


City-Mesh3R: Simulation-Ready City-Scale 3D Mesh Reconstruction from Multi-View Images

Supplementary Material

Sayan Paul , Sourav Ghosh, Siddharth Katageri, Soumyadip Maity, Sanjana Sinha, Brojeshwar Bhowmick

{p.sayan, g.sourav10, siddharth.katageri, soumyadip.maity, sanjana.sinha, b.bhowmick} @ tcs.com

Visual Computing & Embodied AI Lab, TCS Research, India

6. Area partitioning of large-scale sparse SfM

6.1. Support-plane parameterization and regular partition construction

This subsection provides the implementation details omitted from the main paper (Sec. 3.2) for constructing the spatial partitions used before camera selection.

Dominant-support-plane parameterization. Let $\mathcal{P} = \{P_p\}$ denote the sparse 3D points reconstructed by COLMAP. We estimate a dominant support plane from \mathcal{P} via RANSAC plane fitting, yielding parameters (\mathbf{n}, d) such that

$$\mathbf{n}^\top \mathbf{x} + d = 0, \quad (9)$$

where $\mathbf{n} \in \mathbb{R}^3$ is a unit normal and $d \in \mathbb{R}$ is the offset. This introduces only a weak geometric prior: unlike Manhattan-world alignment, we assume only one dominant support plane.

From the plane inliers, we construct a local planar frame $(\mathbf{o}, \mathbf{u}, \mathbf{v})$, where \mathbf{o} is a point on the plane and \mathbf{u}, \mathbf{v} are orthonormal in-plane basis vectors. Each sparse point P_p is mapped to planar coordinates by

$$\mathbf{r}_p = P_p - \mathbf{o}, \quad u_p = \mathbf{r}_p^\top \mathbf{u}, \quad v_p = \mathbf{r}_p^\top \mathbf{v}. \quad (10)$$

In-plane orientation refinement. After fixing the plane normal, the in-plane basis remains ambiguous up to a rotation around \mathbf{n} . To reduce this arbitrariness, we choose the in-plane rotation that minimizes the area of the axis-aligned bounding box of the projected sparse points. Let $(u_p^{(\phi)}, v_p^{(\phi)})$ denote the planar coordinates after rotation by angle ϕ . We choose

$$\phi^* = \arg \min_{\phi} \left(\max_p u_p^{(\phi)} - \min_p u_p^{(\phi)} \right) \left(\max_p v_p^{(\phi)} - \min_p v_p^{(\phi)} \right). \quad (11)$$

Regular grid partitioning with inflated windows. Using the rotated planar coordinates, we define a regular $R \times C$ grid over the support-plane extent. Let

$$[u_{\min}, u_{\max}] \times [v_{\min}, v_{\max}] \quad (12)$$

be the planar bounding box. The base grid boundaries are

$$u_c = u_{\min} + \frac{c}{C}(u_{\max} - u_{\min}), \quad v_r = v_{\min} + \frac{r}{R}(v_{\max} - v_{\min}), \quad (13)$$

for $c = 0, \dots, C$ and $r = 0, \dots, R$.

For each grid cell (r, c) with base rectangle $[u_c, u_{c+1}] \times [v_r, v_{r+1}]$, we enlarge its support around the cell center

$$u_{r,c}^{(m)} = \frac{u_c + u_{c+1}}{2}, \quad v_{r,c}^{(m)} = \frac{v_r + v_{r+1}}{2}. \quad (14)$$

Given inflation factors α_u and α_v , the enlarged side lengths are

$$\Delta u' = (1 + \alpha_u)(u_{c+1} - u_c), \quad \Delta v' = (1 + \alpha_v)(v_{r+1} - v_r). \quad (15)$$

The inflated window becomes

$$\hat{u}_0 = u_{r,c}^{(m)} - \frac{\Delta u'}{2}, \quad \hat{u}_1 = u_{r,c}^{(m)} + \frac{\Delta u'}{2}, \quad (16)$$

$$\hat{v}_0 = v_{r,c}^{(m)} - \frac{\Delta v'}{2}, \quad \hat{v}_1 = v_{r,c}^{(m)} + \frac{\Delta v'}{2}, \quad (17)$$

followed by clipping to the global planar extent. The point set assigned to partition (r, c) is

$$\mathcal{P}_{r,c} = \{P_p \in \mathcal{P} \mid \hat{u}_0 \leq u_p \leq \hat{u}_1, \hat{v}_0 \leq v_p \leq \hat{v}_1\}. \quad (18)$$

These partition point sets are used by the camera-ranking procedure described next.

