

Supplementary Materials: A Novel High-Frequency Surface Shell Radiance Field to Improve Large-Scale Scene Rendering

Anonymous Authors

1 OVERVIEW

Here, we give more detailed information to supplement the main paper. In section 2, we introduce how to calculate the sampling interval (*near* to *far*) of a ray and the bounding box of voxel grids. *Near* and *far* are the distances from the camera entering and leaving the target area along the ray. In section 3, we present the convolutional neural network (CNN) in the post-processing. In sections 4 and 5, configurations of all models, data partitioning, and training settings are described. In section 6, additional renderings of all NeRFs are shown. In section 7, HS-Surf is compared with 3D Gaussian [1]. In section 8, we present additional ablation experiments, including analysis for the feature fusion in initialization and the compensation stage. In section 9, we discuss the limitations of HS-Surf.

Due to the size limitation on supplementary materials, we select two video clips from our video demos to showcase the rendering results. The scene of video1 is *transamerica* [2], and the distance scales undergo drastic changes. The scene of video2 is *building* [3], and the training images are captured by a drone at a stable altitude.

2 NEAR, FAR AND BOUNDING BOX

The target areas are divided into earth and plane types. The calculations of the two types are different. Earth type includes *Transamerica* and *56Leonard* [2]. Because the height of the camera varies from satellite level to ground level, the fact that Earth is a sphere needs to be considered. Plane type includes *building*, *rubble*, *residence*, and *campus* [3, 4]. The drone shoots towards the ground and maintains a stable flight altitude. Therefore, the target area is between the flight plane and the ground.

For the earth type in this work, we reconstruct the area within 250m of the ground, which is between the orange and blue spheres in Figure 1. The distances from the camera to the two spheres along the ray are defined as *near* and *far*. The samples on the ray are located between *near* and *far*.

In Figure 1, G is the coordinate of earth center, and O is the position of camera. dir is the ray direction and is a unit vector. \vec{a} is a vector from O to G , and \vec{b} is a vector from O to the orange or blue sphere. For the blue sphere sitting on the ground, $\|\vec{a} - \vec{b}\|_2 = r$, where $r = 6371011m$ is the radius of earth. For the orange sphere in the sky, $\|\vec{a} - \vec{b}\|_2 = r + 250$. The equations are as follows:

$$\begin{cases} \|G - O - far \cdot dir\|_2 = r \\ \|G - O - near \cdot dir\|_2 = r + 250. \end{cases} \quad (1)$$

The intermediate variables are as follows:

$$\begin{cases} a = \|dir\|_2^2 \\ b = 2 \cdot dir \cdot (O - G) \\ c_{far} = \|G - O\|_2^2 - r^2 \\ c_{near} = \|G - O\|_2^2 - (r + 250)^2, \end{cases} \quad (2)$$

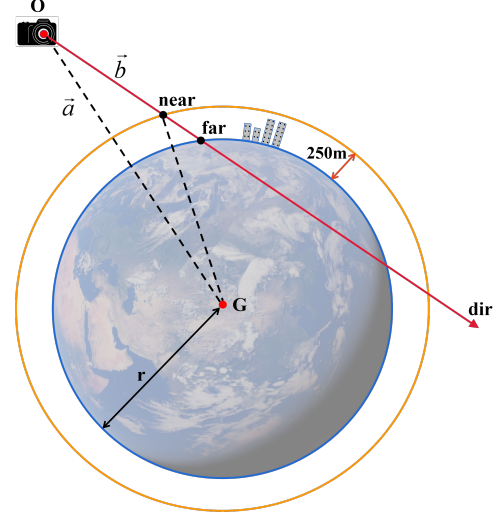


Figure 1: Illustration of the earth type, we reconstruct the area within 250m from the ground, which is located between the orange and blue spheres.

and *near* and *far* are calculated as follows:

$$\begin{cases} far = (-b - \sqrt{b^2 - 4a \cdot c_{far}})/(2a) \\ near = (-b - \sqrt{b^2 - 4a \cdot c_{near}})/(2a). \end{cases} \quad (3)$$

After determining *near* and *far* of each ray, two boundary points can be calculated:

$$\begin{cases} p_n = O + near \cdot dir \\ p_f = O + far \cdot dir. \end{cases} \quad (4)$$

The maximum and minimum values of all boundary points in x , y , z determine the extent of the bounding box. To reduce computation, we only use rays located at the four corners of each image.

