HUG: Hierarchical Urban Gaussian Splatting with Block-Based Reconstruction for Large-Scale Aerial Scenes

Supplementary Material

We provide a more detailed discussion on the settings of the compared methods, view partition results, and additional qualitative comparisons. A rendered video is also included for a more comprehensive comparison.

6. Settings of Compared Methods

We compared our method against several state-of-the-art baselines, including Mega-NeRF [33], 3DGS [11], CityGS [17], OctreeGS [26], VastGS [15], and Hier-GS [12]. The quantitative results for Mega-NeRF [33], 3DGS [11], and CityGS [17] are taken from the CityGS [17] paper. The quantitative results for OctreeGS [26] are sourced from the OctreeGS [26] paper. The results for Hier-GS [12] were obtained using the released source code with all default settings. The results for VastGS [15] are derived from the unofficial code available at https://github.com/kangpeilun/VastGaussian, with the same block number as CityGS [17] and ours. The qualitative results for CityGS [17] were obtained from the checkpoint provided by the authors.

7. Data Partitioning Results

To associate all training images with their respective blocks, we compute the total number of visible sparse points in each view. A threshold of $\tau_p = 800$ is then applied to determine the assignment of each training image to specific blocks. In CityGS [17], a coarse global 3DGS [11] model must first be constructed, followed by multiple renderings of all images corresponding to the number of blocks. VastGS [15] introduces a more complex camera selection strategy, incorporating position-based data selection, visibility-based camera selection, and coverage-based point selection, resulting in a significantly higher number of assigned cameras per block. VastGS[15] has an average of over 1,000 views per block on the MatrixCity dataset, which exceeds GPU memory limits for a single block. This is why we do not report its numerical results on this scene. For visual comparisons, we avoid that specific block whenever possible. In contrast, our data partitioning strategy assigns fewer images to each block, reducing memory consumption during training and allowing more computational resources to be dedicated to refining the in-block reconstruction. Figure 6 visualizes the number of training images per block, illustrating that our method allocates fewer images on average while still achieving superior rendering performance, as demonstrated in Table 1.

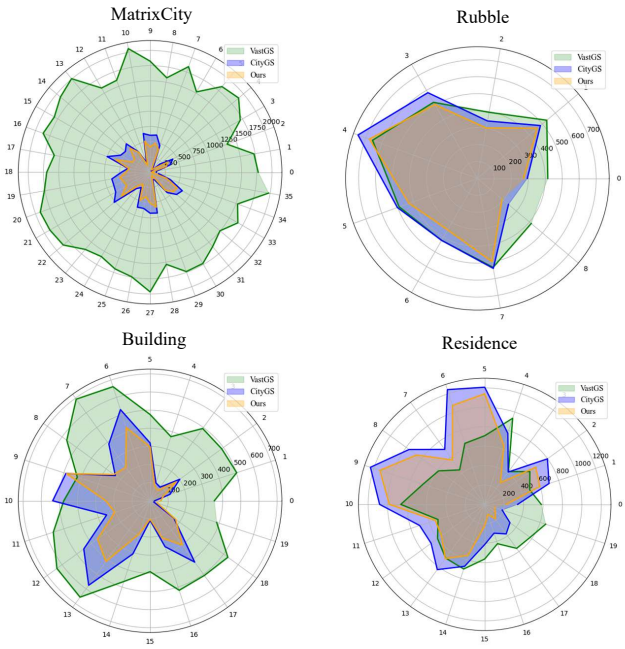

Figure 6. Comparison of view partition results among our method, VastGS [15], and CityGS [17] across four datasets. The radial coordinate indicates the number of training images assigned to each block, with each vertex representing a specific block. As shown, our method consistently assigns fewer images on average, demonstrating a more efficient partition strategy.

8. More Qualitative Comparisons

We provide additional qualitative comparisons in Figure 7 and Figure 8.