6.2. Per-partition geometry-aware camera ranking

This subsection gives the full formulation of the partition-wise camera ranking summarized in the main paper (Sec. 3.2).

For each non-empty partition (r, c) , let $\mathcal{P}_{r,c}$ denote the partition point set. We first form the candidate camera set as the union of all cameras that observe at least one partition point:

$$\mathcal{I}_{r,c}^{\text{cand}} = \bigcup_{p \in \mathcal{P}_{r,c}} \mathcal{I}(p). \quad (19)$$

If $\mathcal{P}_{r,c} = \emptyset$, or if no sufficiently co-visible camera pairs exist within the partition, we return an empty ranked set.

Admissible camera pairs. For each unordered camera pair (i, j) , let m_{ij} be the number of points in $\mathcal{P}_{r,c}$ observed by both cameras, and let $N_{r,c} = |\mathcal{P}_{r,c}|$. We retain only admissible pairs whose co-visibility ratio exceeds a threshold:

$$\rho_{ij} = \frac{m_{ij}}{N_{r,c}}, \quad \mathcal{A} = \{(i, j) \mid \rho_{ij} \geq \tau_{\text{cov}}\}. \quad (20)$$

Pair prior. For each admissible pair $(i, j) \in \mathcal{A}$, we define a prior that captures viewpoint complementarity and local support. Let θ_{ij} denote the relative rotation angle between cameras i and j . We favor moderate relative orientation using

$$P_{\text{rot}}(i, j) = \exp\left(-\frac{(\theta_{ij} - \mu_{\text{rot}})^2}{2\sigma_{\text{rot}}^2}\right). \quad (21)$$

We also map the pairwise co-visibility count to a normalized overlap term

$$P_{\text{overlap}}(i, j) = \frac{\log(1 + m_{ij})}{\log(1 + m_{\max})}, \quad m_{\max} = \max_{(i,j) \in \mathcal{A}} m_{ij}. \quad (22)$$

The resulting pair prior is

$$\text{prior}_{ij} = P_{\text{rot}}(i, j)^{w_{\text{rot}}} \cdot P_{\text{overlap}}(i, j)^{w_{\text{overlap}}}. \quad (23)$$

Point-conditioned pair scoring. For each point $p \in \mathcal{P}_{r,c}$, we consider only admissible pairs that observe it:

$$\mathcal{A}(p) = \{(i, j) \in \mathcal{A} \mid i, j \in \mathcal{I}(p)\}. \quad (24)$$

If $\mathcal{A}(p) = \emptyset$, the point contributes no vote.

Let C_i and C_j be the camera centers, and let P_p be the 3D point. Using the normalized viewing rays from C_i and C_j toward P_p , we compute the triangulation angle $\alpha_{ij}(p)$ and define

$$T_{ij}(p) = \exp\left(-\frac{(\alpha_{ij}(p) - \mu_{\text{triang}})^2}{2\sigma_{\text{triang}}^2}\right). \quad (25)$$

We also project P_p into images i and j , compute the normalized radial distances $r_i(p)$ and $r_j(p)$ from the principal points, and define centrality weights

$$C_i(p) = \exp\left(-\frac{r_i(p)^2}{2\sigma_{\text{cent}}^2}\right), \quad C_j(p) = \exp\left(-\frac{r_j(p)^2}{2\sigma_{\text{cent}}^2}\right). \quad (26)$$

These are combined symmetrically as

$$C_{ij}(p) = \sqrt{C_i(p) C_j(p)}. \quad (27)$$

The final point-pair score is

$$s_{p,ij} = \text{prior}_{ij} \cdot T_{ij}(p)^{w_{\text{triang}}} \cdot C_{ij}(p)^{w_{\text{cent}}}. \quad (28)$$

