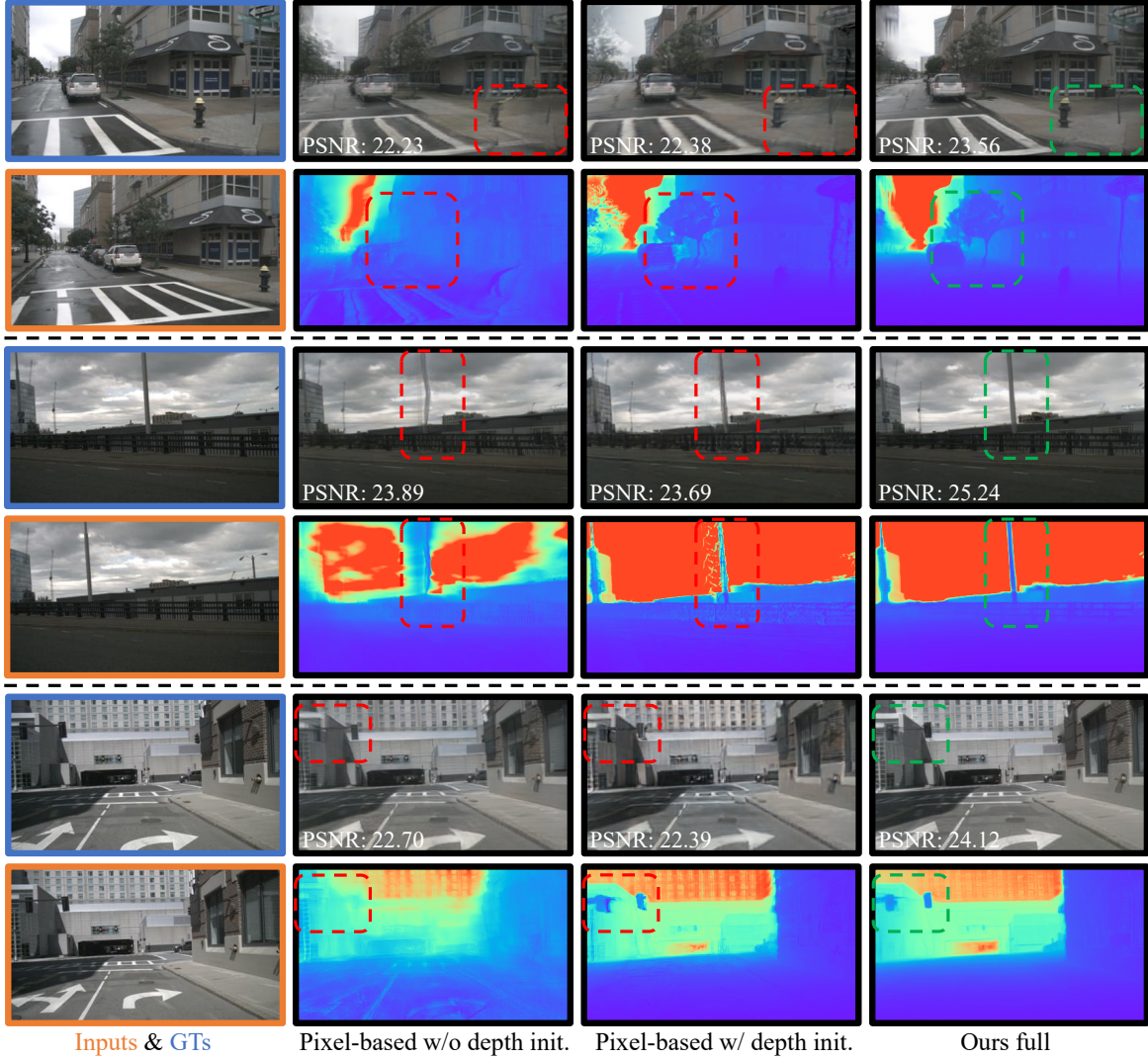


Figure 3. Qualitative ablations on Depth Initialization. The 1st column present images of input views (Inputs) and ground-truth novel views (GTs). The 2nd and the 3rd columns are results generated by two variant models with only pixel-based representation (i.e., Pixel Decorator), one with the depth initialization and one without. The last column denote results generated by our full method. The *red dashed lines* highlight undesirable artifacts, while the *green ones* denote plausibly-rendered areas. PSNR values are shown for better comparison.

