CONTROLGS: FACTORIZED CONTROL FOR CONTINUAL 3D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual 3D Gaussian splatting reconstruction updates a model from new images while preserving unchanged content and earlier states. Existing methods apply a uniform end-to-end procedure across different types of scene evolution, without adequately accounting for their distinct effects on scene content and Gaussian model capacity. Such a rigid paradigm is prone to modifying irrelevant attributes, causing unexpected structural changes, and gradually undermining historical recovery as the Gaussian representation evolves. In this paper, we propose ControlGS, which adaptively updates Gaussian primitives for each scene evolution type by decomposing 3DGS updating into attribute update control, structure update control, and historical replay control. In the attribute update control stage, observed scene variations are combined with visibility, depth, and camera alignment to estimate the updating type and reliability. Each variation and its uncertainty then guide the optimization of geometric or appearance attributes in terms of both scope and strength, while unrelated attributes remain fixed. In the structure update control stage, ControlGS restricts updates to regions where the new observations deviate sufficiently from the original model. Whether a capacity optimization is needed is then assessed based on observation evidence and model complexity.In the historical replay control stage, ControlGS recovers version-annotated Gaussian parameters and topology instead of replaying past images or relying on mutable tensor indices. Persistent identifiers link Gaussians across structural changes, while sparse pre-update records and occasional full snapshots support reverse reconstruction after row reordering. Extensive experiments demonstrate that ControlGS outperforms state-of-the-art methods in reconstruction quality while maintaining real-time rendering. Additional tests verify that irrelevant parameters remain fixed and earlier states are reliably recovered.

Keywords. Continual 3D reconstruction, 3D Gaussian splatting, scene change, historical recovery

1 INTRODUCTION

A scene model used in robotics, mixed reality, or digital-twin maintenance must remain useful after the scene changes. New images may show moved furniture, altered appearance, added objects, or missing objects. A model reconstructed from scratch discards previous geometry; optimization solely against the latest capture can instead modify unchanged regions when residuals arise from illumination, occlusion, or camera error. NeRF (Mildenhall et al., 2020) and 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) provide strong novel-view rendering, but continual updates require an explicit rule for preserving old information.

Continual methods preserve old views through replay or time conditioning (Cai & Müller, 2023), restrict Gaussian optimization to changed regions (Ackermann et al., 2025), or stage appearance, removal, and addition with visibility-aware replay (Zeng et al., 2025). These advances leave three coupled decisions. First, a color residual may justify changing spherical-harmonic coefficients without moving geometry. Second, a new surface may require more primitives, whereas a lighting residual should not trigger a split. Third, Gaussian row indices cease to identify the same primitive after pruning, insertion, or reordering. A continual update needs to specify parameter permissions, capacity transitions, and historical identity separately.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We formulate continual 3DGS as three coordinated controls. Attribute control decides which existing parameters can change and by how much. Structure control decides where the representation may gain or lose Gaussians. History control records each accepted transition so that committed versions remain recoverable. This division yields testable contracts: blocked parameters do not drift, unsupported topology moves are rejected, and recovery is keyed by persistent identity rather than tensor position.

ControlGS obtains change evidence from previous-state renders, multi-view feature and depth cues, visibility, and camera–object registration. Object-state reasoning converts those cues into event labels and uncertainty estimates. The attribute controller compiles an event into masks and continuous budgets; the structure controller combines event permission with rendering support; the history controller captures pre-states and lineage around the accepted changes. Figure 1 summarizes this sequence.

The supplied ten-scene WAT summary reports 27.49 dB mean PSNR, 0.8882 SSIM, and 0.1641 LPIPS for the full-update ControlGS path. The comparisons include CLNeRF (Cai & Müller, 2023) and CL-Splats (Ackermann et al., 2025) reproductions, published GaussianUpdate PSNR and SSIM without LPIPS (Zeng et al., 2025), and the only-new-data baseline. ControlGS leads on nine, seven, and ten scenes in PSNR, SSIM, and LPIPS among available values. The reported ControlGS rows lead their M1 and M2 groups against full fine-tuning, native 3DGS growth (Kerbl et al., 2023), CL-Splats-style and GaussianUpdate-style rows, and an M2 row based on 3DGS-MCMC (Kheradmand et al., 2024) (Table 3). Table 4 reports the smallest mean earlier-scene image drop for M3. Focused checks exercise M1 attribute freezing, M2 split rejection, and M3 state recovery in constructed cases. These reported outcomes favor all three controls, but different update paths and missing run logs prevent attributing image gains to individual controls or verifying recovery over longer sequences.

