

Lineage-Guided Gaussian Filtering for Building-Centric 3D Reconstruction

Matthew Zhang
University of California, Santa Barbara
linghe@ucsb.edu

Kazu Pang
University of California, Santa Barbara
kazu@ucsb.edu

June 15, 2026

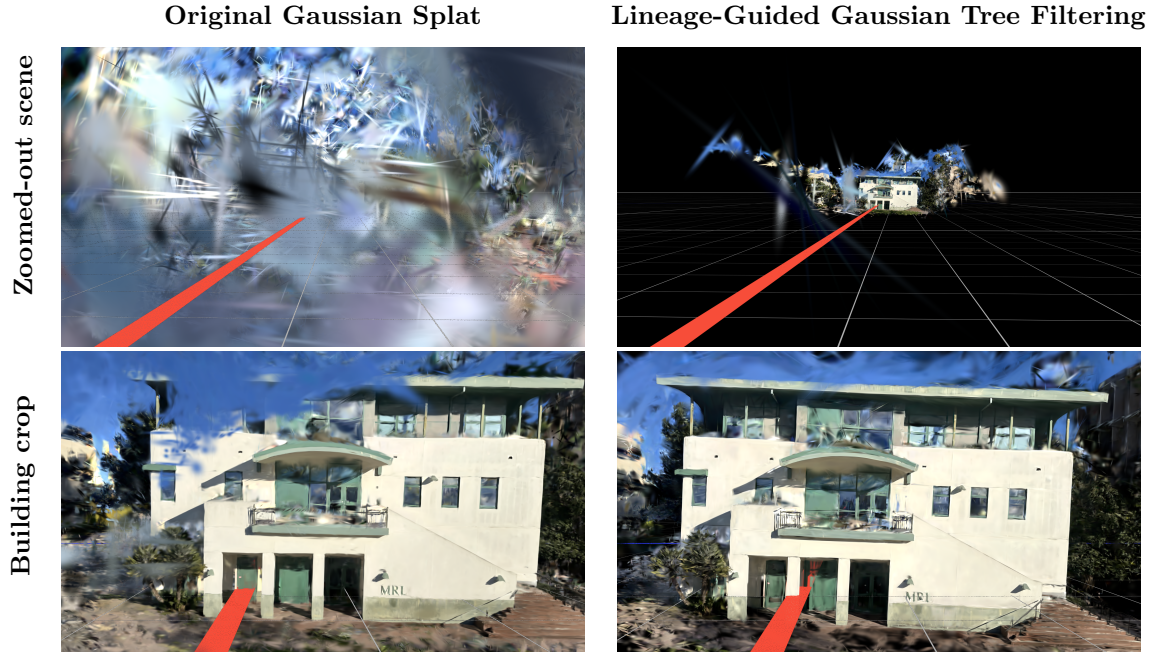


Figure 1: Comparison between the original Gaussian splat reconstruction and our lineage-guided Gaussian tree filtering. The zoomed-out view shows that the original reconstruction is surrounded by a large cloud of noisy Gaussians and floaters, while our method removes most unstable descendant branches. The building crop shows that the main structure is preserved after filtering.

Abstract

Gaussian splatting has become an effective representation for real-time 3D scene reconstruction, but reconstructed scenes often contain floating geometry and noisy Gaussian clusters that extend far beyond the primary structure of interest. These artifacts arise because the optimization is constrained primarily by the captured camera views and is not explicitly required to produce geometrically consistent structure in poorly observed regions. Consequently, the artifacts may remain hidden when the scene is viewed

from positions near or within the original camera trajectory. However, as the viewer moves outside the camera capture radius, these unconstrained Gaussians become visible as floating or stretched geometry, as shown in Fig. 1.

