AFTER TRAINING: TARGETED POST-TRAINING REFINEMENT FOR 4D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Refining a trained dynamic scene requires correcting residual errors without disrupting what has already been reconstructed well. This is challenging in deformation-based four-dimensional Gaussian Splatting (4DGS), where canonical updates affect multiple time steps and newly added Gaussians interact with existing ones through compositing. We propose *targeted post-training refinement*, which controls refinement along both temporal and spatial dimensions. Temporally, additional Gaussians reuse the learned deformation field while learning time-specific corrections for residual errors. Spatially, refinement starts from the source Gaussians used to initialize the new Gaussians and progressively expands to the remaining representation, avoiding unnecessary global updates at the outset. Across two deformation-based 4DGS models and three dynamic-scene datasets, our method consistently improves the input models and outperforms continued-training baselines. Further analysis shows that time-specific correction primarily increases the correction of remaining errors, while restricting the initial refinement scope reduces damage to already well-reconstructed regions.

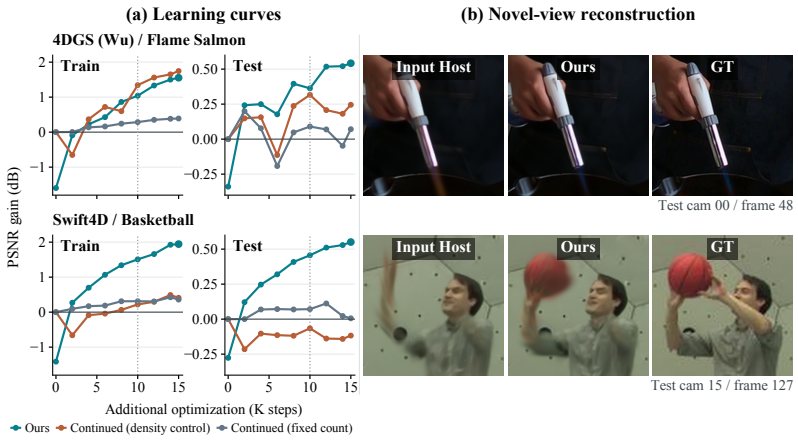


Figure 1: **Unlike simply extending the original training, Ours stably improves Test quality and corrects remaining reconstruction errors.** (a) Continued extends training with Adaptive Density Control (ADC) ON/OFF. Its Train Peak Signal-to-Noise Ratio (PSNR) gains do not translate into Test PSNR gains, whereas Ours generally improves Test quality in both examples. All methods start from the same input Host; the vertical dotted line marks 10K additional steps. (b) At the same Test view, Ours reduces local reconstruction errors remaining in the Host, bringing results closer to Ground-Truth (GT).

1 INTRODUCTION

Advances in dynamic scene representations have enabled four-dimensional Gaussian Splatting (4DGS) to capture time-varying geometry, appearance, and motion in a single learned representation and efficiently generate images at novel viewpoints and times (Wu et al., 2024). Recent studies have continued to improve the training process itself, including motion modeling, Gaussian placement,

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.

and density control, to obtain more accurate and efficient dynamic scene representations. However, post-training refinement has received relatively less attention: further correcting only the remaining errors in a trained dynamic scene while preserving its well-reconstructed content. In practice, it is natural to reuse learned geometry, appearance, and motion to correct these errors rather than retrain the scene from scratch. Beyond longer training, post-training refinement therefore requires deciding both what new corrections to enable and how much of the existing representation to modify. The most straightforward approach extends the original training procedure, further optimizing the entire model and adding Gaussians as needed. Figure 1 shows this continued training from the same pre-trained checkpoint. With adaptive density control (ADC) OFF, the entire model continues training at a fixed Gaussian count; ADC ON reactivates the original ADC, allowing densification and pruning. Both generally improve Train quality, but Test quality does not improve proportionally. In some scenes, Test quality even falls below that of the input model despite added Gaussians, and Train and Test quality can change in opposite directions. Our method starts from the same input model but improves Test quality more stably. These results show that post-training refinement is not simply a matter of extending training or increasing representational capacity.