For the plane type, the Z-axis of the world coordinate system in the aerial data is perpendicular to the ground, and the range of target area on the Z-axis is also specified, as shown in Figure 2. The component of camera position on the Z-axis is z_0 . The upper and lower surfaces of the target area correspond to z_1 and z_2 . The distances from camera to the upper and lower surfaces along the ray direction dir are *near* and *far*, and dir is a unit vector. Therefore, *near* and *far* are calculated as follows:

$$\begin{cases} near = \max(z_0 - z_1, 0)/|dir(z)| \\ far = (z_0 - z_2)/|dir(z)|, \end{cases} \quad (5)$$

where $dir(z)$ represents the component of dir on the Z-axis. For the cameras located inside the scene, *near* = 0. After determining *near* and *far* of each ray, the corresponding bounding box of the scene can be obtained, and the process is the same as the earth type.

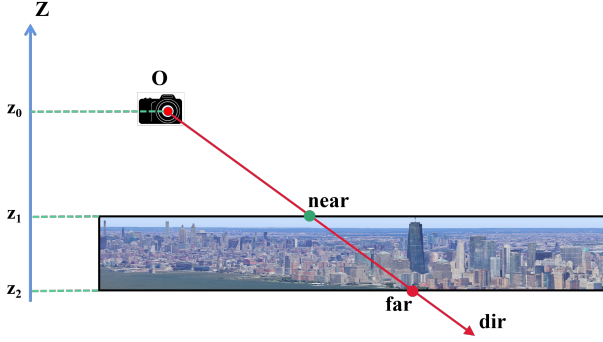


Figure 2: Illustration of the plane type. Z-axis of the world coordinate system is perpendicular to the ground. The projections of upper and lower surfaces of target area on the Z-axis are known.

3 POST PROCESSING

The CNN structure in the post-processing stage is shown in Figure 3, which is a lightweight network containing residual blocks.

4 CONFIGURATION OF COMPARED MODELS

MipNeRF [5] uses a default 8-layer MLP with 256 hidden nodes. The sizes of voxel grids and hash tables in ZipNeRF [6] are the same as those in HS-Surf, and the hidden nodes of the small MLPs are set to 256. In *transamerica* and *56Leonard* [2], the size of hash table is $2^{21} \times 4$. In *building*, *rubble*, *residence*, and *campus* [3, 4], the size of hash table is $2^{22} \times 4$. BungeeNeRF [2] is divided into four stages according to the original paper. The first stage is a four-layer MLP with 256 hidden nodes, and each subsequent stage adds two fully connected layers.

For MegaNeRF [3], we directly use the Python code and checkpoint provided by the authors. GridNeRF [7] contains three ground feature planes with resolutions of 256^2 , 768^2 , and 2304^2 . The dimensions of the density and color features are set to 16 and 48 in each plane. The hidden nodes of MLPs in the grid and NeRF branches are set to 256. GridNeRF with the above configuration can just run on a 3090 GPU. For 3D Gaussian [1], we use the CUDA code and configuration provided in the paper for training, with the number of images in iteration is set to 90000.

5 DATA PARTITIONING AND TRAINING SETTINGS

As shown in Table 1, images in the six scenarios have different resolutions, and 20% of the data is extracted to test all models. For HS-Surf, the batch size of pixels and images are 2048 and 2 in training. For the other models, the batch size of pixels is set to 2048. The training settings of all models are shown in Table 2. Because *campus* has more data than the other scenarios, we appropriately increase the training steps.

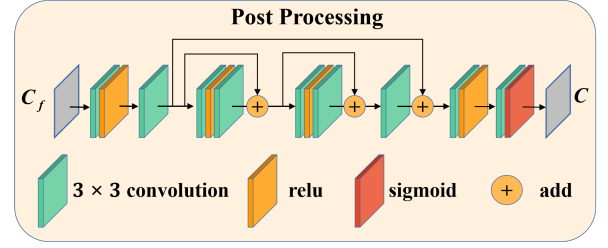


Figure 3: Detailed structure of CNN in the post-processing.