To address this issue while preserving the quality of the target building, we store lineage information during Gaussian splatting by tracing newly grown Gaussians back to their original MAST3R-COLMAP¹ root

¹We use MAST3R-COLMAP to refer to sparse root points initialized from MAST3R feature matches and then refined through COLMAP structure-from-motion and bundle adjust-

points in the sparse cloud. We first show that individual lineage attributes, including lineage depth and distance from the original root, are strong indicators of surrounding noise, but each removes artifacts in different ways. We further observe that when a Gaussian is identified as unstable by these lineage-based signals, its descendants are often unstable as well. Based on this observation, we introduce a lineage-guided Gaussian filtering method that propagates confidence from root Gaussians through the lineage tree. Rather than filtering Gaussians independently, each Gaussian’s confidence depends on its parent confidence, its distance from the original root, and its lineage depth. When a Gaussian’s confidence falls below a threshold, the method prunes that Gaussian along with its descendant branch. This produces cleaner reconstructions in which the target building is visible while preserving most of its structure.

1 Introduction

3D Gaussian Splatting provides a high-fidelity and efficient representation for reconstructing buildings and other 3D structures using optimized anisotropic Gaussian primitives [5]. However, practical reconstructions often contain artifacts that appear acceptable near the training-view camera poses, but become problematic when the scene is inspected from viewpoints farther away. As shown in Figure 1, zooming out reveals that the target structure is enveloped by a surrounding noise cloud.

These artifacts include floating and unstable Gaussians that arise from geometrically inconsistent image regions, such as sky, moving objects like vegetation or people, and peripheral scene content outside the central building view. As a result, some Gaussians explain image observations without corresponding to stable 3D structure, forming noisy clusters that envelop the building.

These surrounding artifacts are not the primary target of many existing 3D Gaussian reduction methods, which focus on reducing memory, improving rendering speed, or removing globally low-importance Gaussians [2, 4, 1, 10, 3]. While these goals are useful, ment to produce the final sparse point cloud [7, 11].

they do not directly address building-centric scene cleanup. A Gaussian may be useful for minimizing training-view photometric error while still being undesirable when the goal is to isolate a target building from surrounding clutter. Therefore, building-centric cleanup requires a different signal.

Recent semantic cleanup methods move closer to this goal by using segmentation masks to identify the target object. For example, Clean-GS uses sparse semantic masks to remove background clutter and floaters from 3D Gaussian reconstructions [9]. However, mask-based cleanup depends heavily on segmentation quality: if a mask cuts off parts of the building, misses thin structures, or fails when the building is not centered clearly, valid building Gaussians may be removed along with noise.

Our method instead uses a signal from deeper in the reconstruction stack. 3D Gaussian Splatting is commonly initialized from sparse structure-from-motion points, and prior anchor-based methods have shown that organizing Gaussians around SfM-derived points can improve stability and reduce redundancy [5, 8]. Motivated by this, we trace each Gaussian back to its original MAST3R-COLMAP root point, which originates from MAST3R feature matches and is refined through COLMAP structure-from-motion and bundle adjustment [7, 11]. These roots provide a reconstruction-aware signal because they correspond to points matched across views and optimized for geometric consistency.

During training, we store Gaussian lineage information as new Gaussians are created from existing ones. After training, we use this lineage to analyze each Gaussian’s distance from its original root and lineage depth. We observe that Gaussians with large distance from their original root or high lineage depth often correspond to surrounding floaters and noise around the building. Moreover, when a Gaussian is identified as unstable based on these two lineage attributes, its descendants are often unstable as well. Based on this observation, we perform tree-based filtering using these two lineage attributes, rather than filtering each Gaussian independently with a single threshold. This greatly reduces the surrounding artifacts in our building-centered scenes while preserving most of the target structure.

Because our pruning signal is rooted in the sparse reconstruction, the method depends on the balance of points in the initial sparse point cloud. If nearby buildings, trees, or other non-target objects contain only a small number of stable roots, lineage filtering can still suppress many of their descendants. However, if those non-target structures contain enough stable roots to rival or overpower the target building, their descendants may also appear geometrically valid and survive pruning. In these cases, the sparse point cloud may need to be cleaned before training so that the root scaffold is biased toward the intended building.