Ablations on Deformable Attentions. As described in Sec.3.1 of our main manuscript, we employ cross-image and cross-plane deformable attentions in our Volume Builder to enhance volumetric feature encoding. Given camera parameters (i.e., intrinsics and extrinsics) that enable 3D-to-2D projection, our cross-image deformable attention module can lift 2D features to the 3D volume space, which enables the prediction of 3D Gaussians directly at the 3D level. This differs from previous methods [3, 5] that require cross-view overlap to estimate per-pixel depths and predict 3D Gaussians at the 2D level. To further address the issue that some elements in 3D might be occluded or truncated for any of the 2D input views, we utilize our cross-plane deformable attention to enhance each triplane query with cross-plane context, which means information absent in one plane can be complemented by those from

other planes at the 3D level. To validate the effectiveness of such dual-path design, we train three Volume Builder models, where one contains both of the cross-image and cross-plane attentions, while the other two contain only one of the attentions. As demonstrated in Table 3 and Fig. 4, the model with both attentions significantly outperforms the other two variants, showing the importance of such dual-path feature encoding to our Volume Builder. We further compare these volume-only variants with our full method

cross-image	cross-plane	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PCC \uparrow
\times	\checkmark	14.29	0.428	0.578	0.539
\checkmark	\times	21.29	0.595	0.412	0.686
\checkmark	\checkmark	22.21	0.640	0.357	0.701

Table 3. Ablations on cross-image & plane deformable attentions.

