# Robust and Efficient 3D Gaussian Splatting for Urban Scene Reconstruction

## Supplementary Material

## A. Our Datasets

The *JNU-ZH* and *BigCity* scenes were collected by our team using drones, and their contents are shown in Figure 7. We employ COLMAP's hierarchical SfM [36] to perform sparse reconstruction for both scenes. After finishing reconstruction, we use COLMAP's geo-registration to align the reconstructed model with GPS coordinates. Subsequently, we compute the Euclidean distance between the estimated camera positions and their corresponding GPS coordinates. Outliers with excessively large distances are discarded, as they typically result from inaccurate pose estimations. This filtering process helps mitigate the negative impact of erroneous data. Finally, we downsample the images to a maximum edge length of 1600 pixels for experimentation. When partitioning, the sizes used for these two scenes are 180m and 400m, respectively.

## B. Implementation Details of Our Method

We implemented our method based on gsplat [50], which offers higher computational and memory efficiency compared to the [8].

The visibility threshold for dataset division is $1/6$. We use three detail levels for all scenes. The first and second levels each last a base of 15,000 iterations, with densification enabled. The third level runs for a base of 30,000 iterations, where densification is applied in the first half and the second half is solely dedicated to optimizing properties. Table 6 presents the hyperparameters for detail level generation in different scenes.

| Scenes | $(B_1, B_2, B_3) \times 100$ | $(T_1, T_2, T_3)$ | $(D_1, D_2, D_3)$ |
|---|---|---|---|
| *Rubble* | $(4096, 8192, 16384)$ | | $(1/2, 1/3, 1)$ |
| *JNU-ZH* | $(4096, 8192, 20480)$ | $(300, 200, 100)$ | $(1/4, 1/2, 1)$ |
| *BigCity* | $(2048, 8192, 20480)$ | | $(1/4, 1/2, 1)$ |

Table 6. **The hyperparameters for the detail level generation.**

In practice, these iteration counts and densification interval $T$ are adjusted proportionally based on the number of images $N$ in each partition, scaled by a factor of $\max(N/600, 1)$.

In the appearance transform model, the MLP consist of 1 hidden layers with 32 channels, followed by a ReLU activation. The output layer followed by a sigmoid activation. The Gaussian embeddings is 16-dimensional, while the image embedding is 32-dimensional. The initial learning rate for the MLP and embeddings is set to 0.01, and an exponential decay scheduler reduces it to a final value of 0.00025. Every 50 iterations, we sample 20,480 Gaussians and select $k = 16$ nearest neighbors to perform similarity regularization, minimizing the computational overhead.

In the scale regularization, the value of $s_{\text{max}}$ is set to a value corresponding to the typical size of most buildings in the scene, and $r_{\text{max}} = 10$.

In the in-partition prioritized densification, The value of $\hat{d}_{\text{max}}$ is identical to the partition size. The maximum gradient threshold factor is $\eta = 4$. The minimum threshold is $\tau_{\text{min}} = 0.0002$ for the 1st and 2nd levels, and is $0.6$ for the 3rd level with AbsGS [51] enabled.