Our contributions are:

- We analyze the relationship between Gaussian lineage attributes, including root distance and lineage depth, and the appearance of surrounding artifacts in building-centric reconstructions.
- We introduce a lineage-based confidence propagation method for Gaussian filtering, together with a web application for interactively adjusting cleanup parameters.
- We show through a case study that our method can reduce floaters and surrounding noise while preserving much of the target building structure, making multi-building visualization cleaner.

2 Related Work

Gaussian Splatting. 3D Gaussian Splatting represents scenes with anisotropic Gaussian primitives optimized through differentiable rendering [5]. Although 3DGS uses adaptive density control, including cloning, splitting, opacity reset, and low-opacity Gaussian removal, these steps are meant to improve training and rendering rather than isolate a target object [5]. As a result, Gaussians may still appear in weakly constrained regions such as sky boundaries, vegetation, reflective surfaces, or other surrounding clutter.

Gaussian Reduction and Compression. Gaussian reduction and compression methods typically

aim to reduce the number of Gaussians while maintaining rendering quality. These methods rank or remove Gaussians using signals such as opacity, image-space contribution, learned importance, redundancy, or sensitivity to reconstruction loss [2, 4, 1, 10, 3]. Such methods are useful when the goal is compactness or faster rendering, but they are not always aligned with building-centric cleanup. In contrast, our method does not primarily ask whether a Gaussian is globally important; it asks whether the Gaussian belongs to a lineage that remains consistent with the target building. Our goal is to use a reconstruction-aware signal to isolate the structure the user intended to capture by removing surrounding floaters and clutter. Existing compression methods could then be applied afterward to further reduce storage or rendering cost.

Segmentation-Guided Gaussian Cleanup.

Segmentation-guided methods transfer object-level information into 3D Gaussian reconstructions by associating 2D masks with Gaussians for selection, editing, or removal [?, 9]. These methods provide useful semantic control, but their results depend on mask quality and cross-view consistency. If the masks miss parts of the building, include nearby clutter, or vary across views, valid building structure may be removed while unwanted artifacts remain. Our method instead uses reconstruction lineage as a complementary signal, relating each Gaussian to geometrically consistent MAST3R-COLMAP roots rather than relying only on external semantic masks.

Geometry-Guided Reconstruction. Sparse geometry, feature correspondences, and multi-view consistency provide important cues for 3D reconstruction. Structure-from-motion pipelines recover sparse points by matching features across views and refining camera poses and point locations through bundle adjustment [11]. MAST3R similarly provides dense image matching and 3D-aware correspondences that can be used to initialize or support downstream reconstruction [7]. Our method uses this geometric signal indirectly: we trace each Gaussian back to its MAST3R-COLMAP root and use lineage attributes

to identify descendant branches that are less consistent with the target building.

Anchor-Based Gaussian Representations. Scaffold-GS introduces a structured Gaussian representation in which SfM-derived points are converted into voxel anchors, and local neural Gaussians are generated from visible anchors during rendering [8]. This anchor-based design improves compactness, convergence, and view-adaptive rendering, and can reduce artifacts caused by redundant Gaussians. Our work is related in that it also uses sparse reconstruction points as a stabilizing signal. However, rather than using anchors to generate Gaussians during training and rendering, we use Gaussian lineage from MAST3R-COLMAP roots as a post-training cleanup signal for building-centric artifact removal.

3 Method

3.1 MAST3R Features and Bundle Adjustment

Our filtering method relies on a sparse geometric scaffold that is stable enough to serve as the root structure for later Gaussian lineage analysis. To obtain this scaffold, we use MAST3R-derived feature correspondences as input to the structure-from-motion and bundle adjustment pipeline. MAST3R formulates image matching as a 3D-grounded correspondence problem and produces dense local feature maps and robust matches across views, making it useful for scenes with wide viewpoint changes and difficult image overlap [7].