We therefore focus on two design questions. First, we need to selectively correct time-specific residual errors without unnecessarily altering learned motion. Because deformation-based 4DGS feeds canonical Gaussians and time into a shared deformation field (Wu et al., 2024; 2025), changes to the canonical representation can affect the states at multiple times. This motivates additional temporal degrees of freedom for adjusting corrections across time separately from canonical updates. Second, we need to determine the appropriate scope of the existing representation to modify alongside the new correction. The contribution of a new Gaussian is determined by alpha compositing with existing Gaussians (Kerbl et al., 2023), but jointly optimizing the entire representation from the outset risks changing regions that have already been reconstructed well. We therefore need a refinement scope that allows interactions with the new correction while limiting unnecessary changes.

We propose *targeted post-training refinement*, which reuses the frozen deformation field while adding time-specific correction capacity and progressively expanding the refinement scope of the existing representation from regions requiring correction. First, new Gaussians reuse existing motion and learn additional time-specific corrections. We select existing Gaussians requiring further correction as *Parents* and generate *Children* from them. Each Child queries the existing deformation field at its own canonical position, then adds a separate time-specific correction to the resulting state. This allows selective correction of time-specific residual errors without changing the shared field's weights. Second, we progressively refine the existing representation from regions requiring correction: jointly optimizing Children with their source Parents, then including Non-parents, rather than modifying the entire representation from the outset. Our main contributions are as follows.

- We show that, in post-training refinement of a trained 4DGS model, most of the existing representation is already well learned, and global readjustment to correct remaining errors can instead damage the existing reconstruction. We therefore study post-training refinement of deformation-based 4DGS: correcting residual errors while limiting damage to the existing reconstruction.
- To this end, we propose *targeted post-training refinement*, which enables time-specific corrections and spatially expands the refinement scope from regions directly connected to the new correction. The added Child Gaussians reuse the learned deformation field while learning time-specific corrections, and the scope of joint optimization of the existing representation progressively expands from the source Parents of the Children to a broader set of surrounding Gaussians.
- We empirically demonstrate that post-training refinement centers on balancing further correction of remaining errors with preservation of well-reconstructed regions, and systematically analyze how temporal selectivity and a progressive refinement scope affect these two aspects, respectively.

2 Related Work

Dynamic Gaussian Scene Representations. Dynamic Gaussian representations encode motion through time-conditioned canonical deformation or explicit time-varying states. Deformation-based 4DGS predicts deformations with spatial and temporal feature planes (Wu et al., 2024); Grid4D and

Swift4D use spatial and temporal hash encoding and static/dynamic Gaussian separation, respectively (Xu et al., 2024; Wu et al., 2025). SC-GS and MoDec-GS model motion with sparse control points or global/local decomposition (Huang et al., 2024; Kwak et al., 2025). Other formulations include STG (temporal opacity), Ex4DGS (keyframe interpolation), and SPIN-4DGS (attribute prediction from spatiotemporal positions) (Li et al., 2024; Lee et al., 2024; Kim et al., 2026).

Gaussian Densification. ADC adds Gaussians in underrepresented regions during training. Three-dimensional Gaussian Splatting (3DGS) clones and splits based on screen-space position gradients and Gaussian size (Kerbl et al., 2023). Subsequent methods improve densification selection using pixel errors (Rota Bulò et al., 2024), gradient magnitudes and directions (Ye et al., 2024; Zhou et al., 2026), and multi-view geometry (Yang et al., 2025). Some dynamic-scene densification methods also consider time-specific visibility and temporal support (Sandu et al., 2026). These methods primarily address where to add Gaussians during training, whereas further densification and continued optimization in mostly learned representations risk unnecessarily changing well-reconstructed regions while correcting remaining errors.