The main contributions are

- We propose ControlGS, which adaptively updates Gaussian primitives for each scene evolution type by decomposing 3DGS updating into attribute update control, structure update control, and historical replay control.
- We introduce an attribute update control scheme that uses observed variations with visibility, depth, and camera alignment to estimate updating type and reliability, guiding geometric or appearance optimization while keeping unrelated attributes fixed.
- We develop a structure update control scheme that restricts updates to regions where new observations deviate sufficiently from the original model and assesses capacity optimization based on observation evidence and model complexity.
- We present a historical replay control scheme that recovers version-annotated Gaussian parameters and topology via persistent identifiers, sparse pre-update records, and occasional full snapshots for reverse reconstruction after row reordering.

2 Related Work

2.1 Continual Gaussian Reconstruction

CLNeRF combines Instant-NGP (Müller et al., 2022) with generated past views and time embeddings, and introduces WAT (Cai & Müller, 2023). Generated replay has roots in continual learning (Shin et al., 2017). CL-NeRF uses expert modules (Wu et al., 2023). Other continual neural-field methods include MEIL-NeRF (Chung et al., 2025), Instant Continual NeRF (Po et al., 2023), and IL-NeRF (Zhang et al., 2025). These methods use earlier predictions, replay, or camera alignment. They do not record individual Gaussian changes. CL-Splats saves local Gaussian history (Ackermann et al., 2025). GaussianUpdate separates appearance, removal, and addition (Zeng et al., 2025). 3DGS-MCMC changes the Gaussian count through relocation (Kheradmand et al., 2024).

3DGS-CD finds object changes from images with unmatched camera poses (Lu et al., 2024). GaME updates a current Gaussian map (Yugay et al., 2026). SplaTAM (Keetha et al., 2024), GS-SLAM (Yan et al., 2024), and Gaussian Splatting SLAM (Matsuki et al., 2024) build maps while tracking cameras. Gaussian Grouping (Ye et al., 2024), LangSplat (Qin et al., 2024), and GaussianEditor (Chen et al., 2024) support object selection or editing. DINOv2 (Oquab et al., 2023), Segment Anything (Kirillov et al., 2023), and Depth Anything V2 (Yang et al., 2024a) provide image, object, and depth cues. ControlGS uses such cues to control updates and save earlier states.

Dynamic Gaussian methods track motion in video. They include Dynamic 3D Gaussians (Luiten et al., 2024), Deformable 3D Gaussians (Yang et al., 2024b), 4D Gaussian Splatting (Wu et al., 2024), Spacetime Gaussian Feature Splatting (Li et al., 2024), and SC-GS (Huang et al., 2024). Our scans arrive at separate times, and objects may be replaced. M3 uses stable Gaussian IDs to recover earlier states after Gaussian rows move. IDs do not solve uncertain matches between similar objects.

3 CONTROLGS

3.1 PROBLEM FORMULATION

At scan t, the inputs are the earlier Gaussian state S_{t-1} and new images with camera poses $\mathcal{D}_t = \{(I_t^v, C_t^v)\}_v$. The state is $S_t = (\Theta_t, L_t, T_t, U_t)$. Here Θ_t contains Gaussian attributes, L_t records when each Gaussian is active, T_t records parent–child links, and U_t maps Gaussian rows to stable IDs. The outputs are the updated state $S_t = F(S_{t-1}, \mathcal{D}_t)$ and a history record for requests $H(\tau) = S_\tau$ with $\tau \leq t$. We seek a good fit to the new images while keeping unchanged content and earlier states.

3.2 EVIDENCE FOR SCENE CHANGES

The local workflow in Figure 1 gives M1 and M2 the same change evidence. We render S_{t-1} into color, depth, visibility, and Gaussian contribution maps. We compare these with new images after camera and object alignment. Areas with little earlier coverage are checked for new objects. Depth and agreement across views place candidate changes in 3D. Alignment helps reject camera errors. The output is a change type, its location, a score c_e, and uncertainty values. Types include appearance, movement, addition, removal, occlusion, and camera error. The scores are not calibrated probabilities. M1 selects attributes using this evidence. M2 uses the same locations to assess changes to the Gaussian set, so a color difference alone does not trigger a geometry edit.

We compare image features only where the earlier model has visible support. Regions without such support are checked separately for additions. A candidate removal must remain visible from more than one new view before the system treats a missing object as a real change. If camera alignment explains an image difference, the system delays the edit. These checks matter because a shadow, a hidden object, and an incorrect camera pose can otherwise produce similar image errors. When the evidence does not distinguish them, the update is delayed or limited to a small appearance change.