Table 1: Number of images and details of the six scenes

scene	train	test	resolution	object center
Transamerica [2]	364	91	1080×1920	✓
56 Leonard [2]	371	92	1080×1920	✓
Building [3]	1552	388	1152×1536	×
Rubble [3]	1343	335	1152×1536	×
Residence [4]	2066	516	1216×1824	×
Campus [4]	4697	1174	1216×1824	×

Table 2: Iteration steps of the models in all scenarios

<i>Transamerica, 56Leonard</i>	
MipNeRF [5]	500k
ZipNeRF [6]	500k
BungeeNeRF [2]	200k per stage (4 stages)
GridNeRF [7]	100k and 400k in the two stages
HS-Surf	450k and 50k in the two stages
<i>Building, rubble, residence</i>	
MipNeRF [5]	1200k
ZipNeRF [6]	1200k
MegaNeRF [3]	Load the model directly
GridNeRF [7]	200k and 1000k in the two stages
HS-Surf	1100k and 100k in the two stages
<i>Campus</i>	
MipNeRF [5]	1500k
ZipNeRF [6]	1500k
MegaNeRF [3]	Load the model directly
GridNeRF [7]	300k and 1200k in the two stages
HS-Surf	1400k and 100k in the two stages

6 MORE RESULTS WITH NERFS

The proposed HS-Surf is compared with MipNeRF [5], ZipNeRF [6], BungeeNeRF [2], MegaNeRF [3], and GridNeRF [7]. Rendering results on *transamerica* and *56Leonard* are shown in Figure 4, where the distance scales undergo drastic changes, from the satellite level to the ground level. Rendering results on the real aerial data (*building*, *rubble*, *residence*, *campus*) are shown in Figures 5 and 6. The drone maintains a stable flight altitude and shoots toward the ground along parallel lines or grid trajectories. Thus, the data distribution is uniform.

Different from the other models, HS-Surf constructs a high-frequency shell based on the current view's scene depth to maximize the utilization of model capacity, efficiently increasing the model capacity on the scene surface for rendering high-frequency textures.