Localized Scene Refinement. Localized methods identify erroneous regions in existing Gaussian scenes and selectively add or reoptimize representation there rather than retraining the whole scene. For static scenes, Augmented Radiance Field adds Gaussians by back-projecting image-space-optimized two dimensional (2D) Gaussians to 3D, then jointly optimizes new Gaussians and existing Gaussian opacity (Yang et al., 2026). For dynamic scenes, CEM4DGS distinguishes missing color from occlusion using rendering errors and multi-view color consistency, then respectively adds new splats and splits foreground splats to correct these errors (Kang et al., 2025). Urban-GS identifies views with unstable training PSNR changes and selectively reoptimizes associated and newly generated anchors (Wang et al., 2026). Existing localized refinement methods address error-driven Gaussian addition, temporal visibility, or selective adjustment of existing attributes. In deformation-based dynamic scenes, however, canonical edits can affect multiple times, and the effects of new Gaussians depend on compositing with existing ones. We account for these temporal and spatial couplings through Child-specific temporal residuals on a frozen deformation field and progressive joint refinement starting from source Parents.

3 METHOD

3.1 PRELIMINARIES

Deformation-based 4DGS. 3DGS represents a scene by Gaussians $g_i = (\boldsymbol{\mu}_i, \mathbf{s}_i, \mathbf{r}_i, o_i, \mathbf{h}_i)$ with position, scale, rotation quaternion, opacity, and spherical harmonics (SH) coefficients (Kerbl et al., 2023). Deformation-based 4DGS feeds canonical position and time into a shared deformation field to predict time-varying attribute changes (Wu et al., 2024; 2025). For example, positions follow

$$\boldsymbol{\Delta}_i(t) = F_\phi(\boldsymbol{\mu}_i, t), \qquad \boldsymbol{\mu}_i(t) = \boldsymbol{\mu}_i + \Delta\boldsymbol{\mu}_i(t), \tag{1}$$

where $\Delta\boldsymbol{\mu}_i(t)$ is the position component of $\boldsymbol{\Delta}_i(t)$. Other attributes, including scale, rotation, and opacity, follow each input model's deformation to form $g_i(t)$.

Gaussian compositing. Gaussians projected onto the image plane at time t yield pixel colors by depth-ordered alpha compositing (Kerbl et al., 2023; Kwak et al., 2025). Omitting camera/time subscripts and the background term,

$$\hat{I}(\mathbf{p}) = \sum_{i=1}^{N} T_i(\mathbf{p})\alpha_i(\mathbf{p})\mathbf{c}_i, \qquad T_i(\mathbf{p}) = \prod_{k=1}^{i-1}\big(1-\alpha_k(\mathbf{p})\big), \tag{2}$$

Here, $\alpha_i(\mathbf{p})$ is Gaussian i's pixel alpha, $T_i(\mathbf{p})$ the transmittance accumulated from preceding Gaussians, and $\mathbf{c}_i$ the color obtained by evaluating its SH coefficients along the viewing direction.

3.2 OBSERVATION: TEMPORAL AND SPATIAL COUPLING IN POST-TRAINING REFINEMENT

We observe that corrections to a trained dynamic Gaussian scene are difficult to make independently of its existing representation in time and space. Canonical updates affect states at multiple times, while the effects of new Gaussians depend on compositing with surrounding existing Gaussians.