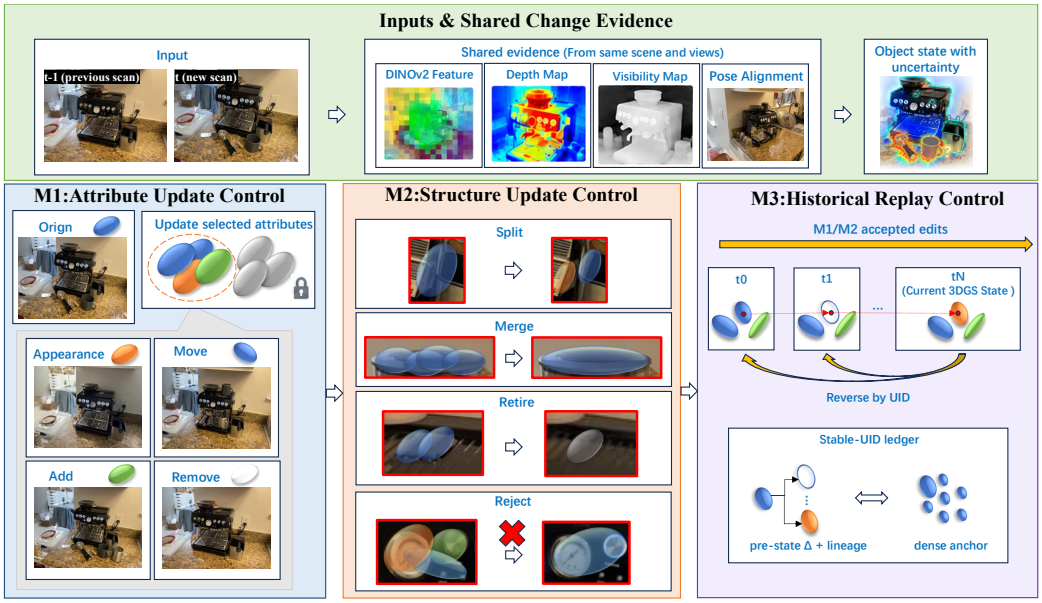

Figure 1. ControlGS local workflow. Image, depth, visibility, and camera cues guide M1 attribute updates and M2 changes to the Gaussian set. M3 saves earlier values and parent–child links by stable Gaussian ID. The ten-scene full-update path updates all attributes and uses standard 3DGS growth.

3.3 M1 Attribute Update Control

M1 has three steps. It selects the Gaussian attributes allowed to change, limits each update according to uncertainty, and prevents the optimizer from changing blocked attributes. Together, these steps keep blocked values fixed.

3.3.1 Selecting Attributes for Update

For each Gaussian i and attribute $a \in \{\text{position}, \text{scale}, \text{rotation}, \text{opacity}, \text{SH}\}$, M1 sets a permission m_i^a to zero or one and an update weight $b_i^a \in [0, 1]$. SH denotes the color coefficients. Table 1 shows the local rules. An appearance change mainly allows color to change. Movement can allow position and rotation to change after object alignment. A new Gaussian changes color and opacity before its geometry is released. M2 handles removal. Occlusion and camera errors block updates.

Table 1. Local M1 rules before uncertainty is applied. Blocked Gaussian attributes and optimizer state stay fixed. M2 handles removal.

Inferred change	Geometry	Opacity / color	Gaussian set
Unchanged	Freeze	Freeze	None
Appearance change	Freeze	Low budget / update	None
Movement	Update	Update	Local split possible
Addition	Delayed release	Update	Birth possible
Removal / replacement	No gradient edit	No gradient edit	Retire old lifetime
Occlusion / pose error	Freeze	Freeze	Defer
Uncertain	Freeze	At most reduced SH	Defer

For a moved object, an estimated object transform sets the initial position and rotation. Scale may also change if allowed. A new Gaussian starts with appearance updates. When the evidence is uncertain, only a small color update may be allowed.

3.3.2 Limiting the Size of Each Update

The permission says whether an attribute may change. The update weight sets how much it can change. Geometry uses the change score c_e and uncertainty about geometry, predictions, visibility, and camera pose.

$$b_{\text{geo}} = c_e[1 - \max(u_{\text{geo}}, u_{\text{epis}}, u_{\text{vis}}, u_{\text{pose}})], \qquad \nabla\theta_i^a \leftarrow m_i^a b_i^a \nabla\theta_i^a. \tag{1}$$

Opacity and color use only their relevant uncertainty values.

$$\begin{aligned} b_{\text{opa}} &= c_e[1 - \max(u_{\text{app}}, u_{\text{vis}}, 0.5u_{\text{pose}})], \\ b_{\text{SH}} &= c_e[1 - \max(u_{\text{app}}, 0.5u_{\text{epis}})]. \end{aligned} \tag{2}$$

Each update weight is limited to $[0, 1]$. A separate rule can set it to zero. For example, poor visibility blocks an opacity update even when the image color changes.

3.3.3 Keeping Blocked Values Fixed

A zero gradient does not always keep a Gaussian row fixed under Adam (Kingma & Ba, 2015). M1 also clears the optimizer's stored values for blocked rows and restores the blocked Gaussian attributes after every step. In the supplied checks, these values remain exactly unchanged. M1 updates its permissions when the Gaussian set changes.

Before optimization, M3 saves the earlier values of any Gaussian that M1 may change. A saved value can then be restored even if the local update later changes its row position. After M2 adds or removes a Gaussian, M1 rebuilds the affected permissions using the new row-to-ID map. Without this step, a permission made for one Gaussian could be applied to another Gaussian after rows move. This order ties the local controls together without forcing all attributes to share the same update rule.