This sparse reconstruction is important because our later filtering signal depends on the relationship between optimized Gaussians and their inherited reconstruction roots. Regions such as sky, moving objects, and weakly textured background clutter are less likely to produce stable multi-view geometric correspondences. In contrast, building facades, edges, windows, and other persistent structures are more likely to generate repeatable matches across images. As a result, the sparse point cloud provides a reconstruction-aware prior that is biased toward

scene regions that are geometrically consistent across views.

Given these correspondences, bundle adjustment refines camera parameters and 3D point locations by minimizing multi-view reprojection error. Rather than directly guaranteeing more points, reliable and sufficiently dense matches provide more candidate constraints for optimization. After geometric verification and filtering, this can produce a more stable sparse point cloud with more usable anchors for our lineage-based filtering. This matters because if the sparse reconstruction contains too few stable building points, then many later Gaussians have weak or missing geometric roots, making lineage confidence less informative.

This also introduces a limitation: the method depends on the sparse point cloud being biased toward the target building. Our dataset naturally has this bias because the cameras are positioned around a single building, so the dominant stable correspondences usually come from the building facade, roofline, windows, and other persistent architectural features. If nearby buildings, trees, or occluding objects contain enough stable sparse points to rival the target structure, their descendant Gaussians may also appear valid and survive filtering. In these cases, the sparse point cloud may need to be cleaned before training so that the root scaffold better represents the intended building.

3.2 Lineage Representation

During Gaussian optimization, we augment each Gaussian with four lineage attributes: a unique identifier, a parent identifier, a root identifier, and a depth value. The unique identifier tracks the Gaussian across optimization. The parent identifier records the Gaussian from which the current Gaussian was created or inherited. The root identifier points to the initial Gaussian associated with the original sparse reconstruction anchor, and the depth value records the number of inheritance steps from that root.

This representation defines a directed forest over the Gaussian set rather than a single global tree. Each tree begins from a root Gaussian initialized from the sparse point cloud used by 3D Gaussian

Splatting [6]. Descendant Gaussians are then created during densification, cloning, splitting, and parameter update steps. In our implementation based on gsplat [13], split densification creates two child Gaussians from one parent, so the split operation naturally forms a binary branching structure. Clone and inherited update operations preserve the same lineage relation while increasing or maintaining depth depending on the operation.

Even when an intermediate parent Gaussian is later removed, the descendant still stores its inherited depth and original root identifier. This is important because the lineage structure records the historical origin of a Gaussian, not only the currently surviving set of Gaussians. Therefore, the representation allows us to analyze each Gaussian by its original sparse reconstruction root and its accumulated lineage depth. These attributes make it possible to study whether unstable floaters and surrounding clutter are correlated with root displacement or excessive depth growth.

3.3 Analysis of Lineage Attributes

This lineage representation allows us to analyze Gaussians using several inheritance-based signals. In practice, we focus on the two attributes that were most effective for removing noise: lineage depth and root distance. Lineage depth measures how many inheritance steps separate a Gaussian from its original sparse reconstruction anchor. Root distance measures how far a Gaussian drifts from its original root anchor.

Lineage also lets us analyze entire descendant branches rather than only individual Gaussians. In a further experiment, when a Gaussian fails a condition, we render the failed Gaussian together with its descendants. We find that descendants of unstable ancestors often form the surrounding cloud artifacts, even when some of them individually satisfy simple distance or depth thresholds. This motivates confidence propagation: once a Gaussian becomes unreliable, its descendants should inherit reduced confidence instead of being judged independently.

3.4 Confidence Propagation

Let g_i be a Gaussian with center μ_i . Each Gaussian stores a parent Gaussian g_p , a root Gaussian g_r , and a lineage depth d_i . If g_i has no valid parent, it is treated as a start node. True roots, which originate from the initial sparse reconstruction, begin with full confidence. In contrast, orphan branches or Gaussians without a valid surviving parent begin with reduced confidence because their geometric origin is less reliable.