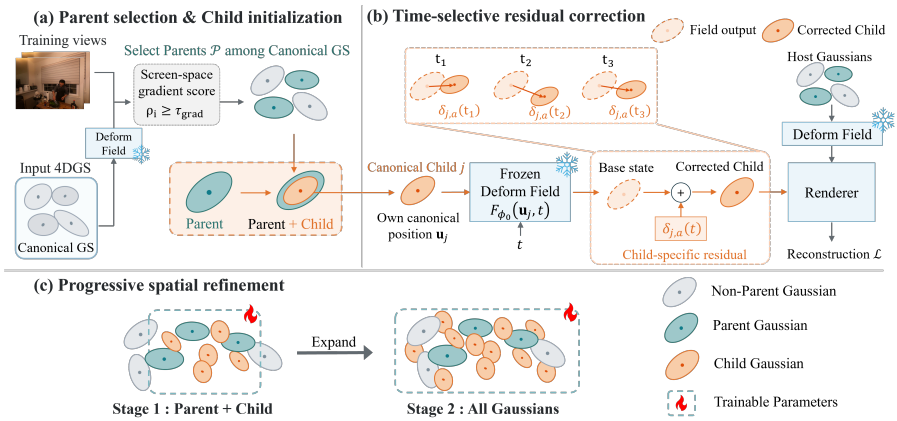


Figure 2: **Overview of targeted post-training refinement.** (a) Screen-space position gradients select existing Gaussians that require correction as Parents, and a new Child Gaussian is initialized from each Parent. (b) Each Child reuses the existing motion through the frozen deformation field at its own canonical position, and adds its own time-specific correction $\delta_{j,a}(t)$ to selectively correct residual errors at individual times. (c) The optimization scope progressively expands from Parents and Children to Non-parents. Existing Gaussians retain their count and only their attributes are adjusted, while densification and pruning apply only to Children.

These couplings indicate that post-training refinement should consider both a correction's temporal extent and the scope of changes to the existing representation.

Temporal coupling. With the learned deformation field frozen, let $X(\mathbf{q}, t) = \mathbf{q} + [F_{\phi_0}(\mathbf{q}, t)]_\mu$ denote the time-varying transformation of a canonical position $\mathbf{q}$. Applying a change $\mathbf{u}$ to the canonical position yields the following position change at time t:

$$\Delta_{\text{can}}(t) = X(\mathbf{q} + \mathbf{u}, t) - X(\mathbf{q}, t) \tag{3}$$

A single canonical position change $\mathbf{u}$ is shared across multiple times, so a correction intended for one time can also affect states at other times. This motivates a time-specific correction path that can be adjusted separately from canonical position updates while keeping the deformation field frozen.

Spatial coupling. The image contribution of a new Gaussian is likewise not determined independently of the existing representation. Along a ray, let C_{pre} and T_{pre} be the accumulated color and transmittance in front of the new Gaussian, and let B be the composited color behind it, including the background. The original pixel color is $I_{\text{old}} = C_{\text{pre}} + T_{\text{pre}}B$. Inserting a new Gaussian $(\alpha_c, \mathbf{c}_c)$ gives, from Equation (2),

$$I_{\text{new}} - I_{\text{old}} = T_{\text{pre}}\alpha_c(\mathbf{c}_c - B) \tag{4}$$

The contribution of the new Gaussian depends on the transmittance determined by the Gaussians in front of it, while also affecting the contributions of those behind it. Thus, new corrections operate in conjunction with the surrounding existing representation. We must consider not only adding Gaussians for correction, but also how broadly to jointly adjust the existing Gaussians.

3.3 Time-Selective Residual Correction

To address temporal coupling, we use additional Gaussians to provide a time-specific correction path separate from canonical updates, with the existing deformation field frozen (Figure 2). We first select existing Gaussians requiring further correction in the training images as *Parents* and generate *Children* from them (Figure 2(a)). Each Child queries the frozen field at its own canonical position to reuse existing motion, then adds a separate time-specific correction to the result (Figure 2(b)).

Selecting Gaussians for refinement. Before refinement, we briefly probe for Gaussians requiring further correction. Without updating model parameters, we render training images in multiple batches and backpropagate each batch's reconstruction loss to collect screen-space position gradients. Each gradient is taken with respect to a Gaussian's projected 2D center, and its magnitude

measures the loss sensitivity to that position. Let $\mathbf{g}_{ib}$ be the gradient-vector sum for existing Gaussian $i \in \mathcal{H}$ in batch b, and $\mathcal{V}_i$ the set of batches where Gaussian i appears on screen. For Gaussians with $|\mathcal{V}_i| > 0$, we define the average gradient magnitude and Parent set as follows:

$$\rho_i = \frac{1}{|\mathcal{V}_i|} \sum_{b \in \mathcal{V}_i} \|\mathbf{g}_{ib}\|_2, \qquad \mathcal{P} = \{i \in \mathcal{H} : |\mathcal{V}_i| > 0,\ \rho_i \geq \tau_{\text{grad}}\}. \tag{5}$$

In Equation (5), ρ_i averages the magnitudes of batch-summed gradient vectors over batches where the Gaussian is observed; τ_{grad} is the Parent selection threshold. We thus select Gaussians whose screen-space position changes strongly affect reconstruction loss as *Parents* requiring further correction. We initialize one *Child* per selected Parent to form $\mathcal{C}$, copying that Parent's canonical position, rotation, and SH coefficients but setting a smaller scale and lower opacity. Each Child is then optimized as a separate Gaussian.