3.4 M2 Structure Update Control

M2 checks up to 256 Gaussians every 64 steps where the new views suggest a change. After an accepted edit, it updates optimizer state, M1 permissions, object assignments, stable IDs, and parent–child links together. The full-update path uses standard 3DGS growth instead.

Evidence for adding, merging, and removing Gaussians. Splitting needs both a visible image error and evidence of a scene change. Removal needs evidence that an object is gone and is delayed if the object may be hidden or the camera pose is uncertain. Merging needs two nearby Gaussians from the same object. The scores use image error R, addition B, undersampling A, change E, uncertainty U, and a cost C for more Gaussians. Removal also uses the effect of removal G_{cf}, survival S, disappearance D, free space F, and opacity O. Merging uses proximity P and separate errors R_i, R_j.

$$\begin{aligned} z_{\mathrm{split}} &= 1.8R + 0.8B + 0.55A + 0.35E + 0.25U - C, \\ z_{\mathrm{death}} &= 1.6G_{\mathrm{cf}} + 0.9(1-S) + 0.6D + 0.45E_{\mathrm{destr}} + 0.35F + 0.25(1-O) - 0.75R + C, \\ z_{\mathrm{merge}} &= 1.1P + 0.45\min(S_i, S_j) + 0.25[1 - \max(U_i, U_j)] - 1.2\max(R_i, R_j) + C. \end{aligned} \tag{3}$$

A sigmoid with temperature 0.35 gives split, removal, and merge thresholds of 0.62, 0.72, and 0.68. Splitting also needs pixel support above 10^{-4} and change evidence. Merging needs a shared object and survival above 0.5. New Gaussians inherit nearby geometry and part of their parent's opacity. Merged Gaussians combine geometry and color by opacity. M3 records earlier values before the edit. These hand-set scores need further study.

Each operation also has a direct check on its input. A split is rejected when the Gaussian has no support in the new images, even if the change score is high. An accepted child inherits its parent's object assignment, moves within the parent's Gaussian extent, and starts with a smaller scale. A removal is delayed for a hidden object or an uncertain camera pose, since a lack of visible pixels does not show that the object has gone. A merge is considered only for nearby Gaussians assigned to the same object. Their color and geometry are combined using opacity weights. M3 saves the earlier values before any of these operations, then records which IDs were created or removed. Finally, M2 updates the optimizer rows, M1 permissions, and row-to-ID map together. The scores and checks describe the local path, while the ten-scene results in Section 4.1 use standard 3DGS growth.

3.5 M3 Historical Replay Control

Each Gaussian has a stable ID (UID) and an active interval $[b_u, d_u)$. M3 saves affected values before a local edit and records new and removed Gaussians and parent–child links. It saves a full snapshot before each full-update step. To recover state τ, it reverses records from a later snapshot or the current state by UID.

$$S_\tau = D_{\tau+1}^{-1} \circ D_{\tau+2}^{-1} \circ \cdots \circ D_T^{-1}(S_T). \tag{4}$$

Here D^{-1} restores saved values. Local records store only affected values. Full snapshots use more space, and a query takes longer when more edits must be reversed.

Let ΔN_t count affected Gaussians at step t, K count full snapshots, F count stored values per Gaussian, and $\bar{N}$ denote the typical Gaussian count. Storage grows as $\mathcal{O}(\sum_t \Delta N_t F + K\bar{N}F)$ before metadata or compression. M3 saves every Gaussian that M1 or M2 may change before the change. A later snapshot cannot restore earlier unsaved values. Gaussian rows may move without changing their UIDs.

Recovery requires every changed value and all added or removed UIDs and parent–child links to be recorded. Reversing these records restores earlier states despite row moves, up to storage precision. Any missed change breaks this condition. M3 can also render an earlier RGB image for training. Optional depth and feature losses are unused.

The record is therefore only as complete as the edits sent to it. A local record stores the UIDs and values touched by its update. If another process may change any Gaussian outside that set, M3 saves a full snapshot before the process runs. An earlier snapshot cannot repair an even earlier value that was never saved. This explains why the full-update path uses more space than a strictly local path. It also makes clear that showing an earlier image is weaker evidence than checking the saved Gaussian values and parent–child links directly.

During training, M3 renders an earlier state without gradients and compares the current image with its RGB target at visible, reliable pixels Ω_τ.

$$\mathcal{L}_{\text{replay}} = \frac{1}{|\Omega_\tau|} \sum_{p \in \Omega_\tau} \left\| \mathcal{R}(S_{\text{current}}, C_\tau)_p - \mathcal{R}(S_\tau, C_\tau)_p \right\|_1 . \tag{5}$$

The implementation uses up to four cameras per earlier scan and applies this loss every 16 steps.