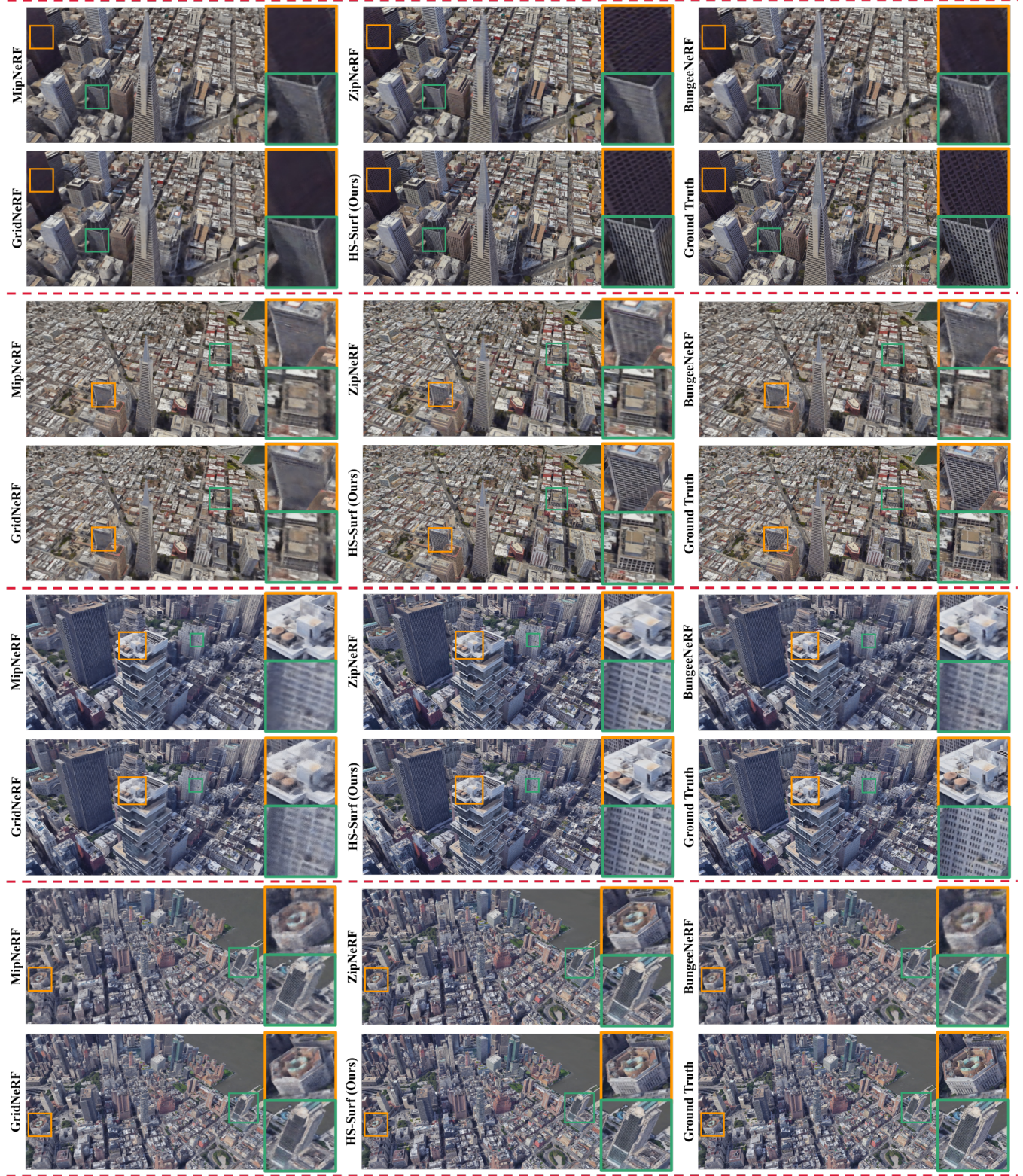


Figure 4: In *transamerica* and *56Leonard*, HS-Surf preserves more high-frequency texture details at multiple distance scales with drastic changes, such as windows and grids on buildings.



Figure 5: Rendering results in *building* and *rubble*. HS-Surf produces finer scene structures, such as iron fences of the factory, as well as textures on railways, etc.

