From Orbit to Ground: Generative City Photogrammetry from Extreme Off-Nadir Satellite Images

Supplementary Material

A. Appendix Overview

In this appendix, we provide additional details and results to complement the main paper. Specifically, Sec. B presents extended implementation details of our geometry and appearance pipelines, including the acquisition of satellite-based point clouds, the optimization of our Z-Monotonic SDF, and details for appearance modeling. Sec. C further describes the experimental settings, including dataset configurations, baseline setups, and evaluation metrics. Finally, Sec. D reports more extensive quantitative and qualitative results, ablation studies, and an additional application example that validate the scalability and robustness of our method. We additionally provide an extra comparison with mesh-recovery baselines under sparse satellite observations, a real-scene geometric evaluation using partial LiDAR, a visualization of the ground-view locations used in the main-paper qualitative results, and further analyses on UV atlas parameterization and input sparsity.

B. Implementation Details

In this section, we introduce additional implementation details in our framework.

B.1. Point Clouds from Multi-View Stereo

In our framework, we first extract rooftop point clouds from MVS. In implementation, we utilize MVSFormer++ and Colmap to extract point clouds. We now describe the details.

To adapt MVSFormer++ [5] for satellite-based MVS, we constructed a new large-scale dataset comprising 2,600 unique satellite scenes with corresponding ground truth depth maps. This dataset was used to train the network specifically for satellite imagery. To enhance point cloud quality, we upgraded the feature extraction backbone in MVSFormer++ by replacing the pre-trained DINOv2 [43] with the more powerful DINOv3 [56]. During inference on real-world multi-view satellite images with their corresponding RPC camera models, we first employ SatelliteSfM [69] to obtain approximate pinhole camera projections, which then serve as the input for our trained MVSFormer++.

B.2. Z-Monotonic SDF Optimization

Our geometry stage reconstructs a high-fidelity, watertight mesh from an MVS point cloud P. We detail the optimization process below, which is applied to 4 spatial divisions of the full scene. The final city model is produced by merging the resulting meshes from each part.

Z-Monotonic SDF Representation For each of the 4 scene divisions, we implement the Z-Monotonic SDF by parameterizing a field of monotonic curves on a 2D (x, y) grid. This is implemented as a learnable 2D parameter grid of resolution 256×256. When querying the SDF value $s(\mathbf{x})$ at a 3D point $\mathbf{x} = (x, y, z)$, we compute the function from Eq. 5. This is achieved by:

- Defining a 2D neighborhood of size 3×3 centered at the query's (x, y) location.
- Retrieving the height offset parameters $\{h_j\}$ for this neighborhood by bilinearly sampling from the learnable parameter grid.
- Calculating the weights $\{w_j\}$ using a softmax of the inverse xy-plane distance to these neighbors.
- Computing the final weighted sum $f(z; x, y) = \sum_{j=1}^{n} w_j \cdot \tanh(k \cdot (z - h_j))$, where k is a fixed hyperparameter controlling the curve sharpness, and we set it to 80.

This formulation ensures the SDF is continuous, differentiable, and inherently Z-monotonic.

Differentiable Optimization We optimize the learnable parameters of the 2D grid for each scene part. We use the Adam optimizer with a learning rate of 0.01 and optimize the grid for 2000 steps. At each optimization step, we extract a mesh M from the current SDF using FlexiCubes [55]. The FlexiCubes module operates on a grid of resolution 128^3, enabling gradients to flow from the mesh-based loss back to the 2D grid parameters. The optimization is supervised by the $\mathcal{L}_{\text{geo}}$ loss

function, as defined in Eq. 2:

$$\mathcal{L}_{\text{geo}} = \sum_{\mathbf{p}\in P} \min_{\mathbf{m}\in M} \|\mathbf{p}_z - \mathbf{m}^*(\mathbf{p})_z\|_1 + \lambda_{\text{Lap}}\mathcal{L}_{\text{Lap}} + \lambda_{\text{Nrm}}\mathcal{L}_{\text{Nrm}}.$$

We detail the definition for each loss term below.

Height Supervision ($\sum_{\mathbf{p}\in P} \min_{\mathbf{m}\in M} \|\mathbf{p}_z - \mathbf{m}^*(\mathbf{p})_z\|_1$) We supervise the Z-Monotonic SDF optimization using height-based supervision. To ensure both efficiency and numerical stability, we avoid computing per-point nearest neighbors on the mesh at each iteration. Instead, we project the MVS point cloud onto a fixed 2D grid and compare it with a rasterized height map rendered from the current mesh. Given the normalized MVS point cloud $P = \{\mathbf{p}_i\}_{i=1}^{N_P}$ in $[-1, 1]^3$, we first convert it into a 2D height map on a regular grid of resolution $R \times R$. For each point $\mathbf{p}_i = (x_i, y_i, z_i)$, we compute its grid indices

$$u_i = \left\lfloor \frac{x_i + 1}{2} R \right\rfloor, \quad v_i = \left\lfloor \frac{y_i + 1}{2} R \right\rfloor, \tag{A1}$$

and accumulate the maximum height per grid cell:

$$H_{\text{target}}(u, v) = \max\{\, z_i \mid (u_i, v_i) = (u, v) \,\}, \tag{A2}$$

for all (u, v) that receive at least one point. Cells that do not receive any point are marked as invalid and excluded from the loss via a binary mask.

In parallel, at each optimization step we render a height map H_{pred} from the current mesh M by rasterizing it onto the same $R \times R$ grid. We first transform the mesh vertices from normalized coordinates $[-1, 1]^3$ to image coordinates:

$$\tilde{\mathbf{v}} = \left(\tfrac{x+1}{2}R,\ \tfrac{y+1}{2}R,\ z\right), \tag{A3}$$

and then use a standard differentiable rasterizer to project the mesh triangles onto the (u, v) plane. For each pixel (u, v), we compute the interpolated height z via barycentric interpolation of the triangle vertices that cover that pixel. This yields a dense height map $H_{\text{pred}}(u, v)$ that is fully differentiable with respect to the mesh vertices, and thus with respect to the underlying SDF parameters.

We then define the height-map loss as an L_1 difference between the two height maps, computed only on valid pixels where MVS observations exist:

$$\mathcal{L}_{\text{H}} = \frac{1}{|\Omega|} \sum_{(u,v)\in\Omega} \left| H_{\text{pred}}(u, v) - H_{\text{target}}(u, v) \right|, \tag{A4}$$

where Ω denotes the set of pixels with valid MVS-derived heights. In implementation, we set $R = 1024$.

Laplacian Regularization ($\mathcal{L}_{\text{Lap}}$) We employ the standard laplacian loss. This loss encourages a smooth mesh surface by penalizing the L_2 distance between each vertex $\mathbf{v}_i$ and the uniform average of its 1-ring neighboring vertices $\mathcal{N}(i)$. The loss is defined as:

$$\mathcal{L}_{\text{Lap}} = \frac{1}{|V|} \sum_{\mathbf{v}_i\in V} \left\| \mathbf{v}_i - \frac{1}{|\mathcal{N}(i)|} \sum_{j\in\mathcal{N}(i)} \mathbf{v}_j \right\|_2^2, \tag{A5}$$

where V is the set of all vertices in the extracted mesh M. This helps to reduce high-frequency "bumpy" artifacts from the MVS point cloud.

In implementation, we set $\lambda_{\text{Lap}} = 0.5$.

Normal Consistency ($\mathcal{L}_{\text{Nrm}}$) The normal consistency term $\mathcal{L}_{\text{Nrm}}$ is implemented as a Total Variation (TV) loss on the *rendered normal map* $\mathbf{N}$. This encourages smooth changes in surface orientation, which is particularly important for flat facades and ground planes.

At each optimization step, we render a normal map $\mathbf{N}$ from the current mesh M. The loss is defined as the sum of the mean L_2 norms of the finite differences of the normalized normal vectors $\hat{\mathbf{n}}$ between adjacent pixels, computed in the horizontal (x) and vertical (y) directions:

$$\mathcal{L}_{\text{Nrm}} = \mathbb{E}_{u,v}\left[\|\hat{\mathbf{n}}_{u,v} - \hat{\mathbf{n}}_{u+1,v}\|_2\right] + \mathbb{E}_{u,v}\left[\|\hat{\mathbf{n}}_{u,v} - \hat{\mathbf{n}}_{u,v+1}\|_2\right], \tag{A6}$$

where $\hat{\mathbf{n}}_{u,v}$ is the 3D unit normal vector at pixel (u, v) in the rendered map $\mathbf{N}$. This loss effectively penalizes sharp, local changes in surface normals, promoting piecewise-planar structures.

In implementation, we set $\lambda_{\text{Nrm}} = 0.01$.

FlexiCubes [55] Regularization Following [55], we also add a regularization term $\lambda_{\text{reg}}\mathcal{L}_{\text{reg}}$ in the geometric optimization, where $\lambda_{\text{reg}} = 0.1$, and $\mathcal{L}_{\text{reg}}$ follows the same definition in [55].

Final Mesh Generation After the optimization converges for all 4 divisions, we extract the final mesh for each part. The vertices are un-normalized to their original world coordinates. These 4 meshes are then merged to produce the single city model. Finally, to prepare the mesh for the appearance stage, we compute a single UV atlas for the entire geometry with a resolution of 8192×8192.