Optimization. Training uses an L1 and D-SSIM image loss ($\lambda_{\text{DSSIM}} = 0.2$) and terms for unchanged areas, object motion, new Gaussians, earlier RGB views, changes to the Gaussian set, and uncertainty. M1 limits which Gaussian attributes the optimizer can update. Reported weights include $\lambda_{\text{rigid}} = 0.10$, $\lambda_{\text{replay}} = 0.02$, $\lambda_{\text{topo}} = 5 \times 10^{-4}$, and $\lambda_{\text{unc}} = 10^{-3}$. The code also includes terms for opacity, color range, and support for new Gaussians. We do not report a study of these weights.

4 Experiments

Data and sources of results. We use ten WAT scenes (Cai & Müller, 2023) with COLMAP camera poses (Schönberger & Frahm, 2016) and hold out every eighth camera. Higher PSNR and SSIM (Wang et al., 2004) and lower LPIPS (Zhang et al., 2018) mean better images. CLNeRF and CL-Splats are supplied reproductions without complete run logs. GaussianUpdate PSNR and SSIM come from its paper (Zeng et al., 2025). Its LPIPS is not reported. Different methods and test settings prevent a controlled ranking.

Paths used in the experiments. The ten-scene results use the full-update path. After $t = 0$, it opens all Gaussian attributes, uses standard 3DGS growth, and stores full snapshots. The Car study has one change, one seed, and 10,000 steps per time. Its ControlGS row may use the full-update path while comparison rows request local updates. Without per-row path logs, this study cannot isolate M1 or M2. All rows use an Adam-based 3DGS model with degree-2 color coefficients. Named comparison rows replace parts of this model rather than reproducing the complete named methods.

The ten scenes are Living Room, Community, Ninja, Street, Spa, Breville, Kitchen, Mac, Car, and Grill. We report the mean of the ten scene values for each available metric. The Car pilot concerns a single transition from one image set to the next. Its row labels identify which part of a shared Gaussian model was changed, not a complete rerun of every cited baseline. The training schedule and stored history may also differ by row. These limits make the pilot useful for examining the reported outcomes, but they prevent a claim that a single control caused the difference.

4.1 Comparison Study

Table 2. WAT results by scene. Red and orange mark the best and second-best available values for each metric. CLNeRF and CL-Splats are supplied reproductions. GaussianUpdate PSNR and SSIM come from its paper (Zeng et al., 2025). Its LPIPS is not reported. Evaluation settings differ across methods.

	Living Room			Community			Ninja			Street			Spa		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
CLNeRF (repr.)	25.57	0.778	0.252	24.15	0.633	0.347	26.89	0.836	0.295	20.23	0.598	0.410	22.71	0.689	0.363
CL-Splats (repr.)	27.81	0.940	0.143	26.04	0.889	0.205	27.95	0.920	0.189	22.51	**0.772**	0.272	23.98	0.821	0.225
GaussianUpdate (reported)	26.10	0.881	–	23.88	0.764	–	27.75	0.921	–	22.80	0.703	–	**27.84**	**0.891**	–
Only new	26.51	0.864	0.139	23.24	0.745	0.203	26.44	0.914	0.190	22.61	0.693	0.297	23.24	0.815	0.237
Ours	**29.22**	**0.943**	**0.113**	**26.71**	**0.892**	**0.202**	**28.14**	**0.922**	**0.180**	**23.16**	0.770	**0.247**	24.25	0.864	**0.216**

	Breville			Kitchen			Mac			Car			Grill		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
CLNeRF (repr.)	27.08	0.810	0.211	25.70	0.676	0.310	26.31	0.732	0.382	21.52	0.557	0.439	22.42	0.534	0.387
CL-Splats (repr.)	30.75	0.944	0.088	28.49	0.917	0.132	30.76	**0.949**	0.159	24.36	0.784	0.186	25.83	0.752	0.159
GaussianUpdate (reported)	30.11	0.927	–	28.02	0.912	–	29.92	0.934	–	23.81	0.742	–	25.31	0.770	–
Only new	29.86	0.922	0.098	27.21	0.904	0.125	28.66	0.917	0.180	24.22	0.728	0.237	24.67	0.737	0.160
Ours	**31.95**	**0.953**	**0.077**	**29.21**	**0.941**	**0.113**	**30.82**	0.938	**0.154**	**24.46**	**0.800**	**0.182**	**26.98**	**0.859**	**0.157**