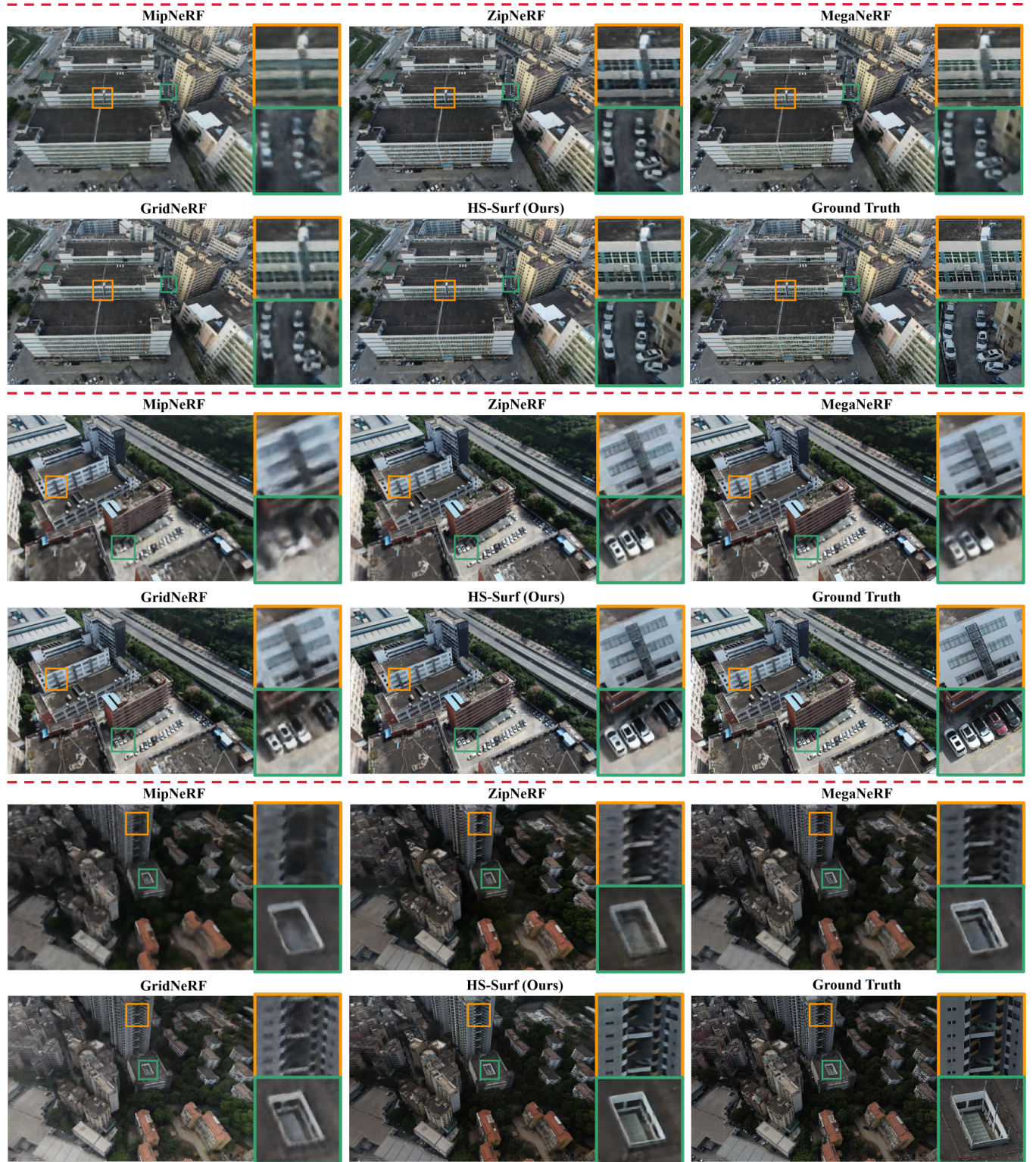


Figure 6: In *residence* and *campus*, HS-Surf renders more texture details in aerial large-scale scenes, including intricate and complex windows, vehicles on the roads, and architectural details on buildings.

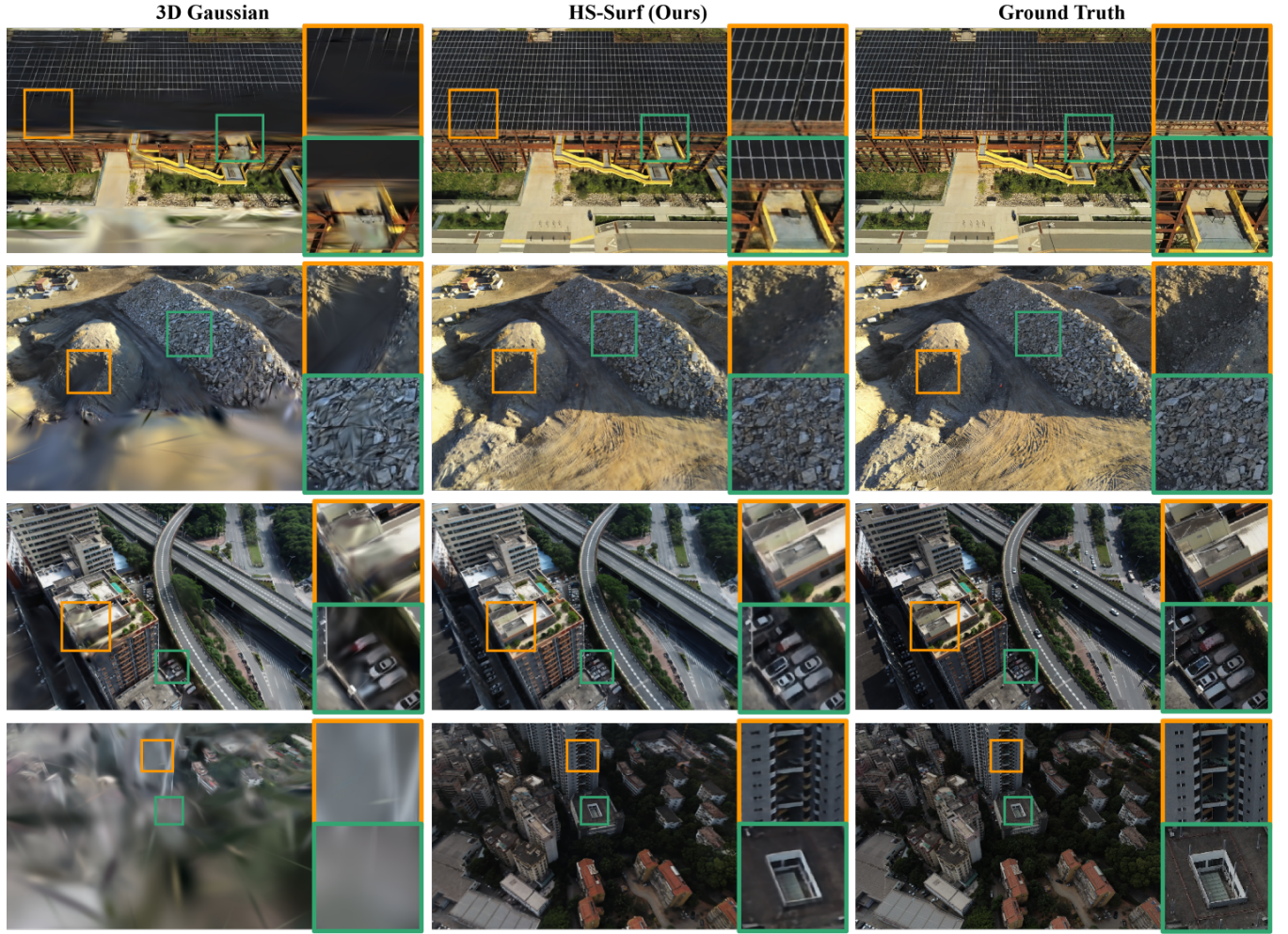


Figure 7: Compared to HS-Surf, 3D Gaussian tends to lose texture details on fine scene structures in large-scale scenes, and their rendering results often contain many continuous blurry areas.