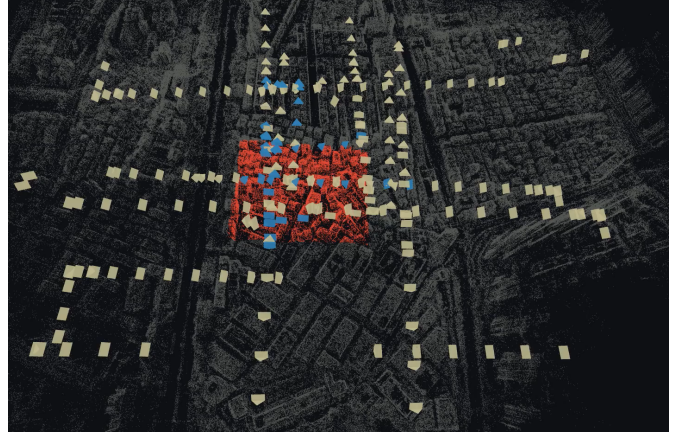


Figure 5. **Area Partitioning Visualizer:** The selected partition’s sparse points are highlighted by orange and all other points are shown in gray. All the candidate cameras of a partition are denoted by yellow and the top-M ranked cameras are denoted by blue.

Camera score aggregation. Initialize $S_i = 0$ for all $i \in \mathcal{I}_{r,c}^{\text{cand}}$. To avoid over-counting many near-equivalent pairs for one point, we retain only the top- K scoring pairs $\tilde{\mathcal{A}}(p) \subseteq \mathcal{A}(p)$ and normalize their scores:

$$w_{p,ij} = \frac{s_{p,ij}}{\sum_{(i,j) \in \tilde{\mathcal{A}}(p)} s_{p,ij} + \varepsilon}. \quad (29)$$

Each selected pair casts a unit-normalized vote split equally between its two cameras:

$$S_i \leftarrow S_i + \frac{1}{2} w_{p,ij}, \quad S_j \leftarrow S_j + \frac{1}{2} w_{p,ij}, \quad (i, j) \in \tilde{\mathcal{A}}(p). \quad (30)$$

Aggregating these votes over all points yields a partition-specific score for every candidate camera. We then sort cameras by S_i and retain the top M :

$$\mathcal{I}_{r,c}^{\text{top}} = \text{TopM}\left(\{S_i\}_{i \in \mathcal{I}_{r,c}^{\text{cand}}}, M\right). \quad (31)$$

7. Dense Reconstruction and Surface Initialization

This section provides the detailed formulation for the dense reconstruction and surface initialization module summarized in Section 3.2.

For each partitioned area, we start from its sparse SfM reconstruction and retain the top- M cameras from the ranking stage. Their intrinsics and poses, $\{K_n, R_n, t_n\}_{n=1}^M$, are kept fixed throughout. We then densify the partition using the depth predictions and pixel correspondences produced by MAST3R. Since MAST3R inference has already been run on image pairs in the original similarity graph, we reuse the cached outputs whenever possible; only pairs newly induced by the final partitioning require fresh inference.

Let \mathcal{V} denote the selected views and \mathcal{E} the set of view pairs for which MAST3R correspondences are available after this cache-or-infer step. For each view $n \in \mathcal{V}$, let D_n denote the predicted depth map, and let $\mathcal{M}^{n,m}$ be the set of matched pixels between views n and m . For a correspondence $u_c^n \leftrightarrow u_c^m$, we define the associated world-space dense points by back-projecting each pixel using the fixed SfM camera:

$$X_c^n = \Pi_n^{-1}(u_c^n, D_n(u_c^n)), \quad X_c^m = \Pi_m^{-1}(u_c^m, D_m(u_c^m)).$$

Thus, X_c^n is the 3D point induced by pixel u_c^n from the predicted depth of view n , expressed in the common SfM world frame. Equivalently, the collection $\{X_c^n\}_c$ can be interpreted as the dense pointmap of view n in world coordinates, consistent with the pointmap representation used in MAST3R.

Although the cameras are fixed, these per-view dense reconstructions are not necessarily globally consistent: they may still suffer from residual scale drift across views and local depth noise. To address this, we adopt a two-stage alignment strategy inspired by MAST3R-SfM, but specialized to our setting by optimizing only scene geometry while keeping all camera parameters frozen.

Coarse alignment of Depth Maps: We first assign each view a global scale $s_n > 0$ and rescale its dense points as $\tilde{X}_c^n = s_n X_c^n$. The scales are estimated by minimizing the discrepancy between matched 3D points:

$$\{s_n^*\} = \arg \min_{\{s_n\}} \sum_{(n,m) \in \mathcal{E}} \sum_{c \in \mathcal{M}^{n,m}} q_c \|s_n X_c^n - s_m X_c^m\|_2^{\lambda_1}, \quad (32)$$

where q_c denotes the confidence of match c , and $0 < \lambda_1 \leq 1$ defines a robust penalty. To remove the gauge ambiguity, we normalize the estimated scales as

$$s_n = s'_n / \min_k s'_k.$$

This stage allows each view to expand or contract globally so that corresponding dense points become metrically compatible across overlapping views.