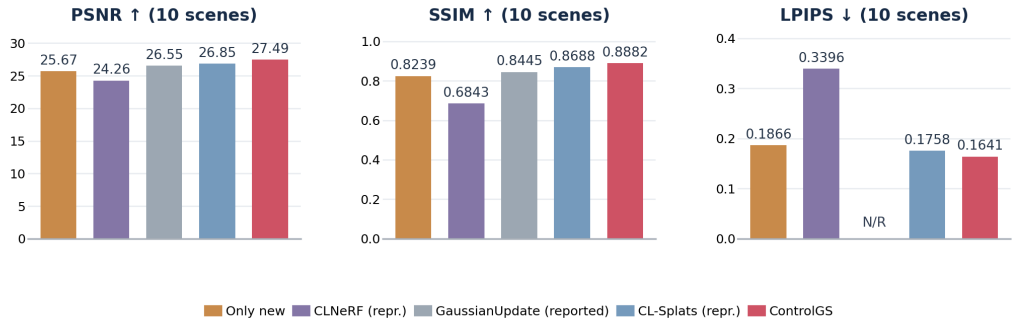

Figure 2. Mean values across the ten WAT scenes in Table 2. Higher PSNR and SSIM and lower LPIPS are better. CLNeRF and CL-Splats are supplied reproductions. GaussianUpdate PSNR and SSIM come from its paper (Zeng et al., 2025). GaussianUpdate LPIPS is not reported. The evaluation settings differ.

Figure 2 summarizes Table 2. ControlGS averages 27.49 dB PSNR, 0.8882 SSIM, and 0.1641 LPIPS. It leads on nine, seven, and ten scenes for these metrics, respectively, among available values. CLNeRF averages 24.26 dB, 0.6843, and 0.3396. The differences are +3.23 dB PSNR, +0.2039 SSIM, and -0.1755 LPIPS. Against the CL-Splats reproduction they are +0.64 dB, +0.0194, and -0.0117. GaussianUpdate reports 26.55 dB and 0.8445 on average but no LPIPS. Different test settings mean these numbers cannot show how much local M1 or M2 control helps.

4.2 Examples From Three WAT Scenes

Figure 3 shows the mouse and lamp in Mac, the car trunk and wheel in Car, and the lamp and window in Living Room. The orange and red insets show details of the new and earlier scenes. These selected images cannot prove that every earlier Gaussian attribute and parent–child link was recovered. That claim also requires complete history records as explained in Section 3.5.

4.3 Ablation Study

As in Section 4.1, we report the mean of the ten WAT scene values for each available image metric. Table 3 compares different ways to update a shared Gaussian model. The M1 group examines updates to existing Gaussian attributes, while the M2 group examines changes to the Gaussian set. For M1, we compare ControlGS attribute control with full fine-tuning, CL-Splats-style updates, and GaussianUpdate-style updates. For M2, we compare ControlGS structure control with native densification and pruning, MCMC-style updates, CL-Splats-style updates, and GaussianUpdate-style updates.

Table 3. Mean new-scene PSNR, SSIM, and LPIPS across the ten WAT scenes for the reported M1 and M2 comparisons. Update paths may differ across rows, so these results do not isolate either control.

Group	Comparison row	PSNR	SSIM	LPIPS
M1	Full fine-tuning	24.99	0.801	0.184
	CL-Splats-style	22.57	0.798	0.257
	GaussianUpdate-style	25.36	0.822	0.197
	ControlGS attribute control	**27.49**	**0.888**	**0.162**
M2	Native densify/prune	22.43	0.741	0.252
	MCMC-style	23.24	0.756	0.255
	CL-Splats-style	25.11	0.799	0.224
	GaussianUpdate-style	26.82	0.818	0.213
	ControlGS structure control	**27.49**	**0.888**	**0.162**