B.3. Appearance Modeling

The core of our appearance enhancement is the image restoration network D, a generative model fine-tuned to deterministically map degraded, low-quality renders to sharp, photorealistic targets.

Network Architecture We construct our restorer D by adapting the pre-trained FLUX-Schnell [28], a state-of-the-art diffusion transformer based on Rectified Flow (RF). While standard RF formulations model image synthesis as an Ordinary Differential Equation (ODE) initiated from Gaussian noise, this stochastic initialization introduces variance that is detrimental to our texture optimization. Our iterative framework strictly requires a stable, one-to-one mapping between the degraded render and the enhanced output to ensure convergence.

To enforce determinism, we reformulate the generation task as a direct, single-step restoration. Instead of sampling a random noise vector, we encode the degraded render I_{low} into a latent code $z_{\text{low}} = E(I_{\text{low}})$ using the pre-trained VAE encoder. This latent code serves as the deterministic boundary condition for the ODE trajectory. Through fine-tuning, the network learns to map the corrupted latent z_{low} directly to its clean counterpart z_{high} in a single function evaluation. This establishes a deterministic mapping $D : z_{\text{low}} \rightarrow z_{\text{high}}$, providing the consistent supervisory signals essential for the stability of our optimization.

Fine-tuning Dataset Construction To train D effectively, we constructed a specialized dataset of 100,000 paired images. This dataset was generated from our extensive internal collection of high-quality 3D urban assets, which is completely disjoint from any of our test scenes to prevent data leakage. From each 3D asset, we rendered multiple sets of paired images $(I_{\text{low}}, I_{\text{high}})$ from low-altitude (e.g., 400–600 m height) and oblique-angle (e.g., $45°$–$60°$ pitch). Each pair consists of a high-resolution, photorealistic rendering I_{high} serving as the ground-truth target, and a corresponding degraded version I_{low} rendered from the identical viewpoint but using a reduced Level-of-Detail (LoD). This degradation strategy naturally suppresses fine-grained textures and high-frequency details, thereby accurately simulating the coarse inputs encountered during the iterative refinement stage.

C. Experimental Settings

C.1. Datasets

In the main paper, we evaluate our method on both synthetic and real-world datasets. We now provide the detailed configurations for each benchmark.

MatrixCity-Satellite We build a synthetic satellite benchmark on top of the MatrixCity-Satellite [30] dataset using the UE5 engine, following the overall protocol described in the main paper. We now detail our capture settings.

For training views, we adopt a clean and stable environment configuration with: no fog, no dynamic weather, and no traffic or moving objects. We place virtual cameras at a fixed altitude of 2000 m above the ground (the maximal value, to the best of knowledge, that yields stable rendering in UE5), with a near-nadir viewing direction. Concretely, we set the yaw angle to $89°$ relative to the ground plane, a resolution of 2560×1440, and a field-of-view (FOV) of $22.42°$ such that the resulting ground sampling distance (GSD) is 0.31 m/px, matching WorldView-3 satellite imagery.

We derive the field-of-view based on the desired ground sampling distance and image resolution. Let H denote the camera altitude above the ground, W the image width in pixels, and θ the horizontal FOV. The ground footprint width L covered by the image is related to the FOV by

$$L \;=\; 2H \tan\!\left(\frac{\theta}{2}\right). \tag{A7}$$

For a target ground sampling distance GSD, the footprint width is also given by

$$L \;=\; W \cdot \mathrm{GSD}. \tag{A8}$$

Equating the two expressions for L yields

$$2H \tan\!\left(\frac{\theta}{2}\right) \;=\; W \cdot \mathrm{GSD}, \tag{A9}$$

from which the FOV can be solved as

$$\theta \;=\; 2\arctan\!\left(\frac{W \cdot \mathrm{GSD}}{2H}\right). \tag{A10}$$

Substituting our settings $H = 2000\,\mathrm{m}$, $W = 2560$, and $\mathrm{GSD} = 0.31\,\mathrm{m/px}$, we obtain

$$\theta \;=\; 2\arctan\!\left(\frac{2560 \times 0.31}{2 \times 2000}\right) \;\approx\; 22.42^\circ, \tag{A11}$$

which is the FOV used in all our synthetic satellite captures.

To ensure sufficient overlap between neighboring satellite views, following practical experience, we enforce approximately 60% horizontal overlap between adjacent captures. Given the horizontal footprint width $L \approx 792.7\,\mathrm{m}$ (derived from the FOV and altitude described above), this corresponds to a target stride of

$$s \;=\; (1 - 0.6)\, L \;\approx\; 0.4 \times 792.7 \;\approx\; 317.1\,\mathrm{m}. \tag{A12}$$

In practice, we set the sampling distance to $s = 317.44\,\mathrm{m}$, which yields a horizontal overlap very close to 60% between neighboring views.

We then translate the camera centers on a regular grid with this fixed stride along the x- and y-axes, covering the central $[-500, 500]\,\mathrm{m} \times [-500, 500]\,\mathrm{m}$ area of the scene. At each grid location, we place two virtual cameras with different off-nadir viewing directions to simulate multi-directional satellite passes. Concretely, we use two sets of rotations:

$$(\mathrm{pitch}, \mathrm{yaw}, \mathrm{roll}) \;=\; (0^\circ, 89^\circ, 0^\circ) \quad \text{and} \quad (0^\circ, 89^\circ, 90^\circ), \tag{A13}$$

corresponding to south-looking and west-looking near-nadir views, respectively. This configuration results in a sparse multi-view satellite observation pattern that closely approximates real-world acquisition geometries.

For ground-truth point clouds, we use the point clouds provided by MatrixCity-Satellite and select the central $[-400, 400]\,\mathrm{m}^2$ region for evaluation. For test views, we follow the default *MatrixCity-Aerial* evaluation protocol and extend it to two altitudes to assess multi-scale performance. We place cameras at heights of $200\,\mathrm{m}$ and $500\,\mathrm{m}$, with a resolution of 1920×1080, FOV of 45°, and a pitch angle of 45° toward the ground. Camera centers are sampled on a uniform grid over the central $[-200, 200]\,\mathrm{m}^2$ region with a fixed sampling interval ($45.01\,\mathrm{m}$) along both axes. This yields 72 test views that cover the evaluation area with sufficient angular diversity while avoiding boundary regions where the reconstruction quality naturally degrades. These views are used for novel view synthesis evaluation in the main paper.

DFC 2019 For the DFC 2019 benchmark, we follow the overall evaluation protocol of Skyfall-GS [29] in terms of AOI selection and input modality. We obtain approximate pinhole camera intrinsics and extrinsics from the provided RPC parameters using SatelliteSfM [69], and use these cameras both for all competing methods and for rendering our results.

To obtain ground-level reference views, we follow Skyfall-GS [29] and use video sequences from Google Earth Studio (GES) along a low-altitude orbital trajectory around each AOI as test views. In practice, directly using the camera trajectories of [29] may lead to invalid pixels near the image boundaries (*e.g.*, black borders). To ensure a fair and robust evaluation, we additionally apply a simple binary mask that removes boundary pixels before computing the image-based metrics.

GoogleEarth For the GoogleEarth benchmark, we also follow the evaluation protocol of Skyfall-GS [29] in terms of AOI selection, input data, and camera trajectories. For each AOI, we use the training and test views from Google Earth Studio, matching the satellite-like input configuration of [29]. Similar to the DFC 2019 case, these renders may contain invalid pixels near the image borders. We therefore apply the same binary boundary mask as in the DFC 2019 evaluation.

In addition, we empirically observe a systematic misalignment in the NYC_004 scene. The camera parameters provided in [29] do not accurately reproduce the released GES images, and this discrepancy cannot be resolved by simple refinement. To avoid introducing bias in the quantitative comparison, we exclude NYC_004 from our metric reporting and only include NYC_010, NYC_219, and NYC_336 in the main paper.

C.2. Baselines

We compare our method against four representative baselines, as described in the main paper: Mip-Splatting [67], 2DGS [21], CityGS-X [19], and Skyfall-GS [29].

For [29], we use the official implementation and default hyperparameters recommended by the authors for satellite-based reconstruction. Specifically, we evaluate the MatrixCity-Satellite dataset with the recommended DFC 2019 hyperparameters, and to match the scale of DFC 2019 scenes, we uniformly scale up the MatrixCity-Satellite scene by a factor of $50\times$ before running Skyfall-GS.

For the other baselines, directly adopting their default hyperparameters (which are typically designed for dense aerial or street-view imagery) leads to numerical instability or divergence when applied to our extreme off-nadir satellite setting. To ensure a fair and stable comparison, we carefully re-tune a small subset of hyperparameters while keeping the overall model architectures unchanged. In particular, for [19, 21, 67], we reduce the initial position learning rate from 1.6×10^{-4} to 1.6×10^{-5}, decrease the densification ratio from 0.01 to 0.001, slightly increase the densification gradient threshold from 2.0×10^{-4} to 4.0×10^{-4}, and disable frequent opacity resets. In addition, we extend the far plane from 100.0 to 1.0×10^{8} and forward the principal point (c_x, c_y) estimated by COLMAP into the projection matrix, so that these baselines can handle our kilometer-scale, satellite-like scenes without modifying their core models.

C.3. Metrics

For PSNR and SSIM, we follow standard image quality evaluation protocols. For LPIPS, we adopt the official implementation with the AlexNet backbone, as commonly done in prior work.