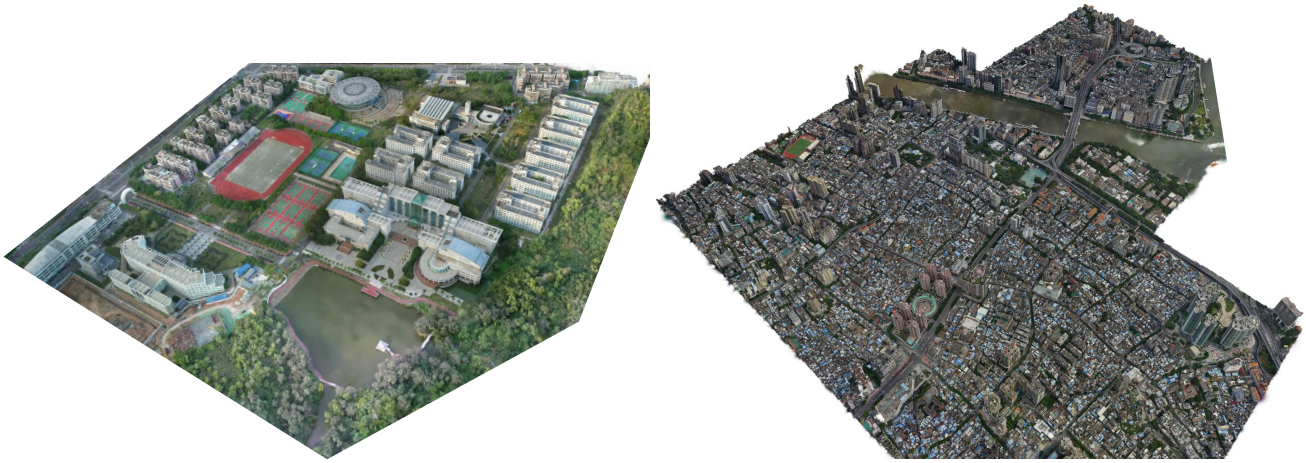
## C. Hyperparameters of Other Methods

For the 3DGS, large-scale scenes generally require more iterations for sufficient optimization. Therefore, the training process was extended to 50 epochs, with densification enabled during the first 25 epochs. We also set the densification interval to $1/6$ of an epoch, ensuring a consistent number of densifications across all scenes. When the number of input images is 600, these adjustments yield consistent hyperparameter with the original settings.

For Switch-NeRF, we utilized the official open-source implementation with its provided hyperparameters. When conducting experiments on our own scenes, we proportionally increased the number of training iterations based on the number of input images.

For the remaining methods, we utilized their official open-source implementations and use a similar number of partitions to reconstruct all the scenes. When evaluating the LOD mode of Hierarchical-3DGS, we used a granularity setting value of 6 pixels.

Due to the large scale and intricate details of the *BigCity* scene, none of the previous 3DGS-based methods can complete the experiment within an 80GB memory limit. Therefore, we made additional adjustments to the hyperparameters for these methods. For 3DGS, we double the densification gradient threshold. For CityGaussian, during the coarse training stage, we tripled the densification cycle and doubled the densification gradient threshold compared to the original settings. During the pruning stage, we increased the pruning ratios from the default 40%, 50%, and 60% to 70%, 80%, and 90%. For Hierarchical-3DGS, the excessive number of Gaussians made it impossible to evaluate the non-LOD mode. When evaluating its metrics under the LOD mode, we doubled the granularity settings from 6 pixels to 12 pixels. In contrast, our method can complete all steps, except for the non-LOD mode, with memory usage not exceeding 24GB.



(a) *JNU-ZH* (b) *BigCity*

Figure 7. **Our datasets: *JNU-ZH* and *BigCity*.**

# D. Appearance Transform Module

## D.1. Metric Calculation

Given that we have the appearance transform model, which optimizes only the embeddings of training set images, we followed a strategy similar to NeRF-W [21] to evaluate the test set images. Specifically, when computing the metrics for test images, we first optimize the image embedding $\ell^{(\mathcal{I})}$ using the left half of the image and compute the metrics using the right half. Each partition is transformed using the embedding of the test image optimized within that partition, ensuring appearance consistency. Then, we optimize the embedding from scratch using the right half and computed the metrics with the left half. Finally, the average of the results from both rounds was taken as the final metric value for the entire image. This approach prevents information leakage and ensures fairness in the evaluation process. In practice, we further smooth transitions between partitions via weighted averaging.

## D.2. Appearance Transformation

After reconstruction, our method enables scene appearance transformation. Using the image embedding $\ell^{(\mathcal{I})}$ of a training image, we can synthesize novel views that match its appearance. As shown in Figure 8, this enables transforming between different states of a building in the *JNU-ZH* scene.