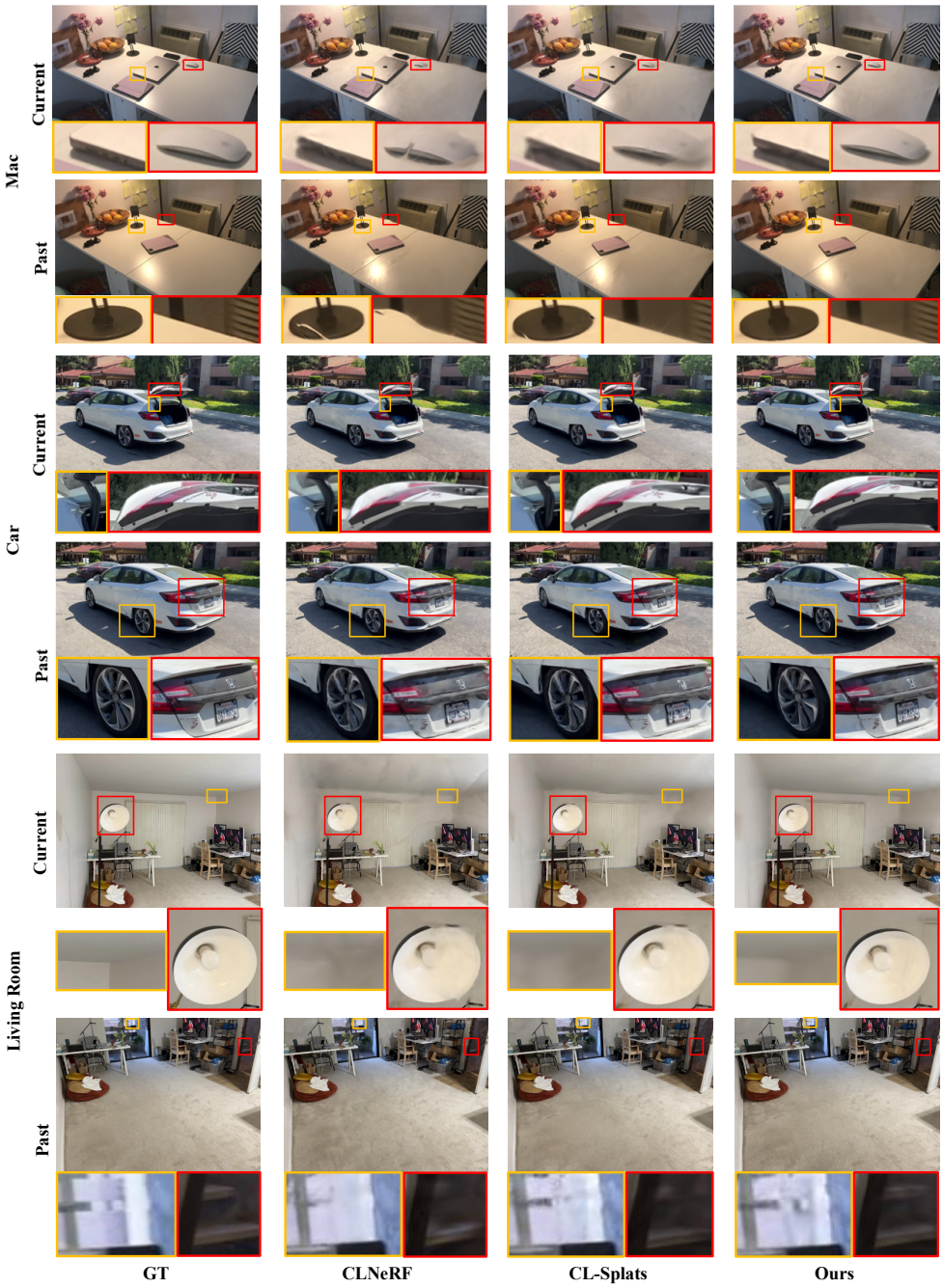

Figure 3. Examples from three WAT scenes. The panels show Mac, Car, and Living Room from top to bottom. Each panel shows the new scene above the earlier scene recovered after the update. The columns show ground truth, CLNeRF (Cai & Müller, 2023), CL-Splats (Ackermann et al., 2025), and ControlGS from left to right. Orange and red boxes mark enlarged regions. In these examples, ControlGS matches the ground truth more closely in the highlighted regions of both the updated and earlier scenes.

In Table 3, the reported ControlGS mean exceeds the next listed mean by 2.13 dB PSNR and 0.066 SSIM in the M1 group and by 0.67 dB and 0.070 in the M2 group. Different update paths may explain part of these gaps, so the gaps do not measure either module's separate effect. The ControlGS LPIPS value in this table is 0.162, while the full-update mean in Section 4.1 is 0.1641. The available records do not explain this difference, so we do not treat the two rows as the same run.

Table 4. Reported means across the ten WAT scenes for changes in earlier-scene image quality and extra history storage. Row-level run files and storage accounting were unavailable for checking. N/R means that a value is not reported.

Comparison row	ΔPSNR (dB) ↑	ΔSSIM ↑	Storage ↓
CL-Splats-style snapshots	−0.10	−0.0100	8–9 GB
CLNeRF-style replay	−0.05	−0.0030	1.17 GB
GaussianUpdate-style visibility	−0.04	N/R	several hundred MB
ControlGS history record	**−0.02**	**−0.0002**	572 MB

Table 4 reports the smallest mean earlier-scene image drop for ControlGS. GaussianUpdate storage is only given as several hundred MB, so it cannot be ranked against 572 MB. The ten-scene means cannot predict storage over longer update sequences. Fair accounting must include every file needed for recovery. Image metrics cannot establish equality of earlier Gaussian attributes and parent–child links. That needs the complete record in Section 3.5.

Focused checks. Four focused software checks were reported. Blocked Gaussian attributes stay fixed even after Adam has built up momentum. A split without image support is rejected. An earlier UID state is restored after a recorded Gaussian is removed and the remaining rows move. A saved full snapshot restores the earlier active Gaussian set after a row is added. These checks do not cover long update sequences, and complete test logs were unavailable for independent review.

Table 5. Focused checks described in the supplied source. A pass applies to the stated constructed case, not to all scene changes or long update sequences.