We assign true root Gaussians an initial confidence of 1.0 and propagate confidence downward through the lineage forest. For a child Gaussian g_i with parent g_p and root anchor g_r , the propagated confidence is

$$C_i = C_p S_r S_d, \quad (1)$$

$$S_r = \exp\left(-\frac{\|\mu_i - \mu_r\|_2}{\sigma_r}\right), \quad (2)$$

$$S_d = \exp(-\lambda_d \Delta d). \quad (3)$$

where C_p is the parent confidence, σ_r controls sensitivity to root displacement, and λ_d controls the lineage depth penalty. The depth increment is defined as

$$\Delta d = \max(1, d_i - d_p). \quad (4)$$

The root-distance term penalizes descendants that drift far away from the original sparse reconstruction anchor. The depth term acts as a branch-length prior: deeper descendants are trusted less unless they remain spatially consistent with their root.

The method also includes several user-controllable filtering parameters. The root scale σ_r controls how far descendants may drift from the original sparse reconstruction root. A smaller value keeps the reconstruction tightly attached to the initial point cloud, while a larger value allows broader growth around the scene. The depth decay λ_d controls how strongly confidence decreases with lineage depth. Increasing this value removes deeper branches more aggressively.

After confidence is propagated, a minimum confidence threshold τ determines which Gaussians are kept:

$$\text{keep}(g_i) = \begin{cases} 1, & C_i \geq \tau, \\ 0, & C_i < \tau. \end{cases} \quad (5)$$

The threshold τ controls the final filtering strength. Higher values produce cleaner but more aggressive filtering, while lower values preserve more Gaussians at the cost of retaining additional clutter.

3.5 Handling Orphan Start Nodes

Not every start node is a true root from the original sparse reconstruction. During Gaussian optimization, some parent Gaussians may be removed by the underlying 3DGS optimization process, for example because their opacity becomes too low or they no longer contribute meaningfully to the rendered views [6, 13]. When this happens, a surviving descendant can lose its valid parent even though it may still originate from a previously stable region.

We therefore do not treat these parent-missing branches as full-confidence roots. Instead, we assign them a reduced orphan confidence prior:

$$C_i = C_{\text{orphan}} \exp(-\lambda_{\text{orphan}} d_i), \quad (6)$$

where C_{orphan} is the base confidence for orphan branches and λ_{orphan} is the orphan depth decay. This gives parent-missing branches a chance to survive if they remain geometrically consistent, while still penalizing them for having a broken lineage path. If an orphan branch begins below the minimum confidence threshold, the entire branch is removed. Similarly, if a parent is removed due to low confidence during our propagation step, its descendants are marked as ancestor-pruned unless they are reached through another valid kept branch.

Together, these parameters allow the filtering process to be adjusted depending on the scene. Root scale controls global drift from the sparse reconstruction, depth decay controls branch growth, minimum confidence controls overall filtering strength, and orphan confidence controls how much trust is assigned to disconnected lineage branches.

4 Experiments

4.1 Experimental Setup

We evaluate our method on building-centric Gaussian splat reconstructions captured with iPhone 14 and iPhone 15 cameras. The scenes are centered around UCSB buildings. For each scene, the photographer walks in a rough circle around the target building and captures images while facing the building. Each reconstruction contains a dominant architectural subject, but also includes surrounding sky, vegetation, boundary artifacts, and occasionally overlapping nearby buildings.

Because the full dataset contains multiple building reconstructions, we show representative examples in the main paper and use them to illustrate the dominant filtering behaviors. The same filtering procedure is applied to all scenes, but not every building is shown due to space constraints.