Depth Refinement: A single scale per view is insufficient to correct local depth errors. We therefore further refine the geometry by directly optimizing the dense 3D points while keeping the cameras fixed. Let \hat{X}_c^n denote the refined 3D points. We optimize them by minimizing the bidirectional reprojection inconsistency:

$$\{\hat{X}_c^n\} = \arg \min_{\{\hat{X}_c^n\}} \sum_{(n,m) \in \mathcal{E}} \sum_{c \in \mathcal{M}^{n,m}} q_c [\rho(u_c^n - \Pi_n(\hat{X}_c^m)) + \rho(u_c^m - \Pi_m(\hat{X}_c^n))], \quad (33)$$

where $\Pi_n(\cdot)$ is the projection operator of the fixed camera n , and $\rho(x) = \|x\|_2^{\lambda_2}$ with $0 < \lambda_2 \leq 1$ is a robust reprojection penalty. This objective is analogous to bundle adjustment, except that only the scene geometry is optimized. The coarse stage removes dominant inter-view scale mismatch, while the refinement stage suppresses local geometric noise by enforcing multi-view reprojection consistency.

Both stages are optimized with Adam for a fixed number of iterations, yielding aligned dense depth maps for the partition.

Volumetric fusion and watertight initialization. The aligned depth maps are fused using TSDF integration, which aggregates multi-view evidence and regularizes local inconsistencies through volumetric averaging. From the TSDF grid, we extract a fused global point cloud with normals. These oriented points are then passed to Screened Poisson Surface Reconstruction to obtain an initial watertight mesh, which is used to initialize the subsequent differentiable-rendering-based mesh refinement stage.

In summary, this module converts fixed sparse-SfM cameras and mostly cached MAST3R predictions into a dense, globally aligned, and denoised surface initialization, without re-estimating camera poses.

8. Differentiable Rendering based Mesh Refinement

This section provides the full formulation and implementation details omitted from Sec. 3.2.

8.1. Rendering Objective

At iteration k , the current mesh is $M^k = (V^k, F^k)$. Given calibrated cameras $\{K_j, T_{wc,j}\}_{j=1}^N$, we optimize

$$\Phi(M^k) = \sum_{j=1}^N \left(\lambda_n \mathcal{L}_n^{(j)}(M^k) + \lambda_s \mathcal{L}_{\text{sil}}^{(j)}(M^k) \right) + \mathcal{R}(M^k), \quad (34)$$

where \mathcal{R} is a lightweight mesh regularizer, e.g. Laplacian smoothing.

For view j , let $S_j(u) \in [0, 1]$ be the target silhouette and $\hat{S}_j(u; M^k)$ the rendered silhouette. The silhouette loss is

$$\mathcal{L}_{\text{sil}}^{(j)}(M^k) = \frac{1}{|\Omega_{\text{img}}|} \sum_{u \in \Omega_{\text{img}}} (\hat{S}_j(u; M^k) - S_j(u))^2. \quad (35)$$

Let $\mathcal{N}_j(u) \in \mathbb{S}^2$ be the predicted unit normal map in camera coordinates and $\hat{\mathcal{N}}_j(u; M^k)$ the rendered unit normal map. On the foreground support $\Omega_j = \{u \mid S_j(u) = 1\}$, the normal-map loss is

$$\mathcal{L}_n^{(j)}(M^k) = \frac{1}{|\Omega_j|} \sum_{u \in \Omega_j} \left(1 - \hat{\mathcal{N}}_j(u; M^k)^\top \mathcal{N}_j(u) \right). \quad (36)$$