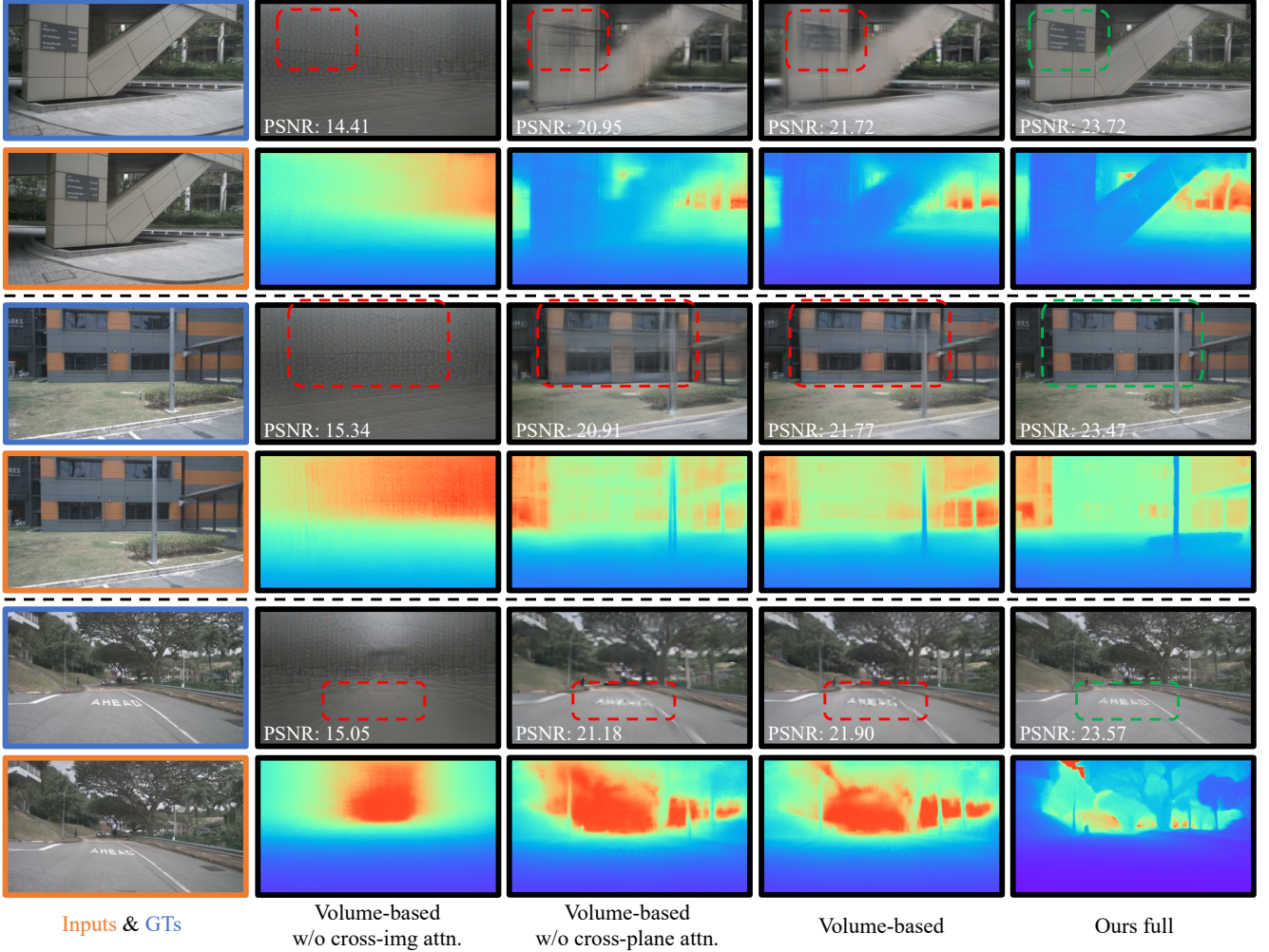


Figure 4. Qualitative ablations on deformable attentions. The 1st column present images of input views (Inputs) and ground-truth novel views (GTs). The 2nd to 4th columns are results generated by three variant models with only volume-based representation (i.e., Volume Builder), one without the cross-image deformable attention (“cross-img attn.”), one without the cross-plane deformable attention (“cross-plane attn.”), and one with both of the attentions. The last column denote results generated by our full method. The *red dashed lines* highlight undesirable artifacts, while the *green ones* denote plausibly-rendered areas. PSNR values are shown for better comparison.

in Fig. 4. It’s observed that results generated by our full method are with better details, showing the effectiveness of our Omni-Gaussian representation.

2.5. Generalizability to Different Bin Sizes

Unless otherwise specified, our experiments are conducted with a bin size of 3.2m as stated in Sec.1.1. To validate whether our method can be generalized to synthesize novel views at farther or closer distances, we preprocess nuScenes [1] into three variants with different bin sizes (i.e., 1.6m, 6.4m, 12.8m) from our original dataset. Here we note that, the larger the bin size, the farther distance between the novel and the input views, which is more challenging for novel view synthesis. Practically, for each dataset variant, we employ two approaches to test our model: (1) The model

trained under a bin size of 3.2m is directly used for evaluation without additional fine-tuning. (2) The model is further fine-tuned with the new bin size for 50,000 steps before evaluation. As can be seen from the 3rd, 5th and 7th rows of Table 4, despite the lack of supervision, our method exhibits minor degradation in performance for novel view synthesis at farther distances. For instance, we observe only 1.38 dB drop of PSNR, and 0.009 drop of PCC for “bin size = 6.4m”, which denotes distances $2\times$ farther than those seen during training. As can be seen from the 4th, 6th, and 8th rows of Table 4, by fine-tuning the model on data renewed with different bin sizes, we can further boost the performance and bring novel view synthesis at farther distances (i.e., “bin size = 6.4m, 12.8m”) very close to the results obtained under the original setting of “bin size = 3.2m”.