# E. Additional Experiments

## E.1. Training Time Comparison

Table 7 presents the training times of all 3DGS-based methods. Except for 3DGS, all results were obtained under parallel training setups. The results show that our method is also competitive in terms of training efficiency, consistently ranking among the best or second-best. We are not consistently the fastest due to the additional overhead introduced by the Appearance Transform Module, anti-aliasing, and various regularization mechanisms. Nonetheless, maintaining such competitive efficiency despite these added components demonstrates the effectiveness of our optimization strategies.

It is worth noting that the *Rubble* and *JNU-ZH* are relatively small, where parallel training provides limited benefits. In contrast, the *BigCity* is significantly larger, and the parallel setup leads to a substantial speedup.

| Scenes | *Rubble* | *JNU-ZH* | *BigCity* |
|---|---|---|---|
| 3DGS | 1.45 | 3.21 | 67.39 |
| CityGaussian | 2.33 | 2.49 | 4.01 |
| Hierarchical-3DGS | **1.00** | <u>2.18</u> | **1.71** |
| Ours | <u>1.30</u> | **2.14** | <u>2.01</u> |

Table 7. **Comparison of training time (in hours).** Except for 3DGS, the results of all other methods were obtained under parallel training mode. VastGaussian is not included as it is not open-sourced.

## E.2. Additional Quantitative Comparison

Table 8 presents experimental results for the *Building* [42], *Residence*, *Sci-Art* and *Campus* [17] scenes. The camera poses are provided by Mega-NeRF. Overall, our method demonstrates a clear advantage in nearly all quality-related metrics. Although our method is not the most optimal in terms of resource consumption and rendering speed, it remains within a reasonable range and is close to the best-performing approach. It fully ensures real-time rendering. It is worth noting that our method can further reduce resource consumption by lowering the budget $B$. Figure 9

(a) Under maintenance, illuminated by the sunset. (b) Maintenance completed, illuminated by overcast light.

Figure 8. **Synthesis the two corresponding states from a new viewpoint based on the embedding vector provided by the reference image (bottom right).**

presents the visualization results for both scenes, demonstrating that our method achieves higher detail preservation and fewer artifacts.

### E.3. Quantitative Comparison of Detail Levels

Table 9 compares the performance of our three detail levels. The results show that all three levels achieve high reconstruction quality. The lower levels exhibit higher numerical values than the higher levels because they are trained and evaluated using downsampled images. Additionally, comparing the #G across levels further confirms that our LOD strategy effectively controls resource consumption.

### E.4. Additional ablations

Table 10 presents the results of ablation studies on the anti-aliasing, AbsGS, and tile-based culling components in our method.

**Anti-aliasing**. The 1th row of Table 10 reports the impact of anti-aliasing techniques. As shown in Figure 10a, it effectively prevents jagged edges from appearing in areas with low detail levels in images. However, this comes at the cost of requiring more Gaussians. In the *BigCity* scene, since the number of Gaussians has already reached the upper limit without anti-aliasing, enabling anti-aliasing does not allow for additional Gaussians, leading to a slight degradation in metrics. Nevertheless, this feature remains beneficial as it significantly enhances the visual experience.

**AbsGS**. The 2nd row of Table 10 provides the results obtained without AbsGS, highlighting its contribution to quality metrics and enhanced detail restoration, as shown in Figure 10b. While it does increase the number of Gaussians in certain scenarios, the increase remains within an acceptable range, ensuring that real-time rendering can still be achieved within the constraints of 24GB of VRAM.

**Tile-based culling**. The 3rd row of Table 10 presents the results without tile-based culling, illustrating its role in rendering efficiency. Tile-based culling noticeable improves rendering speed without negatively impacting rendering quality. This is because it skips over redundant Gaussians with minimal contribution, ensuring efficiency while maintaining the desired quality.

### E.5. Comparison of Gaussian Embedding Lengths

We evaluate the impact of the length of Gaussian embedding on the *JNU-ZH* scene. As shown in Table 11, reducing the length leads to a slight drop in metrics, but it is acceptable if the goal is to reduce memory consumption.