4.2 Qualitative Results

A direct reconstruction loss or image-space error is not always aligned with our goal. Standard reconstruction metrics evaluate whether rendered training views remain close to the input images. Our objective is different: we want to remove surrounding clutter and floating Gaussians while preserving the target building. A Gaussian may have little effect on image-space error while still creating unwanted geometry around the object of interest, especially in sky, vegetation, or background regions.

Therefore, we perform incremental observational filtering studies and compare the resulting renders. For depth, depth 0 keeps only the surviving root Gaussians initialized from the sparse reconstruction. Depth 1 additionally keeps Gaussians produced after one split, clone, or inheritance step. Larger depth thresholds reveal how deeper descendants contribute to the final rendered structure. We perform similar threshold sweeps for root distance. We then inspect whether each threshold removes floaters, preserves facade structure, and avoids cutting away valid building geometry.

4.3 Lineage Parameter Study

Among the tracked lineage attributes, we find that two parameters most directly affect cleanup quality: lineage depth and root distance. The effects of these parameters on the building reconstruction are shown in Figures 2 and 3.

Lineage depth controls how far descendants are allowed to grow from the original sparse reconstruction. Smaller depth thresholds keep only early, stable Gaussians, while larger thresholds preserve more descendants but may also retain floating artifacts. Root distance measures global drift by checking how far a Gaussian moves from its original sparse reconstruction root. Together, these two parameters capture branch growth and long-range drift in the lineage tree.

To compare these parameters fairly, we evaluate each one at matched filtering levels. For each remaining lineage attribute, we choose thresholds that approximately match the filtering strengths from the depth sweep in Table 1. This lets us compare what each parameter removes at similar remaining Gaussian counts, rather than comparing different filtering aggressiveness levels.

At matched filtering levels, the two attributes show different visual tradeoffs. Depth controls how much descendant structure remains, and root distance captures global drift from the original reconstruction anchor. This suggests that the two signals are complementary, motivating our final filtering rule that combines them.

Depth. We sweep depth thresholds from 0 to 4 because this range captures the main tradeoff between preserving building detail and suppressing artifacts. As shown in Table 1 and Figure 2, depth 0 keeps only surviving root Gaussians, producing a stable but sparse reconstruction that misses much of the building geometry. Increasing the threshold preserves more descendants and recovers finer structure.

However, higher depths also reintroduce unstable branches. Depth 4 retains more visible detail, but begins to recover artifacts from nearby buildings, occluding objects, and surrounding clutter. Depth 3 provides a better balance between building coverage

Table 1: Depth threshold study. All other filtering parameters are fixed.

Max depth d	Kept splats	Removed (%)
0	24,191	95.09
1	72,884	85.20
2	115,166	76.61
3	154,316	68.66
4	191,686	61.07

Table 2: Root distance threshold study. All other filtering parameters are fixed.

Max root distance	Kept splats	Removed (%)
0	26,342	94.65
0.0035	71,470	85.49
0.0057	115,993	76.46
0.00755	155,004	68.54
0.0095	192,719	60.88

and artifact suppression, although it still does not remove all clutter on its own. This motivates combining depth with root distance, so that deeper Gaussians are only retained when they remain geometrically consistent with their original root.

Root Distance. We sweep root distance thresholds from 0 to 0.0095 and choose these values so that the number of kept Gaussians approximately matches the filtering levels from the depth study in Table 1. This allows us to compare root distance and depth at similar pruning strengths rather than comparing results with very different remaining Gaussian counts. As shown in Table 2 and Figure 3, a root distance threshold of 0 keeps only Gaussians that remain exactly at their original sparse reconstruction root locations, producing a very sparse but stable reconstruction. Increasing the threshold allows descendants to drift farther from their original root and recovers more of the building structure.