Table 3: Comparison of HS-Surf and 3D Gaussian

Model	Building			Rubble			Residence			Campus		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
3D Gaussian	18.05	0.5388	0.5418	20.49	0.5795	0.5390	19.72	0.5953	0.4917	14.94	0.3154	0.8377
HS-Surf	21.88	0.6039	0.4417	24.24	0.5824	0.4943	22.12	0.5982	0.5015	21.97	0.4639	0.6183

Additionally, our feature fusion strategy embeds frustums representing distance into voxel grids to generate the target scene at different distance scales. Therefore, HS-Surf produces high-fidelity renderings across various distances and renders more texture details in the aerial scenes.

7 COMPARISON WITH 3D GAUSSIAN

HS-Surf is compared with 3D Gaussian [1] on the aerial scenes (*building, rubble, residence, campus*), and performance metrics and rendering results are presented in Table 3 and Figure 7. In Figure 7,

3D Gaussian tends to lose texture information when rendering small objects. The Gaussian spheres may struggle to split into sufficiently small sizes to represent texture details in large-scale scenes. Additionally, 3D Gaussian generates large continuous blurry areas in their rendering results. The possible reason is the sparse coverage of views in the training set, which is not conducive to rendering novel views in the test set.

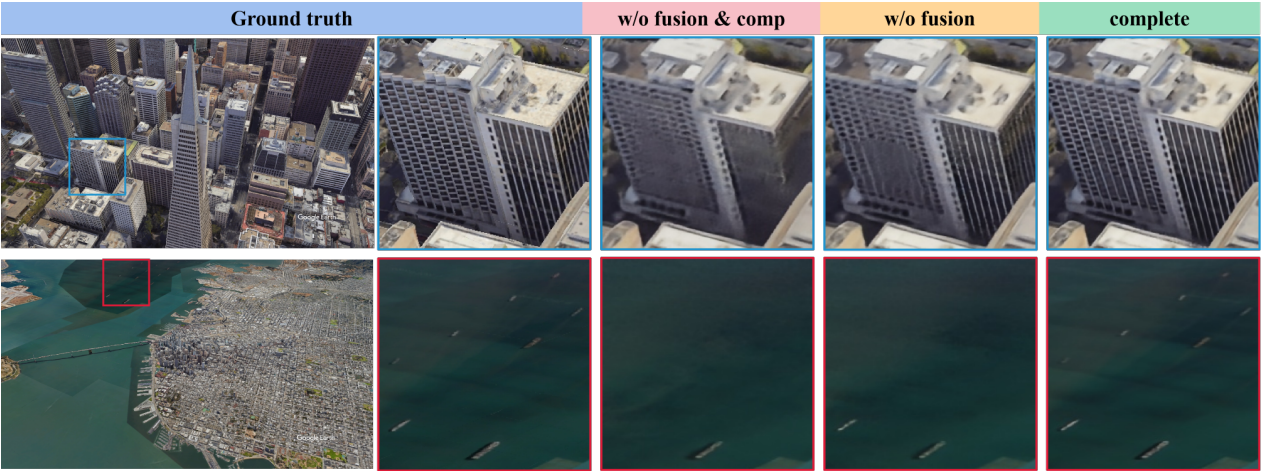


Figure 8: Removing the feature fusion (fusion) in initialization results in a loss of texture details across various distances. The compensation (comp) can complete lots of high-frequency textures, but the performance is still lower than the complete model.