bin size	fine-tuning	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PCC \uparrow
3.2m	–	24.27	0.736	0.237	0.800
1.6m	\times	25.12 \pm 0.85	0.771 \pm 0.035	0.208 \pm 0.030	0.804 \pm 0.004
	\checkmark	25.37 \pm 1.10	0.783 \pm 0.047	0.201 \pm 0.037	0.806 \pm 0.006
6.4m	\times	22.89 \pm 1.38	0.682 \pm 0.054	0.287 \pm 0.050	0.791 \pm 0.009
	\checkmark	24.15 \pm 0.12	0.729 \pm 0.007	0.239 \pm 0.002	0.797 \pm 0.003
12.8m	\times	21.57 \pm 2.70	0.640 \pm 0.096	0.346 \pm 0.109	0.771 \pm 0.029
	\checkmark	23.55 \pm 0.72	0.711 \pm 0.025	0.265 \pm 0.028	0.792 \pm 0.008

Table 4. Results of our method when generalized to different bin sizes with or without additional fine-tuning.

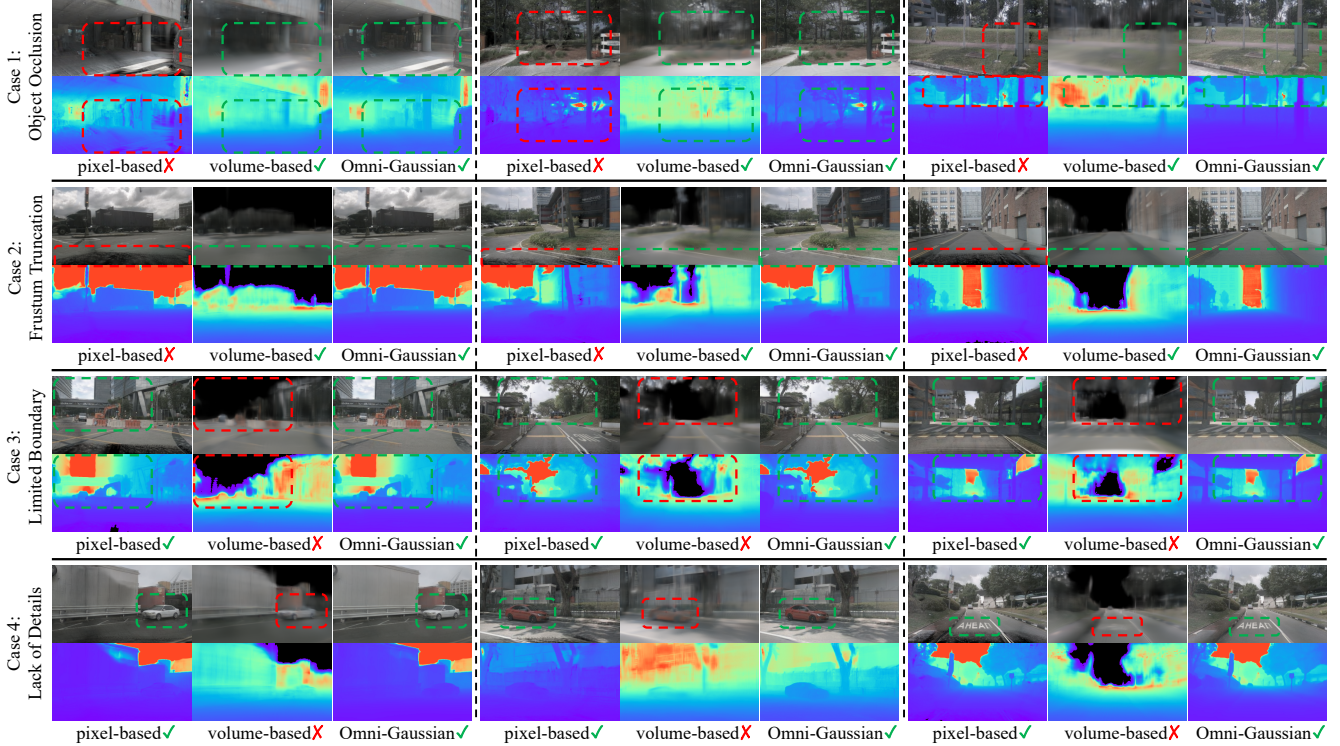


Figure 5. Additional examples of Volume-Pixel Collaboration. The red dashed lines highlight artifacts caused by weaknesses of singular representations, while the green ones outline how the artifacts are eliminated through Volume-Pixel Collaboration.

2.6. Discussion on Volume-Pixel Collaboration

In Fig.2 of our main manuscript, we have showcased pros and cons of the pixel-based and the volume-based Gaussian representations, and have provided the corresponding examples to illustrate *how the two representations complement for each other in our unified model with the proposed Omni-Gaussian representation*. Here, we present more examples in Fig.5 to demonstrate the effectiveness of their collaboration case by case:

- In “Case 1” of Fig.5, when objects in the novel view are occluded in the input views, pixel-based representation focuses on the non-occluded areas, with the occluded parts supplemented by volume-based representation.
- In “Case 2” of Fig.5, when objects in the novel view fall beyond the frustum range for any of the input views, pixel-based representation focuses on the non-truncated areas, with the truncated parts supplemented by volume-

based representation.

- In “Case 3” of Fig.5, for distant elements out of the volume range, volume-based representation concentrates on reconstruction within the volume, leaving the reconstruction of distant elements to pixel-based representation.
- In “Case 4” of Fig.5, for objects with fine-grained details (e.g., cars, lane markings), volume-based representation aims to predict their coarse 3D structures, leaving the surface details to pixel-based representation.

2.7. Additional Comparisons on RealEstate10K

As shown in Fig.6, we present more qualitative comparisons with state-of-the-art methods pixelSplat [3] and MVSplat [5] on RealEstate10K [14], which is a large-scale dataset for scene-centric reconstruction task. We can see that our method can render novel view images and depths with comparable and even superior quality to other methods.