For geometric evaluation, we report the Chamfer distance (CD) and the Precision/Recall/F1 metrics between the reconstructed and ground-truth point clouds, following [19, 21, 36]. The Precision/Recall/F1 metrics are adopted in [19, 36]. For the distance threshold d_τ for aerial reconstructions, it is typically set to 0.006 in normalized scene units, which corresponds to an effective ground sampling distance (GSD) of about 5 cm/px in aerial settings. In our satellite setting, however, the GSD is approximately 31 cm/px, *i.e.*, about six times coarser than in the aerial case. To maintain a comparable geometric tolerance in physical units, we therefore scale the threshold proportionally and set $d_\tau = 0.036$ which yields a consistent relative matching criterion under our satellite imaging configuration.

For point cloud extraction, we follow the native pipelines of each method whenever possible: for [19, 21] we directly use their official mesh export routines, while for [29, 67] we convert the optimized 3D Gaussian representations into meshes using the SOTA method [1]. We then uniformly sample points on all resulting meshes following the protocol of [19] to obtain the predicted point clouds used in our geometric evaluation.

D. More Results

D.1. Quantitative Comparison

Tabs. A1 and A2 present per-scene quantitative comparisons on the DFC 2019 and GoogleEarth datasets, respectively. Across all AOIs, our method consistently matches or surpasses state-of-the-art baselines in PSNR and SSIM, while achieving competitive or better LPIPS. Notably, the "Ours w/o Image Restoration Network" variant already improves over most baselines, and further gains are obtained by our full appearance modeling pipeline, highlighting the contribution of our deterministic restoration network.

D.2. Qualitative Comparison with Ground-Truth Views

Figs. A1 to A8 provide per-scene qualitative comparisons between our method, competing baselines, and the ground-truth test views. Our reconstructions preserve fine-scale structural and texture details, such as building facades and roof patterns, which are often blurred, distorted, or missing in alternative methods.

D.3. Qualitative Analysis at Varying Camera Altitudes

To further investigate how our reconstructed city models behave under varying observation altitudes, we conduct a controlled rendering study with fixed camera intrinsics and identical view geometry. Specifically, we fix the camera's horizontal position, look-at point, and intrinsics ($f_x, f_y = 1500$), while systematically varying the camera height at $[50, 200, 400, 600, 800]$ meters.

Fig. A9 presents a qualitative comparison between our method and Skyfall-GS [29] in real-world scenes. Our model exhibits stronger robustness, consistently delivering superior results across different observation altitudes. As the camera height decreases, the performance gap becomes increasingly evident: our method preserves markedly sharper facade textures and more refined geometric details. While our reconstructions may exhibit minor seams due to the joint optimization over multiple satellite views, Skyfall-GS suffers from pronounced quality degradation at lower altitudes, including widespread blurring and noticeable "floating object" artifacts.

D.4. Close-Up Views

The extreme viewpoint gap between orbital inputs and low-altitude target views poses a particular challenge for satellite-based reconstruction. To further evaluate our method under such challenging conditions, we provide additional close-up comparisons that focus on fine-scale geometric and texture details.

Fig. A10 shows close-up novel views on the MatrixCity-Satellite benchmark, comparing our results against Skyfall-GS [29] and the ground truth. Our method produces sharper facades and more detailed roof textures, with significantly fewer ghosting and aliasing artifacts.

Similarly, Figs. A11 and A12 present close-up comparisons on DFC 2019 and GoogleEarth, respectively. Across all real-world scenes, our reconstructions retain fine-grained details such as window patterns and vertical edges better than Skyfall-GS, validating the effectiveness of our two-stage geometry and appearance modeling for close-up observations.

D.5. Additional Real Urban Scenes

Beyond the benchmark datasets used in the main paper, we further validate the generalization ability of our method on two additional real-world urban scenes. These scenes are of the same type and scale as the *Urban Scene* in the main paper, but are completely disjoint in terms of geographic location and appearance.

Figs. A13 and A14 show bird's-eye overviews of two scenes reconstructed by our method. The results demonstrate that, without dataset-specific tuning, our approach can robustly handle diverse urban layouts, ranging from dense high-rise clusters to wide road networks and large open areas, while maintaining globally coherent geometry and visually plausible textures across each entire scene.

D.6. Ablation Study

We provide additional ablation results to complement the analysis in the main paper.

Fig. A15 compares different geometric representations and meshing strategies on the MatrixCity-Satellite scene. From left to right, we visualize the ground truth, a naive 2.5D mesh obtained via Marching Cubes at low and high resolutions (MC 128/256), a full 3D SDF optimized with FlexiCubes, a variant trained with a Chamfer distance-based loss only, and our full Z-Monotonic SDF model. Higher-resolution Marching Cubes still suffer from stair-step artifacts, and full 3D SDFs completely fail due to topological inconsistencies. Our Z-Monotonic SDF yields cleaner roofs and vertically extruded facades, leading to the most faithful reconstruction.

Fig. A16 examines the impact of different appearance modeling choices. We show the ground truth, a variant excluding our image restoration network ("w/o Image Restoration"), a variant that directly uses the off-the-shelf FLUX-Kontext model for enhancement, and our full appearance pipeline. Without explicit restoration, textures remain blurry and exhibit baked-in projection artifacts. Using FLUX-Kontext improves sharpness but introduces view-inconsistent hallucinations. In contrast, our fine-tuned, deterministic restorer produces sharp and globally consistent textures across views, which is critical for stable texture optimization.

D.7. Applications

We illustrate an additional application of our framework beyond satellite-to-ground reconstruction. Fig. A17 shows an example where our method is applied to reconstruct an aerial-view scene from a set of oblique aerial images [46]. The top row visualizes a subset of the input views, while the bottom row shows renderings from novel viewpoints using our reconstructed mesh and textures. The results demonstrate that our 2.5D geometry prior and generative appearance refinement are also applicable to aerial photogrammetry, yielding high-quality assets.

D.8. Comparison with Remote Sensing Methods

In the remote sensing community, several methods have also investigated 3D reconstruction from satellite imagery [2, 3, 12, 39, 40, 70, 71]. These methods are primarily designed for accurate digital surface model (DSM) estimation and elevation recovery over large areas, and thus mainly focus on metric height accuracy. In contrast, our work targets high-fidelity reconstruction of the *full 3D scene appearance*, including building facades and complex urban details, for photorealistic novel view synthesis from ground and near-ground viewpoints. This difference in objective naturally leads to complementary strengths: DSM-oriented methods excel at producing geographically accurate surface models, whereas our method emphasizes visually coherent, view-consistent 3D assets suitable for immersive rendering and city-scale visual applications.

Following Skyfall-GS [29], we compare our method against two representative methods Sat-NeRF [71] and EOGS [2] on the DFC 2019 dataset in terms of novel view synthesis quality. As shown in Tab. A3, our method achieves superior performance across all metrics, indicating that our reconstruction better preserves the visual fidelity of the 3D scene from challenging novel viewpoints.

We further provide qualitative comparisons in Fig. A18. As illustrated, Sat-NeRF and EoGS are able to recover plausible elevation and coarse structure, but their rendered views tend to exhibit oversmoothed textures and limited facade details when viewed from oblique or near-ground viewpoints. In contrast, our method produces sharper, more realistic facades and richer high-frequency details. These results underscore the complementary nature of our approach with respect to DSM-oriented remote sensing methods.

D.9. Failure Cases

As discussed in the Limitations section, our reliance on a 2.5D Z-monotonic geometric prior prevents our method from faithfully modeling non-monotonic structures such as bridges, overpasses, and other multi-layer configurations. In these cases, the Z-monotonic assumption forces the geometry to collapse along the vertical axis, leading to missing underpasses. Fig. A19 shows two representative examples from the *MatrixCity-Satellite* dataset, where the ground-truth geometry exhibits multiple vertical layers, while our reconstruction either merges them into a single surface or produces incomplete structures. We believe that future work could further alleviate these issues, for example by augmenting the 2.5D scaffold with local full-3D representations or topology-aware modules that handle multi-layer and non-monotonic structures, thereby making the framework more flexible in such challenging cases.

D.10. Additional Comparison with Mesh-Recovery Baselines

To better position our geometry module with respect to mesh-recovery approaches, we further compare against representative mesh-recovery baselines, including NKSR [22], LightweightMR [68], NDC [8], and ODC [24]. Fig. A20 shows that generic mesh-recovery methods remain challenged in this setting because the input MVS observations are dense on roofs and ground but sparse or empty on building facades under extreme off-nadir views. As a result, point-based and SDF-based approaches often produce incomplete wall geometry, while voxel-based methods face a resolution – sparsity trade-off and may introduce holes or discretization artifacts. In contrast, our Z-Monotonic SDF imposes the structural prior directly at the representation level, yielding cleaner watertight meshes with sharper vertical facades.

D.11. Ground-View Locations

To clarify the camera placement of the qualitative results in Fig. 6, we mark the corresponding ground-view locations in Fig. A21. The starred markers indicate the viewpoints used for rendering the close-range results.

D.12. Real-Scene Geometry Evaluation

For real-scene geometry evaluation, dense ground-truth geometry is unavailable for the full Urban Scene. We therefore further collect LiDAR data for a partially overlapping region and report a one-sided Chamfer distance from LiDAR to mesh after alignment on the overlap region. As shown in Tab. A4, our method achieves the lowest error among all baselines.