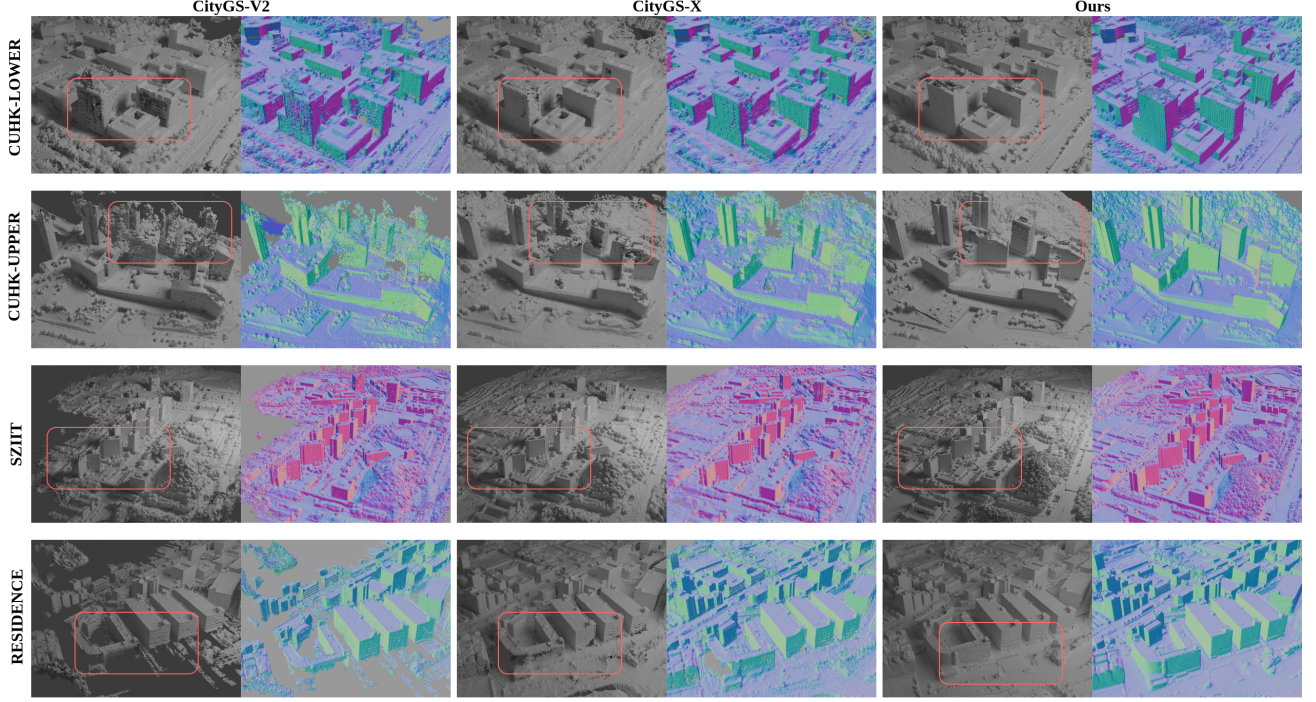


Figure 6. Extra Qualitative Comparison Results with recent city-scale surface reconstruction methods (CityGS-v2 and CityGS-X) on GauU-Scene dataset (CUHK-LOWER, CUHK-UPPER, SZIT) and UrbanScene3D dataset (Residence). Go to Fig. 3 for Main Results.

Gradients of Φ are backpropagated through the renderer to vertex positions, which are updated using the IsotropicAdam optimizer of Continuous Remeshing. As in the original framework, the optimizer also provides a relative vertex velocity $\nu^k(i) \in [0, 1]$, which measures the normalized update magnitude at vertex i and is used as a local indicator of stabilization.

8.2. Curvature-Guided Reference Field

The original controller tends to distribute resolution relatively uniformly, which is inefficient for large scenes: flat regions consume unnecessary triangles, while detailed regions remain under-sampled. We instead decouple *where* fine resolution is ultimately needed from *when* that refinement is allowed.

For each view j , let $\mathbf{n}^{(j)}(u, v) \in \mathbb{S}^2$ be the unit normal field, with image-space Jacobian

$$N^{(j)}(u, v) = [\partial_u \mathbf{n}^{(j)}(u, v) \quad \partial_v \mathbf{n}^{(j)}(u, v)] \in \mathbb{R}^{3 \times 2}. \quad (37)$$

We define

$$M^{(j)}(u, v) = (N^{(j)}(u, v))^\top N^{(j)}(u, v) \in \mathbb{R}^{2 \times 2}, \quad (38)$$

and the worst-case local normal-rotation rate

$$s^{(j)}(u, v) = \sqrt{\lambda_{\max}(M^{(j)}(u, v))} = \|N^{(j)}(u, v)\|_2. \quad (39)$$

Given a normal-rotation tolerance θ_0 (radians), the target projected edge length in pixels is

$$p_{\text{tgt}}^{(j)}(u, v) = \text{clip}\left(\frac{\theta_0}{s^{(j)}(u, v)}, p_{\min}, p_{\max}\right). \quad (40)$$

Hence, regions with rapid normal variation receive smaller projected target edges, while smooth regions permit coarser sampling.