Time-selective correction. Each Child reuses the learned deformation field while separately learning an additional correction at each time. For Child j with learnable canonical position $\mathbf{u}_j$, the field gives its base position at time t; adding the Child-specific time correction $\boldsymbol{\delta}_{j,\mu}(t)$ gives its final position:

$$\bar{\boldsymbol{\mu}}_j(t) = \mathbf{u}_j + \Delta\boldsymbol{\mu}_{\phi_0}(\mathbf{u}_j, t) \tag{6}$$

$$\boldsymbol{\mu}_j(t) = \bar{\boldsymbol{\mu}}_j(t) + \boldsymbol{\delta}_{j,\mu}(t), \qquad j \in \mathcal{C}. \tag{7}$$

Thus, each Child queries the existing deformation field at its own canonical position rather than directly copying its Parent's motion, then adds the required time-specific position correction. Field weights remain frozen, but gradients with respect to $\mathbf{u}_j$ are retained to optimize the Child's canonical position. We represent time-varying corrections as a linear combination of fixed temporal basis functions. The correction of Child j for attribute $a \in \{\mu, s, r, o\}$ is defined as

$$\boldsymbol{\delta}_{j,a}(t) = \sum_{m=1}^{M} b_m(t)\mathbf{w}_{j,a,m} \tag{8}$$

where $b_m(t)$ is a hierarchical piecewise-linear temporal basis function (Bungartz & Griebel, 2004), and $\mathbf{w}_{j,a,m}$ is the Child-specific learnable coefficient for that basis function. Because the basis-function values $b_m(t)$ vary with t, each Child can correct its position, scale, rotation, and opacity differently over time. The additional SH correction is shared across all times. Scale and opacity are corrected in log-space and logit-space, respectively, while rotation corrections are represented in axis-angle form, converted to quaternions, and applied. The Child's canonical position and base attributes are also optimized, and the deformation field weights remain frozen throughout the process.

3.4 Progressive Spatial Refinement

Since the effects of new Child corrections depend on compositing with surrounding existing Gaussians, we first jointly adjust Children and their directly connected source Parents rather than Children alone. Optimizing all existing Gaussians from the outset risks unnecessarily changing well-reconstructed regions, so we initially freeze Non-parents (gray), then include them to progressively expand the refinement scope of the existing representation (Figure 2(c)).

Parent–Child refinement. Each Child is generated at the same position as its Parent selected by Equation (5) and initialized from that Parent's attributes, so the two Gaussians are directly linked to the correction of the same region. During the first K_1 iterations, we optimize only Parent and Child base attributes and Child temporal correction coefficients, jointly adjusting the new correction and its source representation first. Non-parents remain frozen but included in rendering, restricting only the update scope while preserving the entire existing scene's compositing relationships.

Progressive scope expansion. However, it is difficult to identify in advance all existing Gaussians that interact with the Children based solely on the initially selected Parents. As optimization changes Child positions, scales, and opacities, their compositing relationships with existing Gaussians can also change. After K_1, we therefore retain the learned Parents and Children and expand joint adjustment from Children and their directly connected Parents to Non-parents for K_2 additional iterations, preserving the learned correction. We neither reselect Parents nor reinitialize Children, and the shared deformation field remains frozen.

Both stages apply the same reconstruction loss to the full training images (Kerbl et al., 2023):

$$\mathcal{L} = (1 - \lambda) \|\hat{I} - I\|_1 + \lambda \left(1 - \mathrm{SSIM}(\hat{I}, I)\right). \tag{9}$$

Here, I and $\hat{I}$ are the training image and rendered result, respectively, and $\|\cdot\|_1$ is the mean absolute error and SSIM denotes the Structural Similarity Index Measure. Densification and pruning apply only to Children; the existing Gaussian count stays fixed. We allow Child cloning and splitting during K_1 to form the required correction representation. Densification stops during K_2, which focuses on jointly adjusting the resulting Children and existing representation. Low-opacity Child pruning continues in both stages; §4.1 gives the training settings.