### E.6. Comparison of LOD Selection Parameters

We evaluate the impact of our LOD parameters on the *JNU-ZH* scene. Table 12 presents the impact of different rendering-time partition sizes on metrics. It can be observed that while smaller partition sizes effectively reduce the number of Gaussians and lower resource consumption, they also lead to a certain degree of metric degradation. Additionally, a larger number of partitions incurs higher overhead due to the LOD selection. In contrast, larger partition sizes exhibit the opposite behavior. This suggests that an optimal partition size must strike a balance between efficiency and rendering quality. Table 13 illustrates the impact of different distance thresholds for detail levels. Increasing the distance thresholds generally improves rendering quality but also leads to higher resource consumption and reduced rendering speed. Therefore, selecting an appropriate distance threshold requires a trade-off between efficiency and quality.

### E.7. Evaluation of Similarity Regularization

Similarity regularization enhances the generalization ability of the appearance transformation module. When performing out-of-domain inference, such as predicting the unobserved regions of an image using its embedding, this regularization effectively mitigates abrupt color changes and suppresses artifacts, as illustrated in the first row of Figure 11.

| Scene | *Building* | | | | | *Residence* | | | | | *Sci-Art* | | | | | *Campus* | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | SSIM | PSNR | LPIPS | #G | FPS | SSIM | PSNR | LPIPS | #G | FPS | SSIM | PSNR | LPIPS | #G | FPS | SSIM | PSNR | LPIPS | #G | FPS |
| Switch-NeRF | 0.579 | 21.54 | 0.474 | – | <0.1 | 0.654 | 22.57 | 0.457 | – | <0.1 | 0.795 | 26.52 | 0.360 | – | <0.1 | 0.541 | 23.62 | 0.609 | – | <0.1 |
| VastGaussian | 0.804 | 23.50 | – | – | – | **0.852** | 24.25 | – | – | – | **0.885** | 26.81 | – | – | – | **0.816** | 26.00 | – | – | – |
| CityGaussian (no LOD) | 0.784 | 21.96 | 0.243 | 13.30 | 37.6 | 0.813 | 22.00 | 0.211 | 10.80 | 41.0 | 0.837 | 21.39 | 0.230 | 3.80 | 82.3 | 0.666 | 19.61 | 0.403 | 16.41 | 35.1 |
| Hierarchical-3DGS (no LOD) | 0.720 | 20.55 | 0.270 | 14.79 | 36.4 | 0.753 | 19.85 | 0.230 | 13.68 | 39.9 | 0.792 | 19.85 | 0.273 | 9.13 | 31.5 | 0.741 | 22.66 | 0.297 | 29.32 | 13.9 |
| 3DGS | 0.787 | 22.42 | 0.282 | **13.02** | **64.2** | 0.807 | 21.96 | 0.256 | **7.06** | **102.2** | 0.833 | 21.26 | 0.285 | **2.28** | **172.4** | 0.718 | 19.83 | 0.370 | **10.55** | 18.5 |
| Ours (no LOD) | **0.808** | **24.12** | **0.219** | 18.22 | 62.5 | 0.845 | **24.93** | **0.201** | 14.15 | 71.5 | 0.876 | **27.78** | **0.190** | 8.39 | 86.9 | 0.788 | **26.63** | **0.278** | 42.42 | **38.5** |
| CityGaussian | 0.769 | 21.75 | 0.257 | **3.49** | **83.6** | 0.805 | 21.90 | **0.217** | **3.13** | 65.7 | 0.833 | 21.34 | 0.232 | **1.77** | **113.4** | N/A(encountered a bug) | | | | |
| Hierarchical-3DGS | 0.695 | 20.18 | 0.296 | 6.59 | 46.3 | 0.741 | 19.70 | 0.243 | 10.01 | 44.2 | 0.788 | 19.82 | 0.278 | 6.67 | 36.0 | 0.724 | 22.43 | 0.316 | 10.55 | 18.5 |
| Ours | **0.799** | **24.03** | **0.233** | 5.18 | 82.0 | **0.818** | **24.32** | 0.232 | 4.09 | **90.9** | **0.859** | **27.09** | **0.208** | 2.79 | 99.4 | **0.778** | **26.41** | **0.293** | **5.68** | **93.7** |

Table 8. **Quantitative evaluation on *Building*, *Residence*, *Sci-Art* and *Campus*.** The results for VastGaussian are only partially available as it is not open-sourced and can only be obtained from its paper. All missing results are denoted by a "–".