However, root-distance pruning removes a different set of unstable branches than depth pruning. While depth limits how far a Gaussian can grow through the lineage tree, root distance limits how far the Gaus-

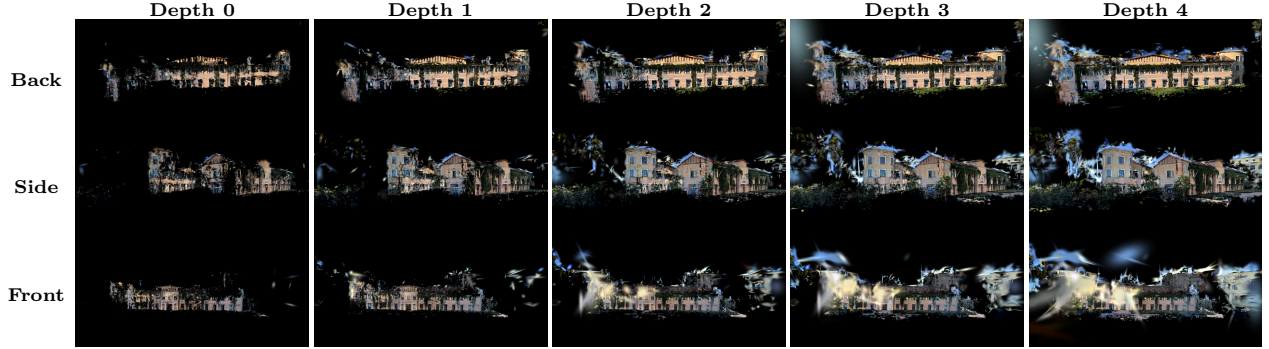


Figure 2: Depth-based visualization of Gaussian lineage filtering across three viewpoints. Columns show increasing lineage depth, while rows show back, side, and front renderings.

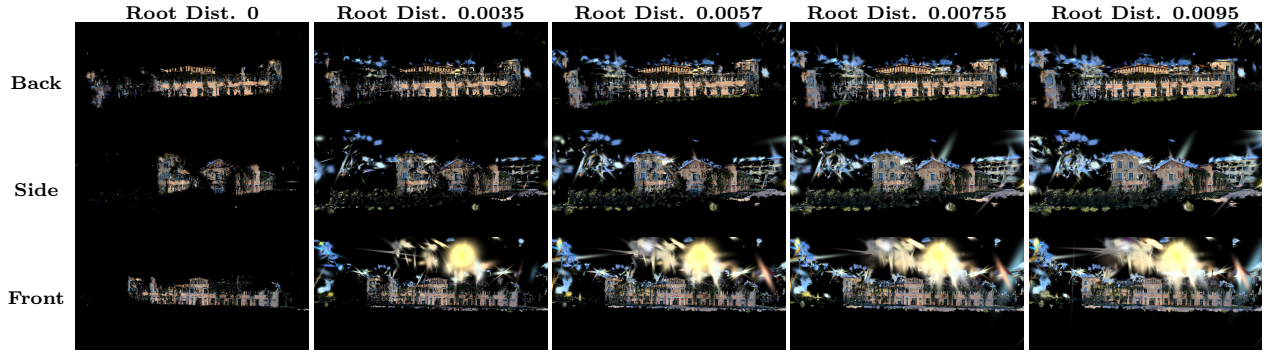


Figure 3: Root-distance-based visualization of Gaussian lineage filtering across three viewpoints. Columns show increasing maximum distance from the original sparse reconstruction root, while rows show back, side, and front renderings.

sian can spatially drift from its original MAST3R-COLMAP root. As a result, some Gaussians that survive a depth threshold may still be removed by root distance because they have moved too far from the original reconstruction scaffold. Conversely, some deeper descendants may survive root-distance pruning if they remain close to their original root. Comparing Figure 2 and Figure 3 shows that the two filters suppress different surrounding artifacts, which motivates treating depth and root distance as complementary lineage signals rather than interchangeable thresholds.