Control	Constructed check	Reported result
M1	Block an attribute after Adam has built up stored momentum.	Pass
M2	Propose a split with no support from the new images.	Pass
M3 local	Remove a recorded Gaussian and reorder the remaining rows.	Pass
M3 full	Add a row after saving the complete earlier state.	Pass

These checks address failures that image metrics can miss. A model can render a plausible earlier image while its saved Gaussian values differ from the earlier state. Likewise, a zero gradient can still allow a blocked row to move through Adam's stored values. The checks exercise one controlled example of each failure. A longer test should mix additions, removals, merges, full updates, and queries for several earlier states in the same run. It should compare recovered Gaussian attributes and active IDs as well as rendered images.

5 Limitations and Conclusion

Limitations. Continual 3DGS must distinguish lasting scene changes from changes caused by lighting, occlusion, and camera pose across repeated scans. As updates accumulate, it must also balance image quality, Gaussian count, and the cost of storing and recovering earlier states. ControlGS addresses these challenges through change evidence and separate controls. Longer sequences and more varied scenes would help assess how these decisions scale, while matched local-update comparisons would clarify the contribution of each control.

Conclusion. ControlGS organizes continual 3DGS into attribute update control, structure update control, and historical replay control. On ten WAT scenes, the reported full-update path achieves strong reconstruction quality among the available results. The ControlGS rows lead the reported M1 and M2 comparison groups, while M3 shows the smallest mean drop in earlier-scene image quality. Focused checks show that blocked attributes stay fixed, unsupported splits are rejected, and earlier states can be recovered after Gaussian rows change in the tested cases. These results support the proposed approach and provide a basis for studying longer and more varied update histories.

AI USE STATEMENT

Generative AI tools were used to polish wording, correct grammar, improve formatting, and make sentences read more smoothly. The authors take responsibility for the final content.

REFERENCES

Jan Ackermann, Jonas Kulhanek, Shengqu Cai, Haofei Xu, Marc Pollefeys, Gordon Wetzstein, Leonidas J. Guibas, and Songyou Peng. CL-Splats: Continual learning of gaussian splatting with local optimization. In *IEEE/CVF International Conference on Computer Vision*, pp. 7808–7817, 2025.

Zhipeng Cai and Matthias Müller. CLNeRF: Continual learning meets NeRF. In *IEEE/CVF International Conference on Computer Vision*, pp. 23185–23194, 2023.

Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. GaussianEditor: Swift and controllable 3D editing with gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21476–21485, 2024.

Jaeyoung Chung, Kanggeon Lee, Sungyong Baik, and Kyoung Mu Lee. MEIL-NeRF: Memory-efficient incremental learning of neural radiance fields. *IEEE Access*, 13:130420–130429, 2025. doi: 10.1109/ACCESS.2025.3578393.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. SC-GS: Sparse-controlled gaussian splatting for editable dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. SplaTAM: Splat, track & map 3D gaussians for dense RGB-D SLAM. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Jeff Tseng, Hussein Isack, Amlan Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3D Gaussian Splatting as markov chain monte carlo. *arXiv preprint arXiv:2404.09591*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8508–8520, 2024.

Ziqi Lu, Jianbo Ye, and John Leonard. 3DGS-CD: 3D Gaussian Splatting-based change detection for physical object rearrangement. *arXiv preprint arXiv:2411.03706*, 2024.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3D gaussians: Tracking by persistent dynamic view synthesis. In *International Conference on 3D Vision*, 2024.

Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian splatting SLAM. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, 2020.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick

Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Ryan Po, Zhengyang Dong, Alexander W. Bergman, and Gordon Wetzstein. Instant continual learning of neural radiance fields. In *IEEE/CVF International Conference on Computer Vision Workshops*, pp. 3334–3344, 2023.

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. LangSplat: 3D language gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4104–4113, 2016.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, 2017.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D Gaussian Splatting for real-time dynamic scene rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20310–20320, 2024.

Xiuzhe Wu, Peng Dai, Weipeng Deng, Handi Chen, Yang Wu, Yan-Pei Cao, Ying Shan, and Xiaojuan Qi. CL-NeRF: Continual learning of neural radiance fields for evolving scene representation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. GS-SLAM: Dense visual SLAM with 3D Gaussian Splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. *arXiv preprint arXiv:2406.09414*, 2024a.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3D gaussians for high-fidelity monocular dynamic scene reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3D scenes. In *European Conference on Computer Vision*, 2024.

Vladimir Yugay, Thies Kersten, Luca Carlone, Theo Gevers, Martin R. Oswald, and Lukas Schmid. Gaussian mapping for evolving scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18903–18912, 2026.

Lin Zeng, Boming Zhao, Jiarui Hu, Xujie Shen, Ziqiang Dang, Hujun Bao, and Zhaopeng Cui. GaussianUpdate: Continual 3D Gaussian Splatting update for changing environments. In *IEEE/CVF International Conference on Computer Vision*, pp. 25800–25809, 2025.

Letian Zhang, Ming Li, Chen Chen, and Jie Xu. IL-NeRF: Incremental learning for neural radiance fields with camera pose alignment. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 358–368, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.