D.13. UV Atlas Visualization

We further provide a visualization of the UV atlas used in our appearance stage, as well as an additional ablation on the number of input satellite views. For UV parameterization, we use the UV atlasing tool in Open3D on the final merged mesh and keep the atlas fixed during texture optimization. Fig. A22 (a) visualizes a representative UV map on the Urban Scene.

D.14. Input Sparsity Ablation

To evaluate robustness to input sparsity, we reconstruct the same real Urban Scene using 3, 5, and 11 input satellite images sampled from the same capture set. As shown in Fig. A22 (b) – (d), our method maintains comparable geometry and visual quality even with fewer views, indicating that the geometric prior effectively stabilizes optimization in sparse-view regimes.

D.15. Additional Clarifications

Why the problem is ill-posed. Under extreme off-nadir satellite imaging, vertical facades exhibit very limited parallax and are heavily foreshortened. Consequently, different 3D wall geometries may induce nearly indistinguishable observations in the input views. The inverse mapping from sparse satellite images to full 3D city geometry is therefore non-unique, especially on facades where MVS points are sparse or absent. Our Z-Monotonic prior reduces this ambiguity by restricting the solution space to 2.5D urban surfaces.

Training strategy of the restoration network. The restoration network D is trained once on a large paired dataset of urban renderings and is then reused across all benchmarks without per-dataset retraining or test-scene finetuning.

Optimization details of iterative texture refinement. At each refinement iteration, we sample pseudo views from a regular grid of simulated UAV poses over the scene bounding box. These poses are synthetically defined rather than obtained from ground-truth UAV trajectories. The rendered views are passed through D to obtain restored targets, which are then used to supervise the texture optimization. We randomly shuffle the sampled views within each iteration and empirically observe negligible sensitivity to the optimization order.

Snow simulation. The snowy results in Fig. 8 are produced using Blender on top of our reconstructed mesh. This application demonstrates that the mesh output of our method can serve as an application-ready asset for downstream urban simulation.

Scene	Method	PSNR ↑	SSIM ↑	LPIPS ↓
JAX_004	Mip-Splatting	13.124	0.262	0.868
	2DGS	6.958	0.209	0.906
	CityGS-X	FAIL	FAIL	FAIL
	Skyfall-GS	12.955	0.248	0.790
	Ours w/o Image Restoration Network	13.524	0.267	0.664
	Ours	13.857	0.294	0.606
JAX_068	Mip-Splatting	7.950	0.314	0.834
	2DGS	8.079	0.300	0.839
	CityGS-X	FAIL	FAIL	FAIL
	Skyfall-GS	11.855	0.300	0.762
	Ours w/o Image Restoration Network	12.824	0.348	0.576
	Ours	12.935	0.341	0.554
JAX_214	Mip-Splatting	9.003	0.404	0.766
	2DGS	6.794	0.357	0.813
	CityGS-X	FAIL	FAIL	FAIL
	Skyfall-GS	12.318	0.398	0.696
	Ours w/o Image Restoration Network	12.566	0.386	0.582
	Ours	12.467	0.397	0.539
JAX_260	Mip-Splatting	11.078	0.402	0.795
	2DGS	7.633	0.349	0.832
	CityGS-X	FAIL	FAIL	FAIL
	Skyfall-GS	12.713	0.371	0.715
	Ours w/o Image Restoration Network	12.793	0.373	0.576
	Ours	12.977	0.400	0.526

Table A1. Quantitative evaluation of our method compared to prior works on every scene of **DFC 2019** datasets.

Scene	Method	PSNR ↑	SSIM ↑	LPIPS ↓
NYC_336	Mip-Splatting	13.389	0.453	0.597
	2DGS	11.014	0.311	0.681
	CityGS-X	14.143	0.393	0.661
	Skyfall-GS	13.497	0.420	0.507
	Ours w/o Image Restoration Network	13.512	0.434	0.491
	Ours	13.894	0.436	0.496
NYC_010	Mip-Splatting	12.022	0.148	0.533
	2DGS	10.899	0.151	0.642
	CityGS-X	12.151	0.140	0.544
	Skyfall-GS	12.184	0.142	0.531
	Ours w/o Image Restoration Network	12.007	0.164	0.538
	Ours	12.513	0.162	0.577
NYC_219	Mip-Splatting	11.230	0.135	0.524
	2DGS	11.151	0.129	0.542
	CityGS-X	11.727	0.135	0.569
	Skyfall-GS	11.686	0.135	0.526
	Ours w/o Image Restoration Network	11.474	0.161	0.534
	Ours	11.901	0.160	0.566

Table A2. Quantitative evaluation of our method compared to prior works on every scene of **GoogleEarth** datasets.

Figure A1. **Results of the MatrixCity-Satellite scene**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery.

Figure A2. **Results of the JAX_004 scene in the DFC 2019 dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery. Results of CityGS-X are removed since the method crashes while recovering this scene.

Figure A3. **Results of the JAX_068 scene in the DFC 2019 dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery. Results of CityGS-X are removed since the method crashes while recovering this scene.

Figure A4. **Results of the JAX_214 scene in the DFC 2019 dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery. Results of CityGS-X are removed since the method crashes while recovering this scene.

Figure A5. **Results of the JAX_260 scene in the DFC 2019 dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery. Results of CityGS-X are removed since the method crashes while recovering this scene.

Figure A6. **Results of the NYC_010 scene in the GoogleEarth dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery.

Figure A7. **Results of the NYC_219 scene in the GoogleEarth dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery.

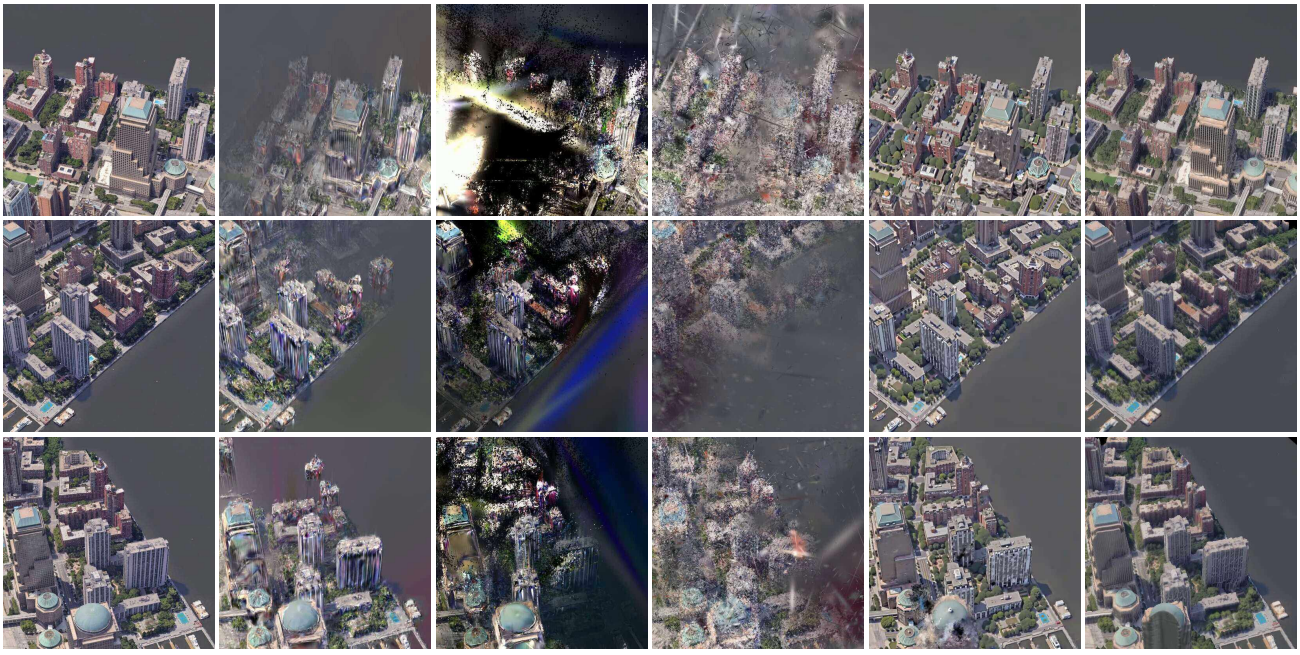


Figure A8. **Results of the NYC_336 scene in the GoogleEarth dataset**. Compared to baselines, our method successfully achieves high-quality city reconstruction from satellite imagery.