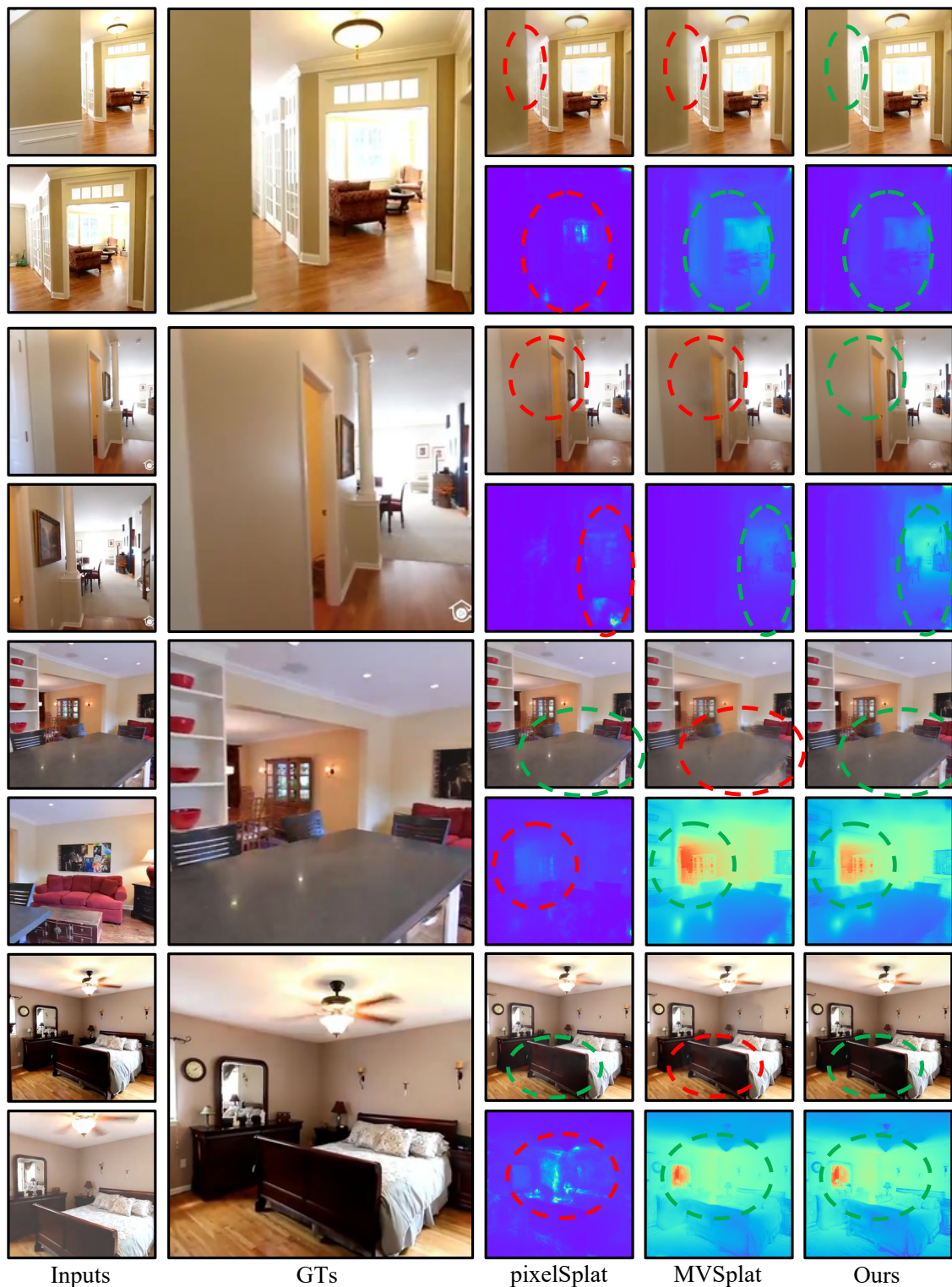


Figure 6. Additional qualitative results on scene-centric reconstruction performed on RealEstate10K [14]. The first two columns are images of input views and ground-truth novel views. The remaining three columns are results generated by pixelSplat [3], MVSplat [5] and our method, respectively. The *red dashed lines* highlight undesirable artifacts, while the *green ones* denote plausibly-rendered areas.

References

- [1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multi-modal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. [1](#), [2](#), [5](#)
- [2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. [1](#)
- [3] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent 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*, pages 19457–19467, 2024. [2](#), [4](#), [6](#), [7](#)
- [4] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart- σ : Weak-to-strong training of diffusion transformer for 4k text-to-image generation. *arXiv preprint arXiv:2403.04692*, 2024. [1](#)
- [5] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. *arXiv preprint arXiv:2403.14627*, 2024. [2](#), [4](#), [6](#), [7](#)
- [6] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4970–4980, 2023. [2](#)
- [7] Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhenguo Li, and Qiang Xu. Magicdrive3d: Controllable 3d generation for any-view rendering in street scenes. *arXiv preprint arXiv:2405.14475*, 2024. [2](#)
- [8] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. [2](#)
- [9] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [1](#)
- [10] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. [1](#)
- [11] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. [2](#)
- [12] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8269–8279, 2022. [2](#)
- [13] Haofei Xu, Anpei Chen, Yuedong Chen, Christos Sakaridis, Yulun Zhang, Marc Pollefeys, Andreas Geiger, and Fisher Yu. Murf: Multi-baseline radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20041–20050, 2024. [2](#)
- [14] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. [6](#), [7](#)