We then lift this per-view pixel-domain target to the mesh surface. For a vertex i visible in view j , let $\pi^{(j)}(\mathbf{v}_i)$ be its projection and $z_i^{(j)}$ its camera-space depth. Sampling the pixel target gives $p_{ij} = p_{\text{tgt}}^{(j)}(\pi^{(j)}(\mathbf{v}_i))$, which is converted to world units as

$$L_{ij} = \frac{z_i^{(j)}}{f^{(j)}} p_{ij}, \quad (41)$$

where $f^{(j)}$ is the focal length in pixels. Pooling across visible views \mathcal{V}_i yields the per-vertex curvature-guided baseline

$$L_{\text{ref-curv}}(i) = \text{median}_{j \in \mathcal{V}_i} L_{ij}. \quad (42)$$

In practice, remeshing acts on edge targets derived from the vertex field by endpoint averaging. This lets us maintain the reference field on vertices while applying split/collapse decisions on edges.

8.3. Speed-Aware Slack Schedule

To control when refinement is released, we introduce a non-negative slack $\zeta^k(i)$ on top of the curvature-guided baseline. The slack is initialized as

$$\zeta^0(i) \leftarrow \max(0, L_{\text{curr}}^0(i) - L_{\text{ref-curv}}^0(i)), \quad (43)$$

where $L_{\text{curr}}^0(i)$ denotes the current local edge scale at initialization.

The slack is then updated using the optimizer’s relative vertex velocity:

$$\zeta^k(i) \leftarrow \max(0, \zeta^{k-1}(i) + g(\nu^k(i) - \nu_{\text{ref}})), \quad (44)$$

where $g > 0$ is a gain and ν_{ref} is a reference speed. The resulting per-vertex reference edge length is

$$L_{\text{ref}}^k(i) = \text{clip}(L_{\text{ref-curv}}^k(i) + \zeta^k(i), L_{\min}, L_{\max}). \quad (45)$$

The corresponding per-edge reference length is obtained from the two incident vertex values. Intuitively, curvature specifies where fine resolution is eventually required, while velocity controls when the local mesh is allowed to approach that target.

8.4. Automatic Estimation of Global Length Bounds

The global bounds L_{\min} and L_{\max} are estimated automatically from camera intrinsics and a robust scene-depth statistic. Let $f_x = K_{00}$, $f_y = K_{11}$, and $f_{\text{iso}} = \sqrt{f_x f_y}$. Flatten all valid rendered depths of the initial mesh and let z_p be a robust percentile. Then

$$\text{wlpp} = \frac{z_p}{f_{\text{iso}}}, \quad L_{\min} = \text{wlpp}. \quad (46)$$

If \mathcal{E} denotes the set of unique mesh edges and

$$L_{\text{med}} = \text{median}_{(i,m) \in \mathcal{E}} \|\mathbf{v}_i - \mathbf{v}_m\|_2, \quad (47)$$

we set

$$L_{\max} = \max(k_1 L_{\text{med}}, k_2 L_{\min}), \quad (48)$$

with $k_1, k_2 > 1$. This prevents both sub-pixel triangles and overly coarse edges.

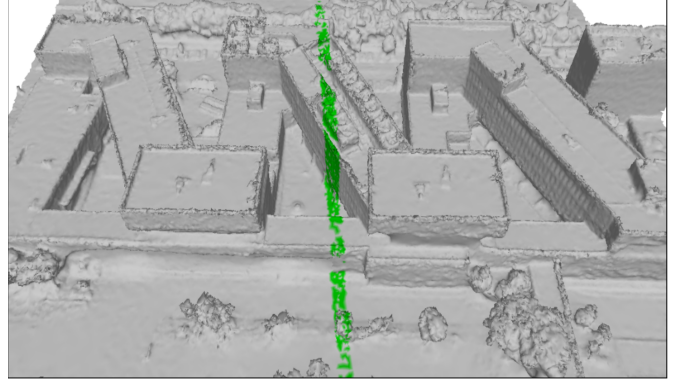
8.5. Remeshing Rules

After each vertex update, we invoke the remesher under the current reference field L_{ref}^k . As in Continuous Remeshing, remeshing is embedded directly into the optimization loop and consists of local **collapse**, **split**, and **flip** operations. Let $\ell(e)$ denote the current length of edge e . Edges are collapsed when