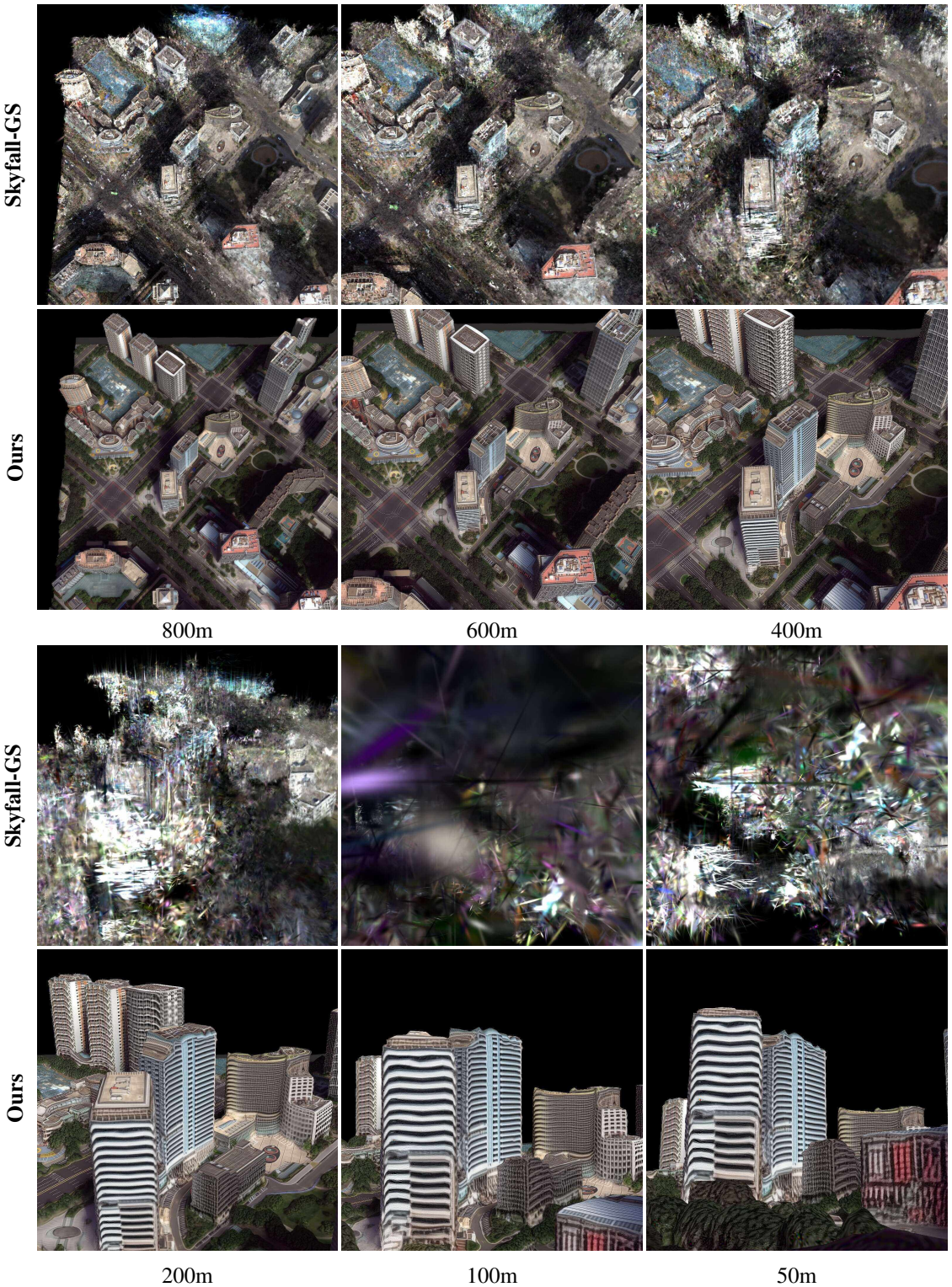
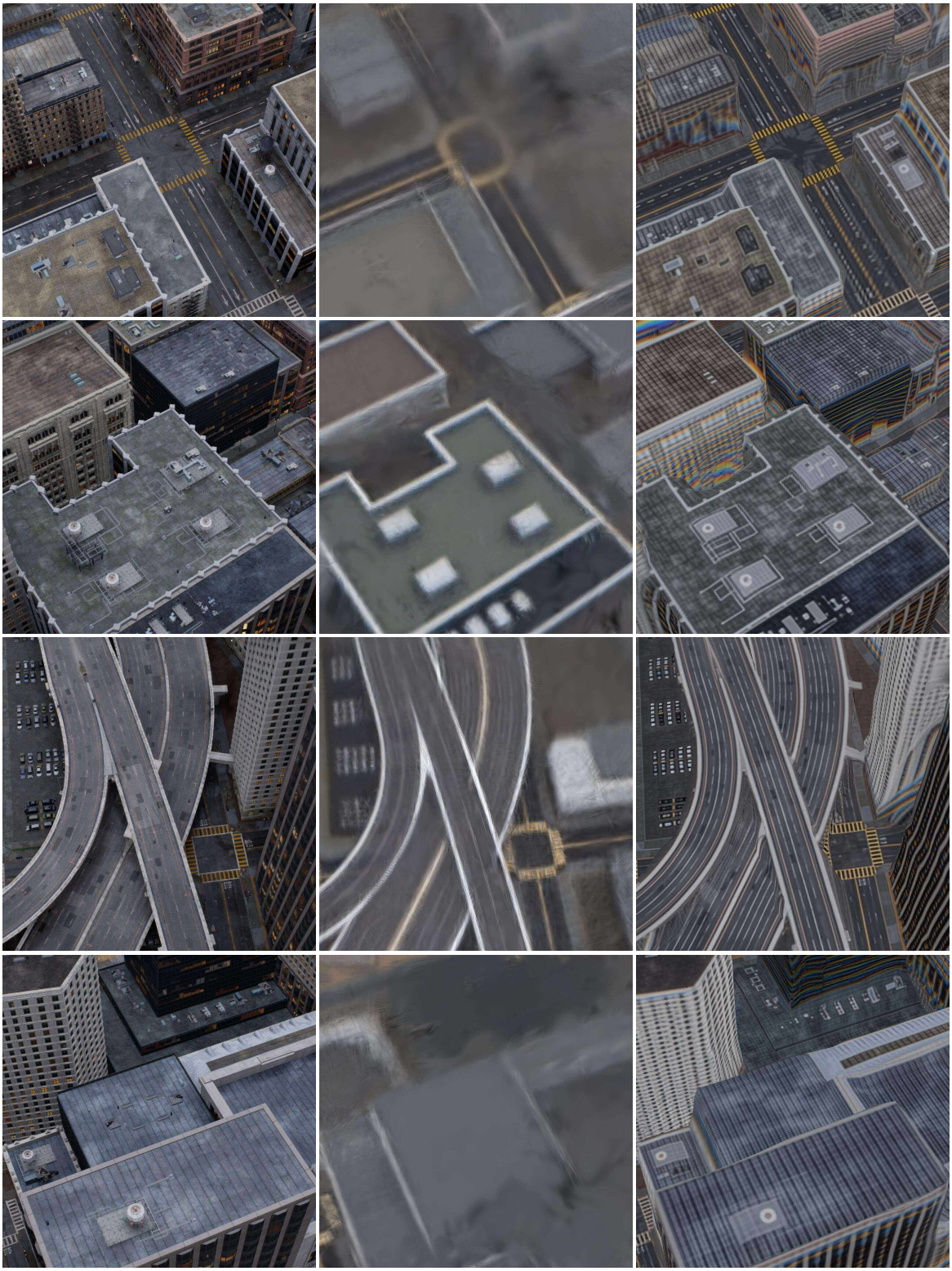


Figure A9. **Qualitative comparison between our method and Skyfall-GS across different camera altitudes**. While our reconstructions may exhibit minor seam artifacts, Skyfall-GS even suffers from substantial quality degradation at lower altitudes.



G.T. Skyfall-GS Ours

Figure A10. **Close-Up views of reconstruction results of the MatrixCity-Satellite scene**. Our method retains small-scale details such as window grids, facade lines, and roof textures significantly better than Skyfall-GS [29], evidencing the benefit of our deterministic diffusion-based texture refinement.