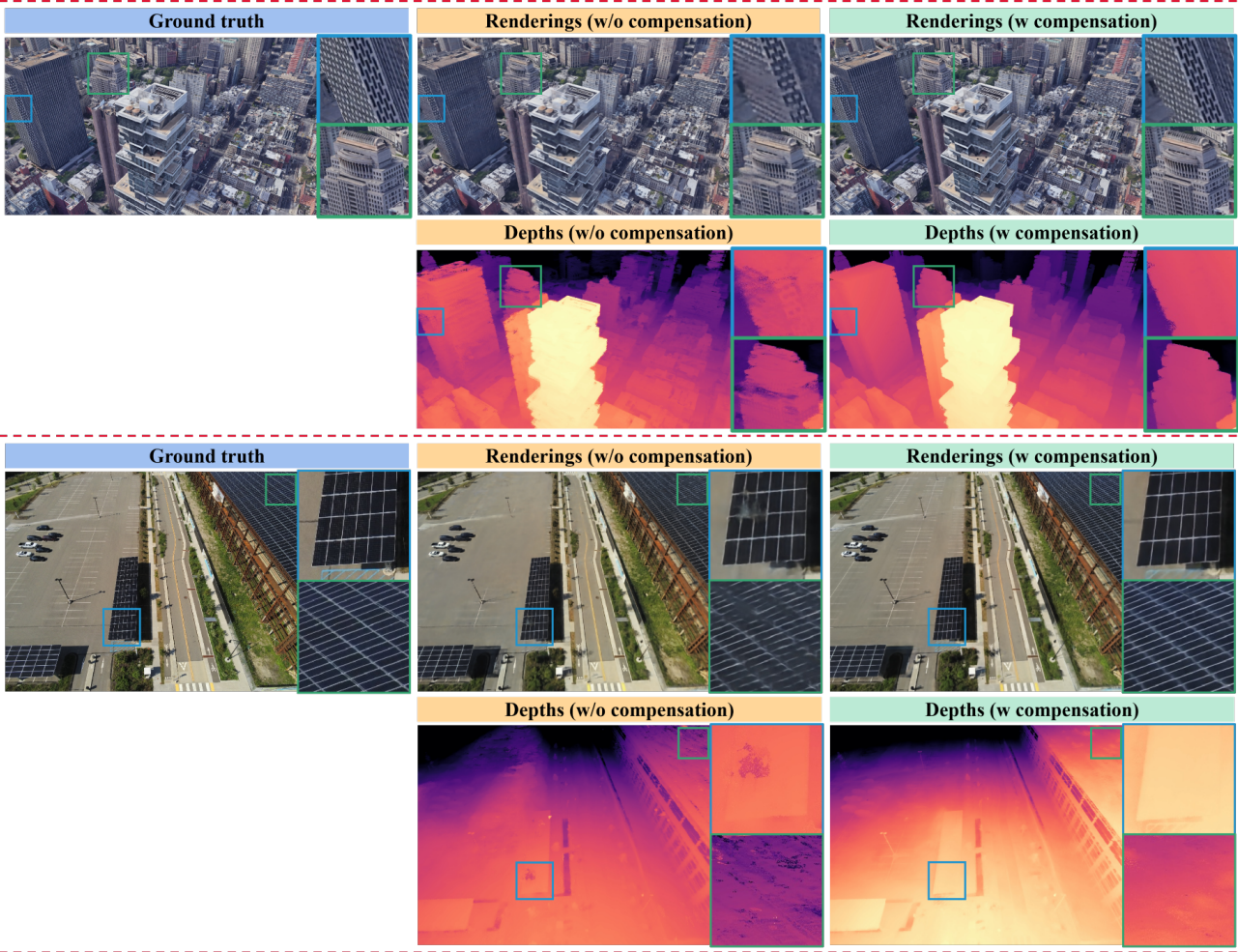


Figure 9: When the compensation stage is removed, the model produces unsmooth scene geometry and a large number of holes. Simultaneously, the corresponding rendering results also lose texture information.

Table 4: Ablation experiments of the feature fusion (fusion) and the compensation stage (comp)

Transamerica model	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
A) w/o fusion & comp	24.31	0.7707	0.3526
B) w/o fusion	24.50	0.7814	0.3383
C) complete	25.59	0.8304	0.2941
Building model	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
A) w/o fusion & comp	20.97	0.5290	0.5025
B) w/o fusion	21.35	0.5567	0.4788
C) complete	21.88	0.6039	0.4417

8 MORE ABLATION EXPERIMENTS

We additionally analyze the benefits brought by the combination of feature fusion strategy and compensation stage, and conduct relevant ablation experiments on *transamerica* and *building*, as shown in Table 4 and Figure 8. The findings are summarized here. A) When both the feature fusion and the compensation stage are removed, the model produces blurry rendering results and loses a lot of high-frequency textures at different distance scales. B) When only the feature fusion is removed, the compensation stage can fill in many missing texture details, but the results are still different from the ground truth. C) The combination of feature fusion and compensation stages brings the maximum benefit, filling in high-frequency textures at different distance scales.