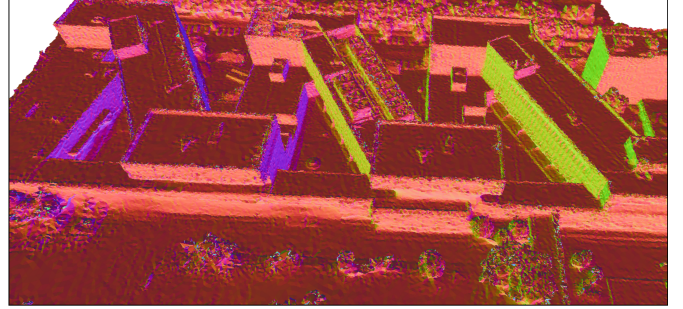
$$\ell(e) < \tau_c L_{\text{ref}}^k(e), \quad (49)$$

split when

$$\ell(e) > \tau_s L_{\text{ref}}^k(e), \quad (50)$$



(A)



(B)

Figure 7. **Mesh Stitching Result:** (A) Two adjacent partition meshes with seam-region points highlighted in green. (B) Surface normals of the final stitched mesh. Our seam stitching yields a topologically consistent and geometrically smooth merge, preserving clean boundaries and normal continuity across partitions.

and flipped when doing so improves local triangle quality and valence regularity, subject to the same geometric validity checks as in the original framework. Thus, the remesher keeps local edge lengths near the target field while preserving a well-shaped triangulation.

The overall refinement alternates $\text{render} \rightarrow \text{compare} \rightarrow \text{update} \rightarrow \text{remesh}$, and the final mesh is selected as the best snapshot $M^* = \arg \min_k \Phi(M^k)$.

9. Mesh Quality Metrics

We assess mesh quality using the following metrics, which quantify complementary aspects of geometric validity, topological consistency, and structural regularity.

Aspect Ratio (AR). For a triangular face $f \in \mathcal{F}$, let h_f denote its longest edge length and r_f its inradius. The face-wise aspect ratio is defined as

$$\text{AR}(f) = \frac{h_f}{2r_f}, \quad (51)$$

Metric	Ours	MiLo	MeshSplatting	RadianceMesh
	488K V · 973K F	1384K V · 2771K F	1190K V · 3503K F	483K V · 1101K F
Aspect Ratio (AR) ↓	3.025	15.652	5.224	3.147
Angle bad ratio (ANG) ↓	1.92%	28.83%	15.25%	2.93%
Degenerate Triangle Ratio (DTR) ↓	0.0000	0.0003	0.0000	0.0000
Non-Manifold Edge Ratio (NME) ↓	0.0039%	0.0000%	27.34%	1.28%
Non-Manifold Vertex Ratio (NMV) ↓	0.0027%	0.0014%	31.34%	6.26%
Vertex Valence Deviation (VVD) ↓	0.505	1.421	3.045	1.914
Connected Components (CC) ↓	9	918	340	232
Interior Boundary Loops (IBL) ↓	24	61	6554	58

Table 6. Mesh Quality Metric Comparison across four reconstruction methods on the Garden scene of the MipNeRF360 dataset. Best values are highlighted in **green** and bold, second-best in **blue**. Lower is better for all metrics. For the metric definitions, please refer to Sec. 9.

and the mesh-level aspect ratio is given by

$$\text{AR} = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \text{AR}(f). \quad (52)$$

Lower values indicate better-shaped triangles.

Angle Bad Ratio (ANG). Let $\theta_{\min}(f)$ and $\theta_{\max}(f)$ denote the minimum and maximum interior angles of face f , respectively. Given angular thresholds θ_{lo} and θ_{hi} , the angle bad ratio is defined as

$$\text{ANG} = \frac{|\{f \in \mathcal{F} \mid \theta_{\min}(f) < \theta_{\text{lo}} \text{ or } \theta_{\max}(f) > \theta_{\text{hi}}\}|}{|\mathcal{F}|}. \quad (53)$$

This metric measures the fraction of triangles with poor angular quality. Lower values are preferred.