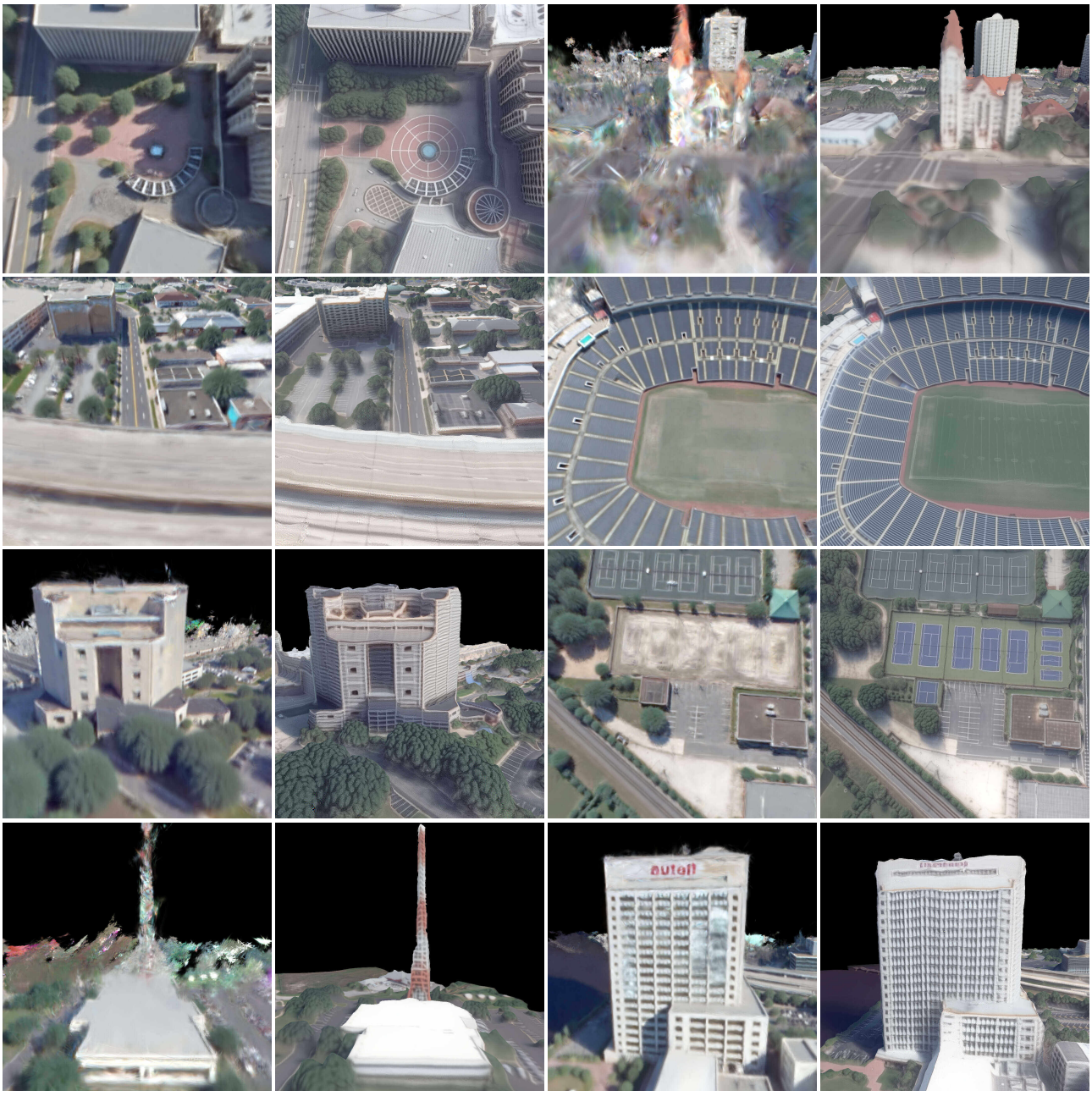


Figure A11. **Close-Up views of reconstruction results of the DFC 2019 datasets**. Our reconstructions preserve fine facade structures and building edges under low-altitude viewpoints, whereas Skyfall-GS [29] often produces blurred or smeared textures, confirming our improved near-view fidelity on real satellite data.

Figure A12. **Close-Up views of reconstruction results of the Google Earth datasets**. Our reconstructions preserve fine facade structures and building edges under low-altitude viewpoints, whereas Skyfall-GS [29] produces smeared textures and even broken parts, confirming our improved near-view fidelity.

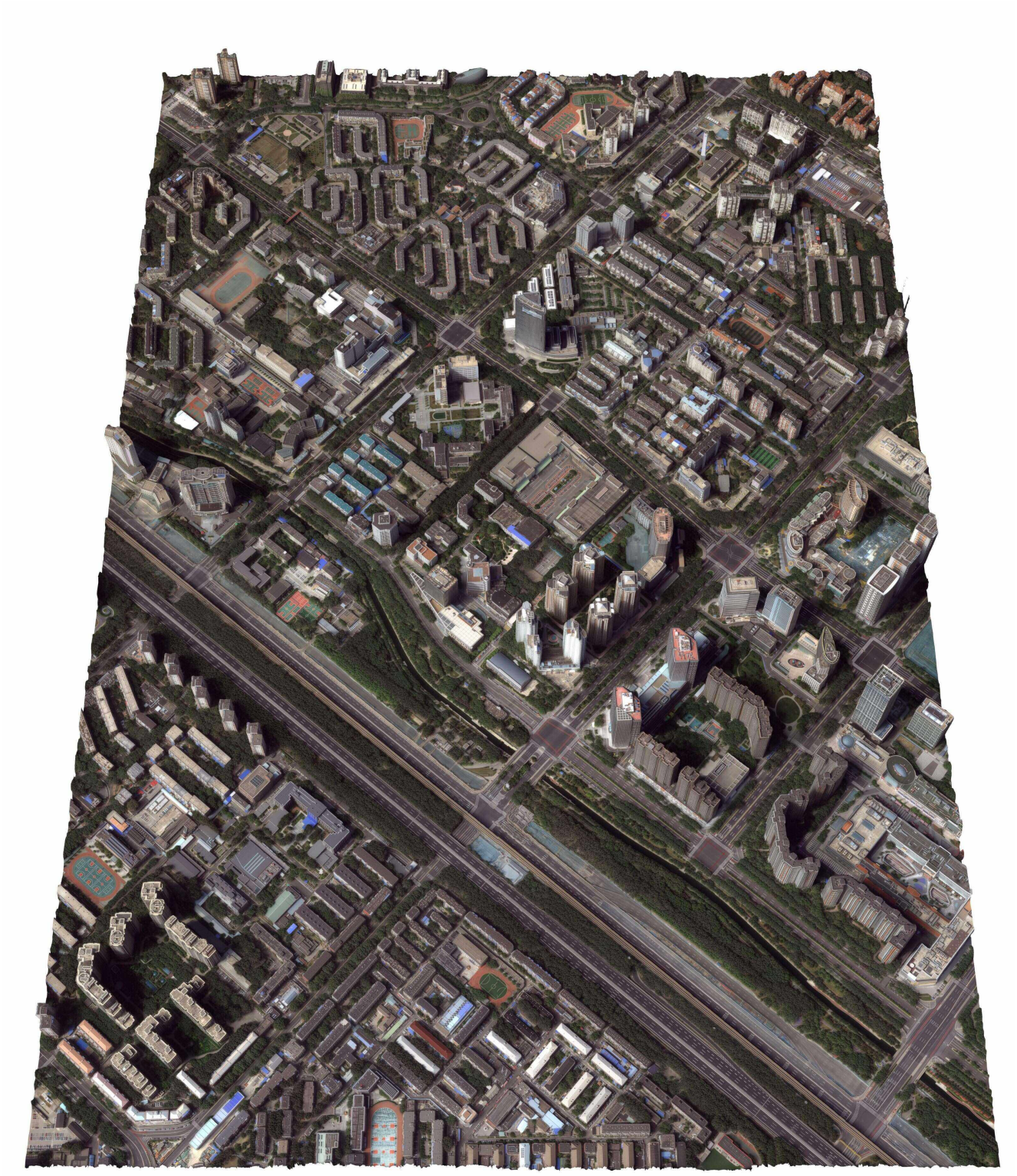


Figure A13. **Bird's-Eye view of a Real Urban Scene Reconstruction**. Our pipeline generalizes to new urban areas without dataset-specific tuning while preserving overall layout, building geometry, and texture plausibility.

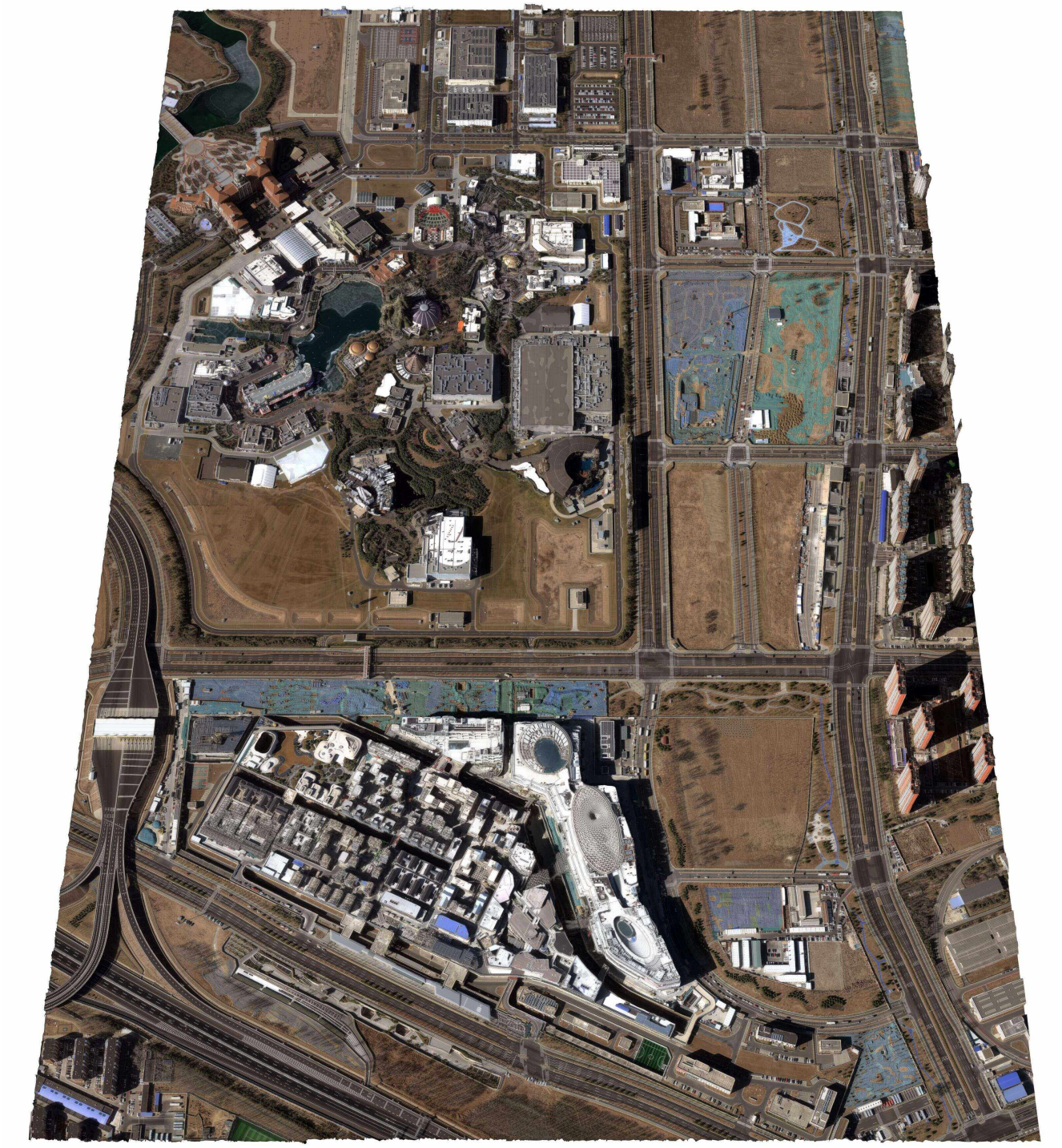


Figure A14. **Bird's-Eye view of a Real Urban Scene Reconstruction**. Our pipeline generalizes to new urban areas without dataset-specific tuning while preserving overall layout, building geometry, and texture plausibility.