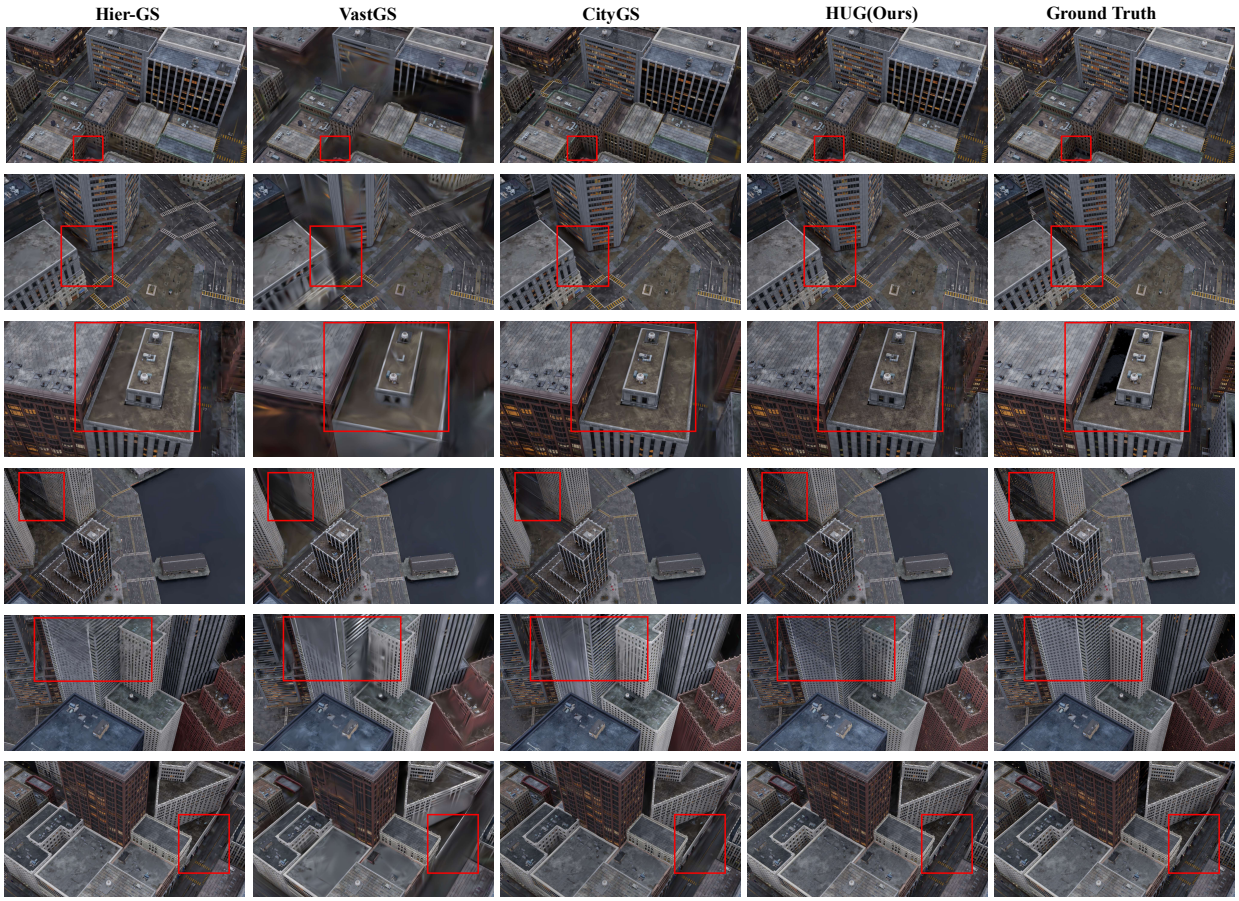

Figure 7. More qualitative comparisons on *MatrixCity* dataset.

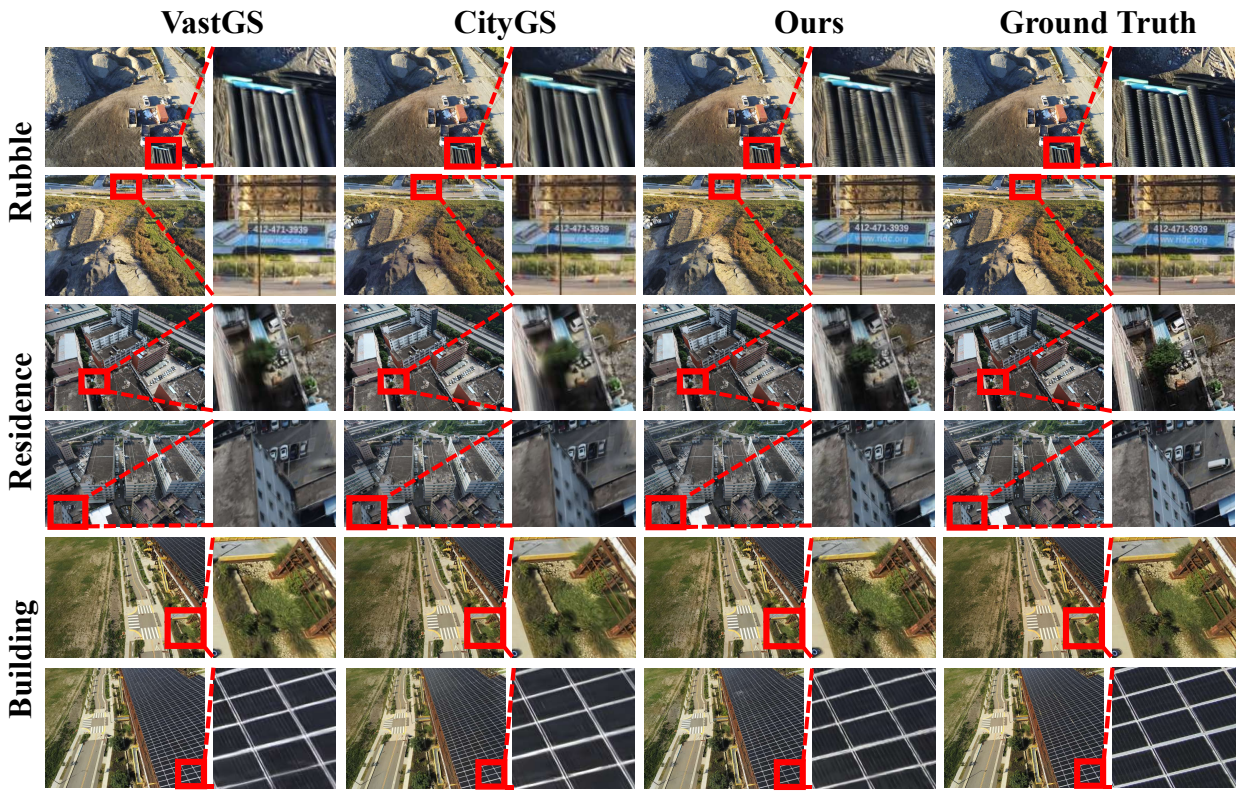

Figure 8. More qualitative comparisons on *Rubble*, *Residence* and *Building* datasets.