4.4 Complementary Lineage Signals

As shown in Figures 2 and 3, depth and root-distance filtering preserve different parts of the building even when they retain approximately the same number of Gaussians. Depth measures how far a Gaussian has propagated through the lineage tree, whereas root distance measures how far it has spatially moved from its original reconstruction root. Because the two signals classify different Gaussians as unstable and recover different structural regions, they should be treated as complementary rather than interchangeable hard thresholds.

4.5 Lineage Confidence Propagation

We therefore assign each Gaussian a confidence value that is propagated from its parent through the lineage tree. Confidence decreases as a Gaussian moves farther from its parent or original root and as its lineage depth increases. This progressively penalizes descendants that drift away from the geometrically supported sparse reconstruction while preserving nearby descendants that continue to represent the main structure.

4.6 Interactive Tree-Based Tuning

Because different signals remove and preserve different Gaussian branches, pruning can be viewed as a decision process over the lineage tree. The user can adjust the contributions of depth and root distance, along with the final confidence threshold, to control the cleanup strength. This provides a more flexible tradeoff between artifact removal and structural preservation than using a single fixed filtering rule. Our interactive tuning tool is available at github.com/Kazu-Pang/splat-param-viewer.

5 Case Study

To test whether the cleaned Gaussian models are useful beyond isolated renderings, we built an interactive UCSB campus viewer containing 20 reconstructed campus buildings. Each building was pruned using our lineage-based filtering method and then rendered in a web application using Spark, a Three.js-based 3D Gaussian Splatting renderer for the browser [12]. This setup lets users inspect multiple buildings directly in the browser rather than viewing each reconstruction as a separate static result.

The case study shows the practical benefit of removing surrounding artifacts before deployment. In the unpruned reconstructions, nearby buildings, floaters, and spherical clutter often occlude the target structure or make the scene difficult to navigate. After lineage-based pruning, the buildings appear more isolated, the view is less blocked by surrounding artifacts, and the Gaussian count can be

further reduced without destroying the main structure. The web viewer is available at: <https://frontend-showcase-seven.vercel.app/>.

6 Conclusion

We presented lineage-guided Gaussian filtering for object-centric building reconstruction. By propagating confidence through parent-child Gaussian lineage trees, our method removes descendants that drift from their root anchor or grow too deep in the lineage tree. This enables hierarchical removal of unstable branches such as floaters, sky fringes, and peripheral artifacts. The method is especially suited for cases where the goal is not only compact rendering, but clean isolation of the main building structure.

References

- [1] Muhammad Salman Ali, Luis Roldao, Joao Sequeira, Adrien Gaidon, Marc Habermann, and Justus Thies. Trimming the fat: Efficient compression of 3d gaussian splats through pruning. *arXiv preprint arXiv:2406.18214*, 2024.
- [2] Zeyu Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. In *Advances in Neural Information Processing Systems*, 2024.
- [3] Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. *arXiv preprint arXiv:2412.00578*, 2024.
- [4] Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. *arXiv preprint arXiv:2406.10219*, 2024.
- [5] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):1–14, 2023.
- [6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian

- splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023.
- [7] Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision (ECCV)*, 2024.
 - [8] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
 - [9] Subhankar Mishra. Clean-gs: Semantic mask-guided pruning for 3d gaussian splatting. *arXiv preprint arXiv:2601.00913*, 2026.
 - [10] Panagiotis Papantonakis, Georgios Kopanas, Bernhard Kerbl, Alexandre Lanvin, and George Dretakis. Reducing the memory footprint of 3d gaussian splatting. *arXiv preprint arXiv:2406.17074*, 2024.
 - [11] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.
 - [12] World Labs. Spark: An advanced 3d gaussian splatting renderer for @miscspark, title = Spark: An advanced 3D Gaussian Splatting renderer for Three.js, author = World Labs, year = 2026, howpublished = <https://github.com/sparkjsdev/spark>, note = Accessed 2026 three.js. <https://github.com/sparkjsdev/spark>, 2026. Accessed 2026.
 - [13] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *Journal of Machine Learning Research*, 26(34):1–17, 2025.