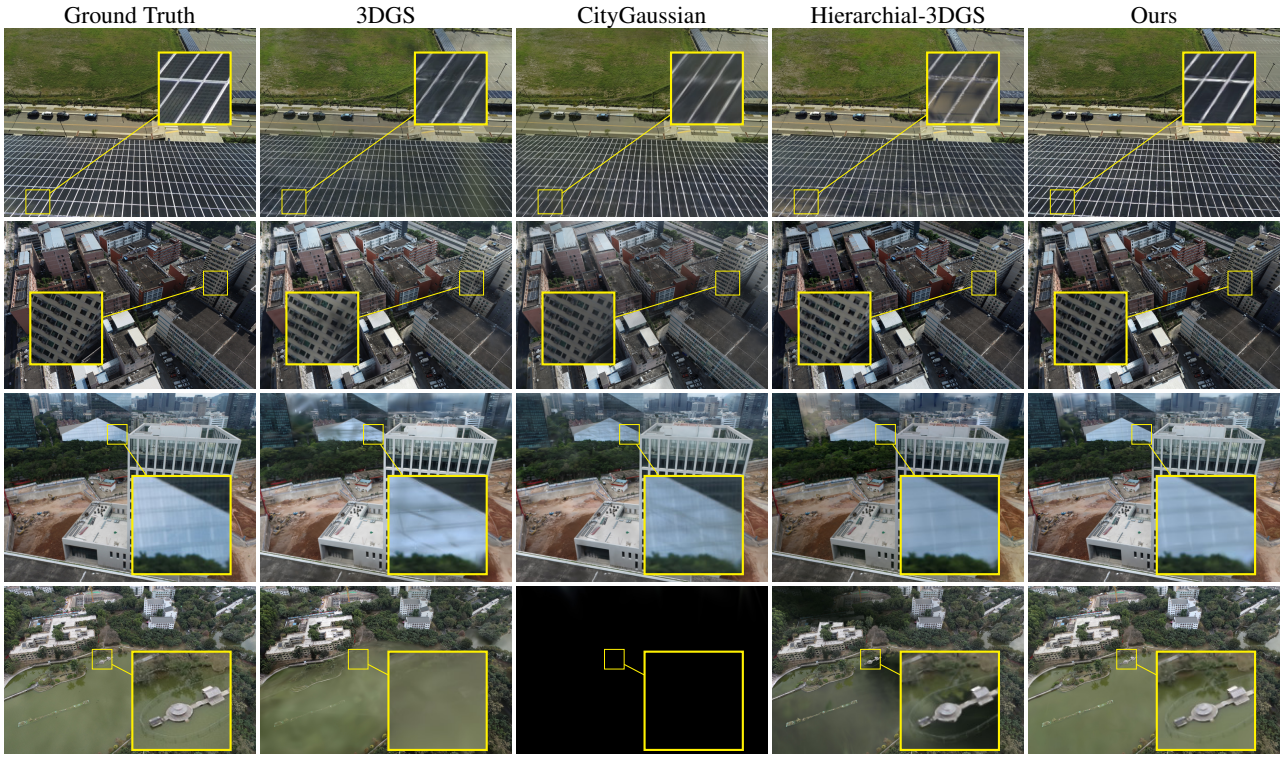



Figure 9. **Visualization results on *Building*, *Residence*, *Sci-Art* and *Campus* of ours and previous work.** All methods, except for 3DGS, render in LOD mode. The LOD mode of CityGaussian encountered a bug in the *Campus*, resulting in a completely black rendered image.