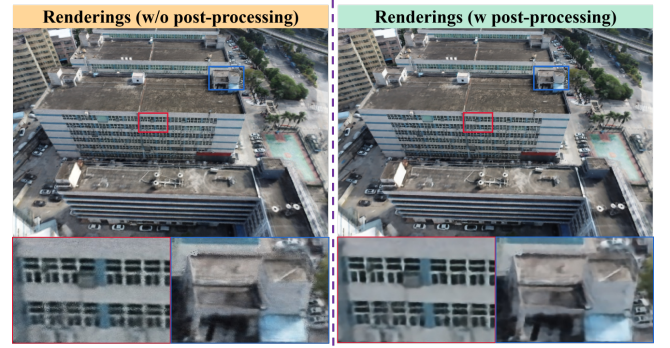
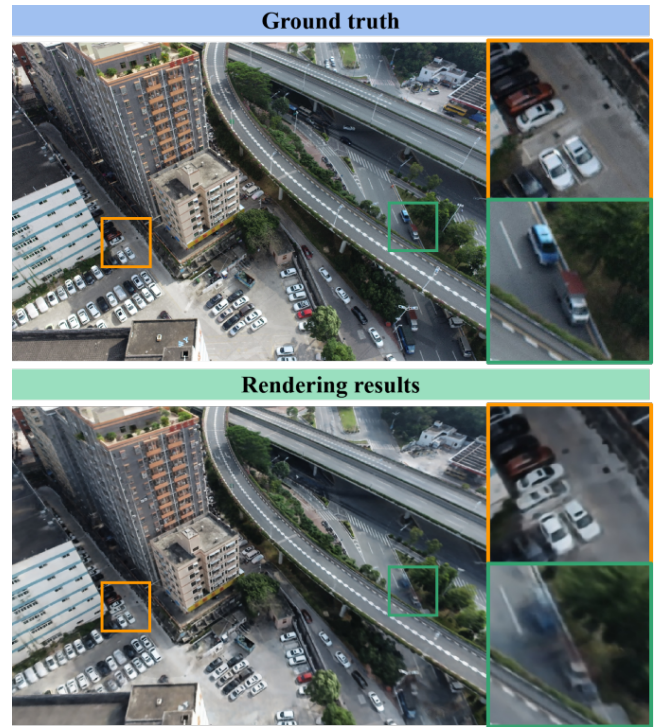
The purpose of the compensation stage is to refine the scene geometry and fill in the missing high-frequency textures. Figure 9 shows more ablation experiments of this stage. If the compensation is removed, the scene geometry is not smooth and contains a large number of holes due to the lack of depth augmentation. Simultaneously, the rendering results are blurry, and the texture information is lost. The role of the post-processing stage is to remove noise in the renderings and smooth the image, as shown in Figure 10.

9 LIMITATIONS

Although HS-Surf achieves state-of-the-art rendering across various distance scales for large-scale scenes, there are several limitations that represent opportunities for future improvement: 1) Frequent movement of dynamic objects in the scene leads to artifacts in the rendering results, as shown in Figure 11. HS-Surf uses photometric loss to supervise the training process. When dynamic objects appear at a certain location in the scene, the model needs to generate artifacts to minimize the loss function. 2) Shadows in images might lead to collapse geometry, because the reconstruction process of geometry does not introduce lighting information.

REFERENCES

- [1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.
- [2] Yuanbo Xiangli, Linning XU, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. *Eur. Conf. Comput. Vis.*, 2022.

**Figure 10: The post-processing stage is used to smooth and denoise the rendered results.****Figure 11: Due to the large number of dynamic objects in large-scale scenes, HS-Surf may generate artifacts.**

- [3] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12912–12921, 2022.
- [4] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: the urbanscene3d dataset. *Eur. Conf. Comput. Vis.*, 2022.
- [5] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *Int. Conf. Comput. Vis.*, 2021.
- [6] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *Int. Conf. Comput. Vis.*, 2023.
- [7] Linning Xu, Yuanbo Xiangli, Sida Peng, Xingang Pan, Nanxuan Zhao, Christian Theobalt, Bo Dai, and Dahua Lin. Grid-guided neural radiance fields for large urban scenes. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8296–8306, 2023.