Degenerate Triangle Ratio (DTR). Let $A(f)$ denote the area of face $f \in \mathcal{F}$, and let $\varepsilon_A > 0$ denote a small numerical threshold. The degenerate triangle ratio is defined as

$$\text{DTR} = \frac{|\{f \in \mathcal{F} \mid A(f) \leq \varepsilon_A\}|}{|\mathcal{F}|}. \quad (54)$$

This metric quantifies the proportion of triangles that are degenerate or nearly degenerate. Lower values are better.

Non-Manifold Edge Ratio (NME). Let \mathcal{E} denote the set of mesh edges, and let $\deg(e)$ denote the number of incident faces of edge $e \in \mathcal{E}$. The non-manifold edge ratio is defined as

$$\text{NME} = \frac{|\{e \in \mathcal{E} \mid \deg(e) > 2\}|}{|\mathcal{E}|}. \quad (55)$$

Lower values indicate better topological consistency.

Non-Manifold Vertex Ratio (NMV). Let \mathcal{V} denote the set of mesh vertices. The non-manifold vertex ratio is defined as

$$\text{NMV} = \frac{|\{v \in \mathcal{V} \mid v \text{ is non-manifold}\}|}{|\mathcal{V}|}. \quad (56)$$

Lower values indicate fewer vertex-level topological defects.

Vertex Valence Deviation (VVD). For each vertex $v \in \mathcal{V}$, let $\text{val}(v)$ denote its valence, i.e., the number of one-ring neighboring vertices. We define vertex valence deviation as

$$\text{VVD} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} |\text{val}(v) - 6|. \quad (57)$$

This metric measures deviation from regular triangular connectivity. Lower values indicate a more regular tessellation.

Connected Components (CC). Let \mathcal{C} denote the set of connected components of the mesh under standard edge-based connectivity. We define

$$\text{CC} = |\mathcal{C}|. \quad (58)$$

Lower values indicate a more globally coherent mesh, with $\text{CC} = 1$ corresponding to a single connected surface.

Interior Boundary Loops (IBL). Let \mathcal{C}_{\max} be the largest connected component of the mesh, and let $\mathcal{B}(\mathcal{C}_{\max})$ denote its boundary loops. For outdoor scene meshes, which are typically open surfaces, we treat the longest loop as the intended exterior boundary of the dominant surface, and define

$$\text{IBL} = \max(|\mathcal{B}(\mathcal{C}_{\max})| - 1, 0). \quad (59)$$

IBL counts the remaining boundary loops after excluding this primary exterior boundary. Lower values indicate fewer residual open-boundary artifacts, such as unintended holes, tears, or missing patches. It does not capture handle- or genus-type topological holes that do not induce boundaries.

9.1. Quantitative Results

As shown in Table 6, our method achieves the best overall mesh quality. It ranks first in aspect ratio (3.025), angle bad ratio (1.92%), vertex valence deviation (0.505), connected components (9), and interior boundary loops (24), while also achieving zero degenerate triangles. This indicates better triangle quality, more regular tessellation, and markedly stronger global coherence than the baselines.

Although MiLo attains slightly lower non-manifold edge and vertex ratios, it remains highly fragmented (918 connected components, 61 interior boundary loops). RadianceMesh also achieves zero degenerate triangles, but is substantially worse in connectivity and boundary quality (232 connected components, 58 interior boundary loops). MeshSplatting further achieves zero degenerate triangles, yet suffers from severe topological defects, with very high non-manifold ratios (27.34% NME, 31.34% NMV), 340 connected components, and 6554 interior boundary loops.

These results show that our method yields the most structurally clean and simulation-ready meshes among the compared approaches.

10. Limitations

Although our pipeline is highly scalable and produces clean, simulation-ready meshes, a few practical constraints remain. Our method inherits any residual errors from upstream SfM or pretrained depth prediction models, and extremely texture-poor or reflective regions can still challenge local reconstruction quality. Seam merging is robust, but very fine details at partition boundaries may experience mild smoothing. Finally, as clusters are processed independently, the system currently does not leverage global structural priors such as repeated façades or urban symmetries. Overall, these limitations are localized and do not affect the method’s strong scalability, stability, and suitability for city-scale digital-twin applications.