The second row of Figure 11 presents a statistical analysis of the similarity among Gaussians within a small local region, where 513 Gaussians are selected, and the similarities between 512 of them and a central reference Gaussian are computed to generate a histogram. In the absence of similarity regularization, most Gaussians exhibit low similarity, clustering around 0.1. Such low similarity results in significant differences in appearance transformations among Gaussians, leading to visible artifacts. In contrast, with similarity regularization applied, the similarity values among Gaussians predominantly exceed 0.8. This high degree of similarity ensures more consistent appearance adjustments across Gaussians, effectively preventing the emergence of artifacts.

| Scene | *Rubble* | | | | *JNU-ZH* | | | | *BigCity* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | SSIM | PSNR | LPIPS | #G | SSIM | PSNR | LPIPS | #G | SSIM | PSNR | LPIPS | #G |
| 1st level | 0.870 | 28.34 | 0.153 | 3.71 | 0.889 | 26.57 | 0.114 | 5.41 | 0.925 | 27.48 | 0.098 | 10.38 |
| 2nd level | 0.825 | 27.16 | 0.224 | 7.13 | 0.835 | 25.74 | 0.197 | 13.99 | 0.855 | 26.10 | 0.198 | 30.47 |
| 3rd level | 0.826 | 27.29 | 0.228 | 13.52 | 0.822 | 25.85 | 0.232 | 25.58 | 0.847 | 26.62 | 0.219 | 75.15 |

Table 9. **Quantitative evaluation of all the levels of our method, evaluated using the same downsampling factor as during training.**

| Scene | *Rubble* | | | | | *JNU-ZH* | | | | | *BigCity* | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | SSIM | PSNR | LPIPS | #G | FPS | SSIM | PSNR | LPIPS | #G | FPS | SSIM | PSNR | LPIPS | #G | FPS |
| w/o anti-aliasing | **0.817** | 26.85 | **0.237** | **3.02** | 103.5 | **0.817** | 25.69 | **0.233** | **4.92** | 67.8 | **0.847** | **26.52** | **0.213** | **6.68** | 73.6 |
| w/o absgrad | 0.795 | 26.67 | 0.275 | 3.15 | **109.8** | 0.808 | 25.53 | 0.251 | 6.78 | **68.3** | 0.831 | 26.28 | 0.244 | 6.74 | **78.1** |
| w/o tile-based cull. | 0.814 | **27.03** | 0.245 | 3.60 | 83.0 | 0.816 | **25.71** | 0.240 | 6.65 | 54.6 | 0.838 | 26.41 | 0.231 | 6.84 | 65.8 |
| full | 0.814 | **27.03** | 0.245 | 3.60 | 99.7 | 0.816 | **25.71** | 0.240 | 6.65 | 63.9 | 0.838 | 26.41 | 0.231 | 6.84 | 73.0 |

Table 10. **Additional qualitative ablations.**

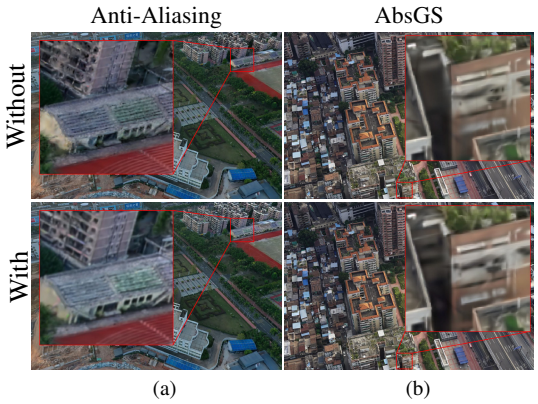



Figure 10. **Visualization results of ablation on Anti-Aliasing and AbsGS.**

| Length | SSIM | PSNR | LPIPS |
|---|---|---|---|
| 4 | 0.816 | 25.58 | 0.237 |
| 8 | 0.815 | 25.62 | 0.240 |
| 16 | **0.822** | **25.85** | **0.232** |