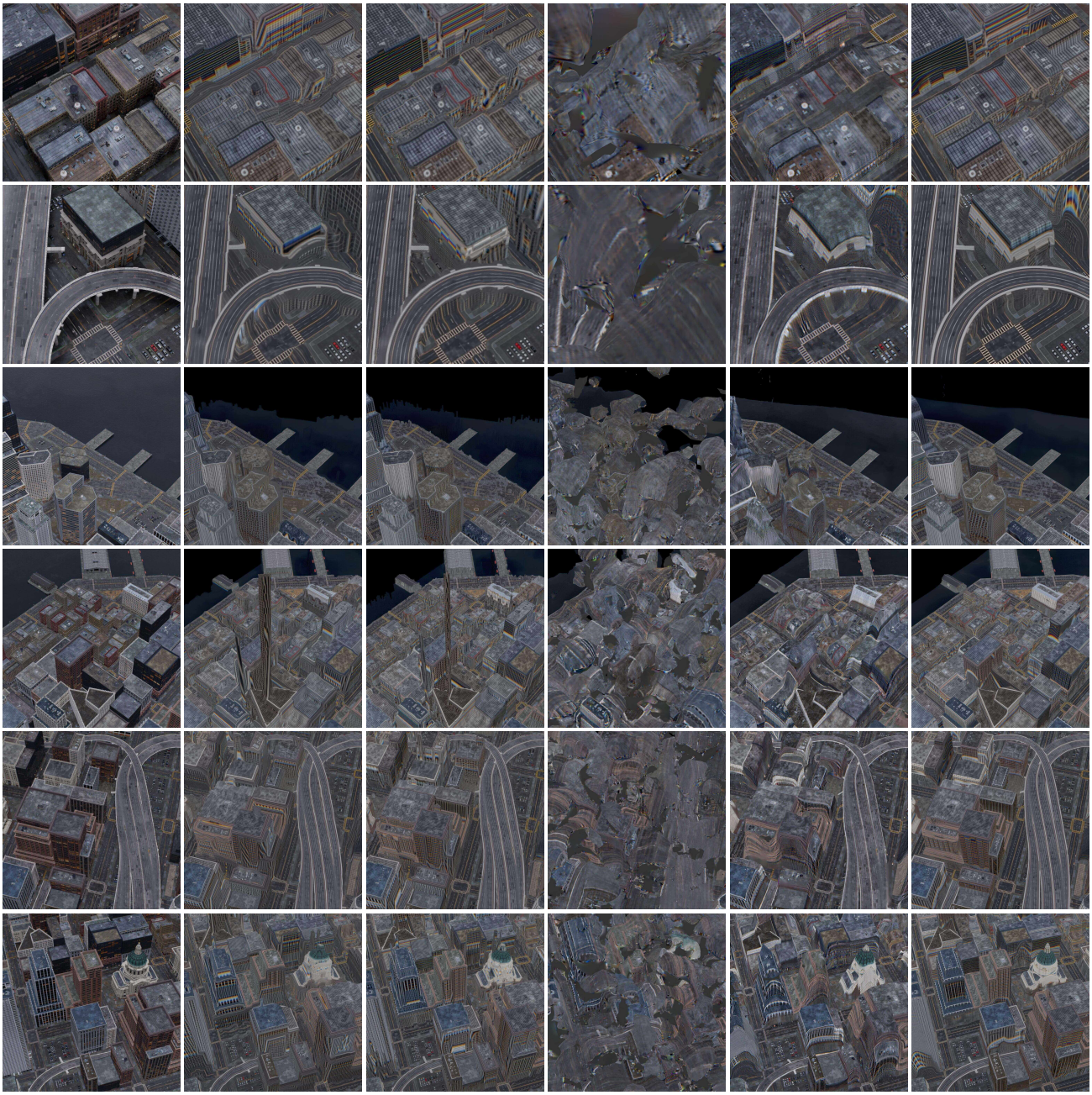


Figure A15. **Visualization of ablation results on geometry**. We compare our full geometry pipeline against variants using low-/high-resolution Marching Cubes, a generic 3D SDF with FlexiCubes, and a Chamfer-distance-only supervision. Only our Z-Monotonic SDF with height-map + regularized training recovers clean roofs, vertical facades, and watertight, artifact-free structures.

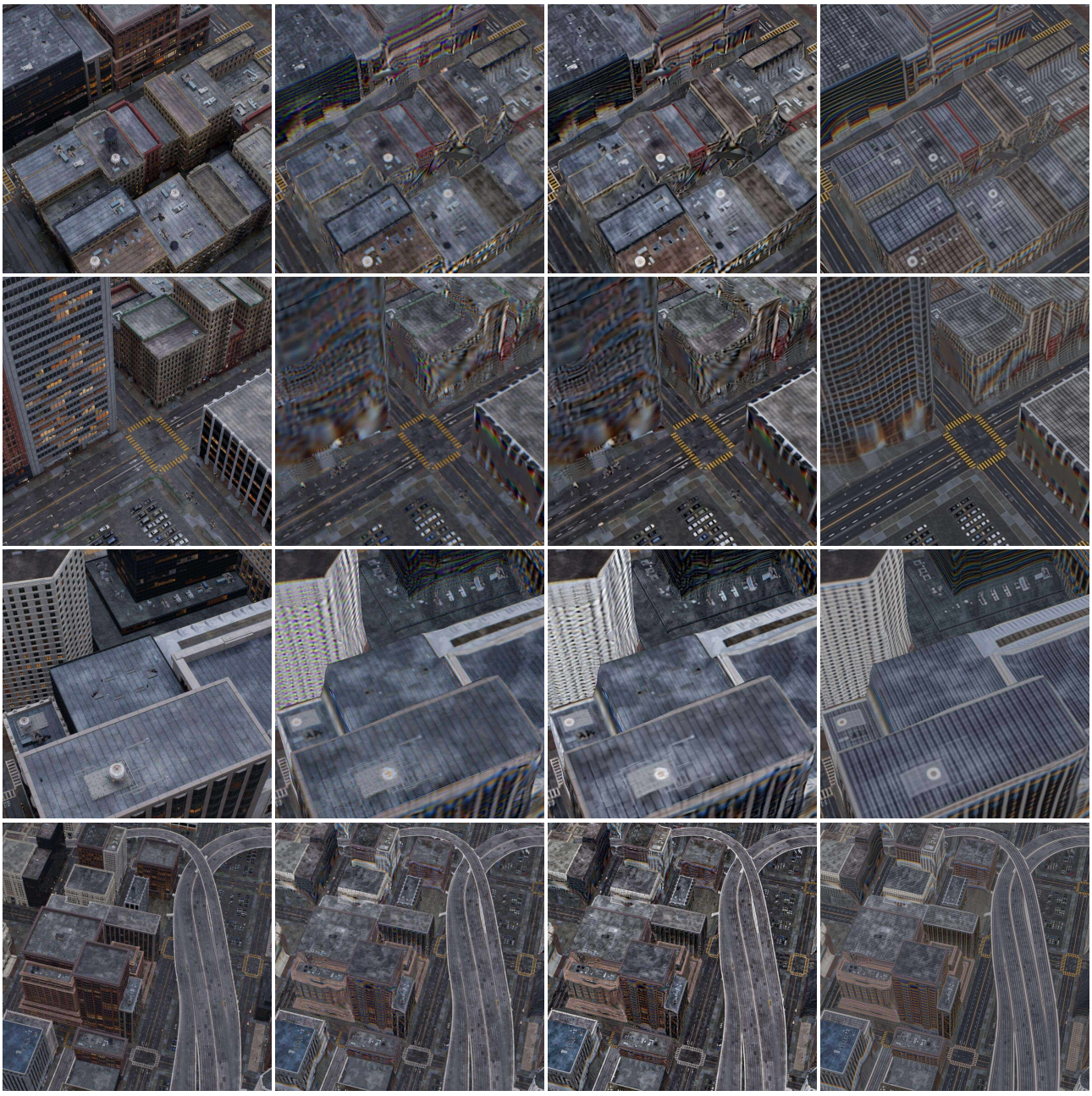


Figure A16. **Visualization of ablation results on appearance modeling**. Disabling image restoration or directly plugging in a generic FLUX-Kontext [28] model produces blurry and inconsistent textures, whereas our fine-tuned, deterministic restorer yields both sharp and view-consistent appearances that best match the ground truth.