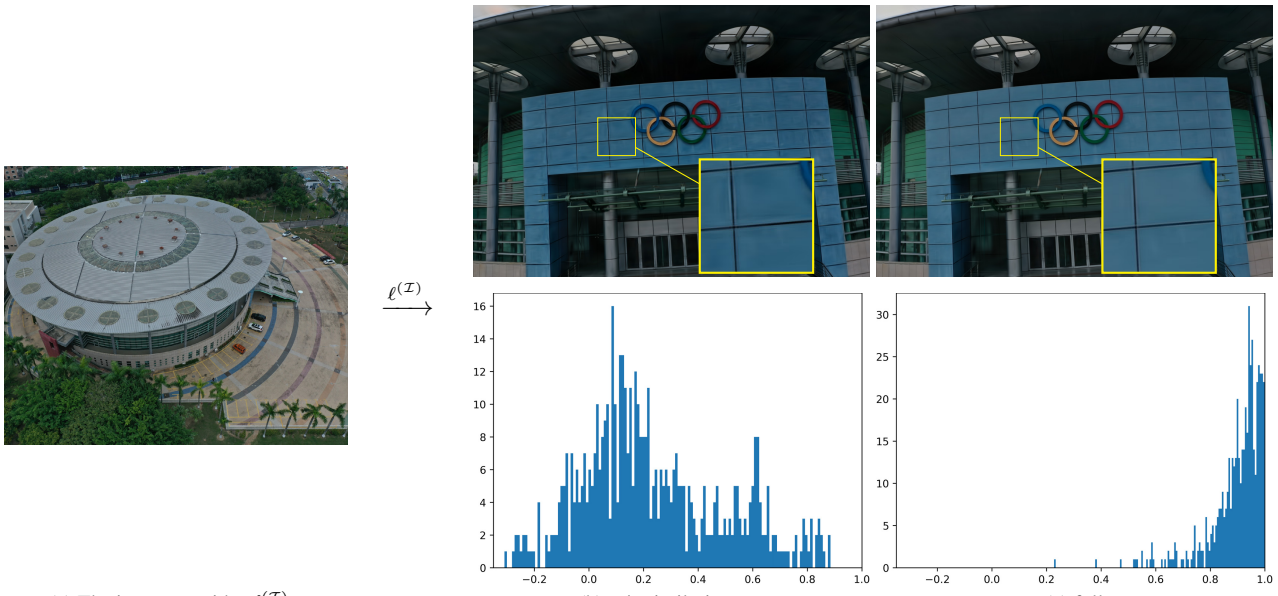
Table 11. **The impact of the length of $\ell^{(\mathcal{G})}$.**

| Part. Size | SSIM | PSNR | LPIPS | FPS | #G ($10^6$) | #P |
|---|---|---|---|---|---|---|
| 45m | 0.803 | 25.27 | 0.256 | 35.9 | **2.11** | 860 |
| 90m | 0.808 | 25.46 | 0.250 | 61.7 | 3.06 | 228 |
| 135m | 0.810 | 25.50 | 0.247 | **64.4** | 3.68 | 140 |
| 180m | **0.813** | **25.62** | **0.243** | 63.1 | 5.19 | 64 |

Table 12. **Qualitative ablations of different rendering-time partition size on the *JNU-ZH* scene.** #P represents the number of partitions.

| Distances | SSIM | PSNR | LPIPS | FPS | #G ($10^6$) |
|---|---|---|---|---|---|
| (45m, 90m, $\infty$) | 0.789 | 25.00 | 0.271 | **66.5** | **2.40** |
| (90m, 180m, $\infty$) | 0.808 | 25.46 | 0.250 | <u>61.7</u> | <u>3.06</u> |
| (135m, 270m, $\infty$) | <u>0.815</u> | <u>25.65</u> | <u>0.242</u> | 58.8 | 3.62 |
| (180m, 360m, $\infty$) | **0.818** | **25.73** | **0.238** | 56.2 | 4.13 |

Table 13. **Qualitative ablations of distance thresholds on the *JNU-ZH* scene with a partition size of 90m.** The distance values represent the maximum distances at which the 3rd, 2nd, and 1st LOD levels are used.




(a) The image provides $\ell^{(\mathcal{I})}$ (b) w/o similarity reg. (c) full

Figure 11. **A visual comparison of results with and without similarity regularization.**