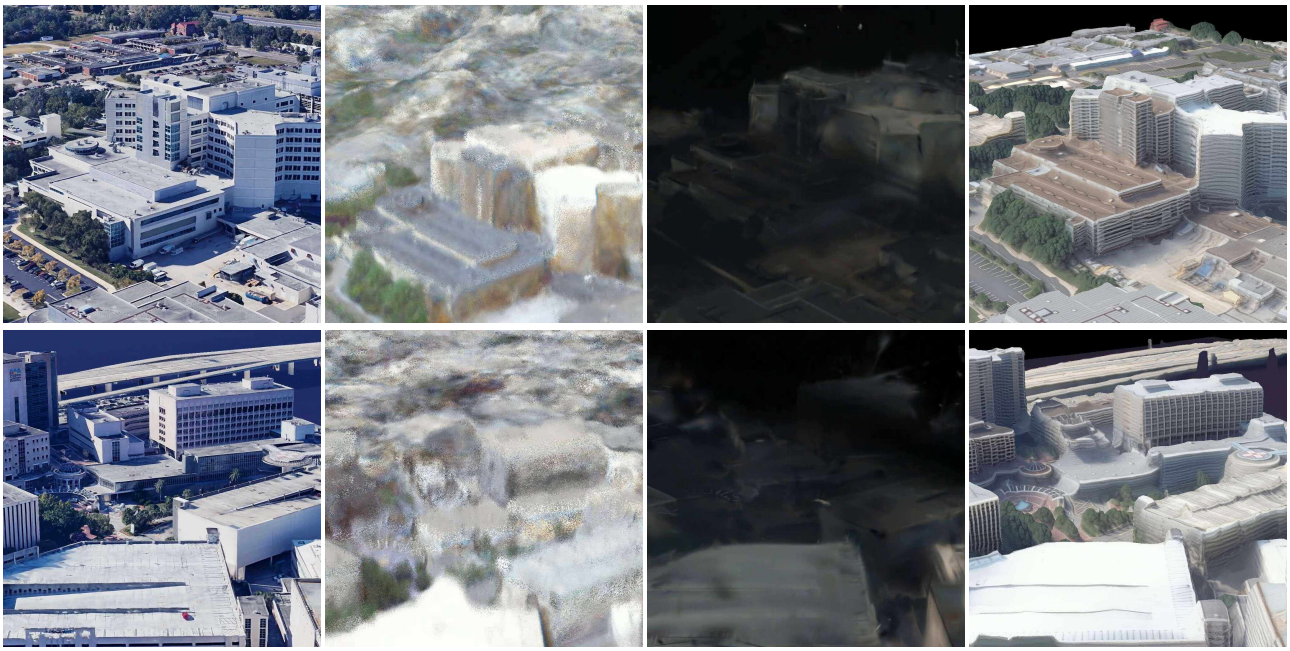
Input

Result

Figure A17. **Overview of aerial-view reconstruction results of our method.** We demonstrate that the same pipeline can be applied to an aerial-view reconstruction task: given a set of oblique aerial images (top), our method reconstructs a geometrically accurate and visually realistic scene (bottom), illustrating its applicability beyond satellite imagery to general photogrammetric settings.

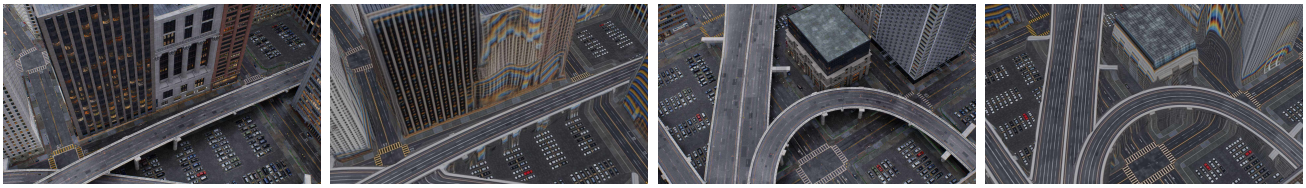
Method	PSNR ↑	SSIM ↑	LPIPS ↓
Sat-NeRF [71]	10.220	0.278	0.816
EOGS [2]	7.338	0.181	0.931
Ours	**13.059**	**0.358**	**0.556**

Table A3. **Quantitative comparison with remote sensing methods on DFC 2019.** Our method achieves higher PSNR and SSIM and lower LPIPS for novel view synthesis compared to Sat-NeRF and EOGS, highlighting its focus on reconstructing visually faithful 3D scene appearance.



G.T. Sat-NeRF EOGS Ours

Figure A18. **Qualitative comparison with remote sensing methods on DFC 2019.** Compared to Sat-NeRF and EOGS, our method produces sharper facades and more realistic textures from oblique and near-ground viewpoints, reflecting our focus on reconstructing the visual appearance of the full 3D scene.



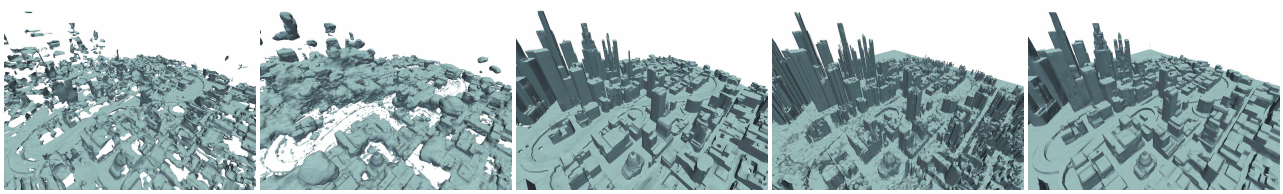
G.T. Ours G.T. Ours

Figure A19. **Failure case on non-monotonic structures.** Non-monotonic scenes from the *MatrixCity-Satellite* dataset where our 2.5D prior fails to capture the multi-layer geometry. We believe that future work could further alleviate these issues.

Method	2DGS	Mip-Splatting	CityGS-X	Skyfall-GS	Ours
CD ↓	33.064	5.785	FAIL	30.345	**3.825**

Table A4. One-sided Chamfer Distance (LiDAR → mesh) on *Urban Scene* after alignment on the overlap region. The lower the better.

Method	NKSR (PC)	LightweightMR (PC / SDF)	NDC (Vox)	ODC (Vox)	Ours
CD ↓	0.0822	0.1461	0.0499	0.0633	**0.0357**



NKSR (PC) L. MR (PC / SDF) NDC (Vox) ODC (Vox) Ours

Figure A20. Mesh-recovery baselines on *MatrixCity-Satellite* dataset with same input. Due to extremely limited wall parallax, MVS points are sparse or empty on walls, making (PC) point or (SDF) SDF-based methods incomplete. (Vox) voxel-based methods face a resolution-sparsity trade-off, often producing holes or artifacts.

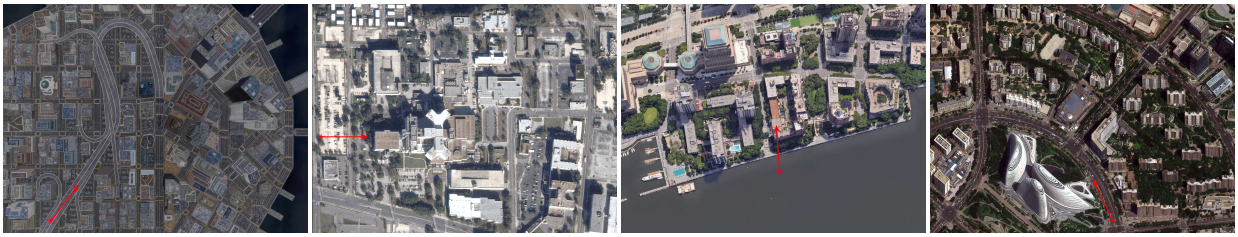
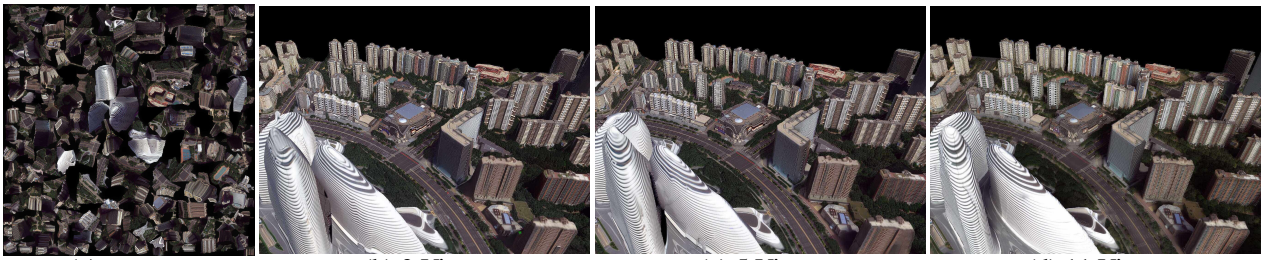


Figure A21. ⋆ denotes viewpoints of Fig. 6 results in the main paper.



(a) UV Map (b) 3 Views (c) 5 Views (d) 11 Views

Figure A22. (a) A UV map of results on the Urban Scene. (b) – (d) Qualitative ablation on input sparsity using 3, 5, and 11 input satellite images, respectively. The quality of our results remains comparable with fewer views.