4 Experiments

4.1 Experimental Setup

To evaluate post-training refinement of trained dynamic scenes and the contributions of time-selective correction and progressive joint adjustment to quality improvements, we use 4DGS (Wu) (Wu et al., 2024) and Swift4D (Wu et al., 2025), which have different deformation architectures. Each pretrained model serves as the *Host*. We use publicly available official checkpoints when available; otherwise, refinement starts from models trained to approach the originally reported performance. We do not assume full Host convergence. All methods start from the same Host checkpoint and perform additional optimization to compare its effects. The test scenes, views, and resolutions follow existing evaluation settings for 6 N3DV scenes (Wu et al., 2024; 2025), 5 Technicolor scenes (Lee et al., 2024), and 6 Panoptic Sports scenes (Kim et al., 2026). We measure PSNR, SSIM (Wang et al., 2004), and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018) on the existing Test views for a total of 34 combinations of the two Hosts and 17 scenes. Unless otherwise stated, we select Parents using gradients from 500 Train batches with $\tau_{\mathrm{grad}} = 8 \times 10^{-5}$. Parent–Child refinement runs for $K_1 = 10{,}000$ iterations, followed by $K_2 = 5{,}000$ iterations over all Gaussians. We use loss weight $\lambda = 0.2$ and $M = \lceil F/2 \rceil$ temporal basis functions, where F is the frame count. Optimization starts with a fresh Adam optimizer (Kingma & Ba, 2015); its state and learning rates carry over between stages. All experiments (batch size 2) use RTX 4090 GPUs.

4.2 Post-Training Refinement Performance

All methods perform 15,000 additional optimization iterations from the same Host. Continued training jointly optimizes Gaussian attributes and the deformation field; Frozen-field training optimizes only the attributes with the field frozen. Neither uses additional Child-specific time corrections. Each baseline keeps the Gaussian count fixed or uses density control during the first 10,000 iterations. Child-based methods share initial Children and temporal representation: Same-Child Joint optimizes all existing Gaussians from the outset, P+C-only optimizes only Parents and Children throughout, and Ours expands joint optimization from P+C to all Gaussians. Continued and Frozen-field training follow each Host's learning-rate schedule from the input checkpoint iteration. Within each Host–scene comparison, Child-based methods share learning-rate settings for corresponding trainable parameter groups.

Effect of post-training refinement. In Table 1, Ours consistently improves PSNR, SSIM, and LPIPS over the input Host for both Hosts and all three datasets. In contrast, all four Continued/Frozen-field conditions decrease mean PSNR on N3DV, and whether quality improves or declines on Panoptic also depends on the Host. Moreover, on Swift4D N3DV, Ours achieves 0.30 dB higher PSNR than Continued training with density control despite nearly identical Gaussian counts (328.1K/328.5K), and achieves higher PSNR than the Frozen-field baselines in all Host–dataset combinations. The gains therefore cannot be explained solely by additional training, an increase in Gaussians, or freezing the field.

Effect of the joint optimization scope. Ours achieves higher PSNR than Same-Child Joint in all 24 Host–scene combinations on N3DV and Panoptic, confirming the benefit of starting joint adjustment from Parents directly connected to the new correction. Compared with P+C-only, expanding the scope to Non-parents provides up to 0.37 dB of additional gain on Technicolor and on Panoptic with 4DGS (Wu). This shows that Parents provide a useful starting point, while expanding the refinement scope to a broader portion of the existing representation is effective when needed.

Table 1: **Additional training results from the same input Host.** ΔPSNR is the change relative to the Host (dB). The **best** and second-best quality are highlighted for each Host–dataset combination.

	4DGS (Wu)					Swift4D				
Method	PSNR↑	ΔPSNR↑	SSIM↑	LPIPS↓	#GS (K)	PSNR↑	ΔPSNR↑	SSIM↑	LPIPS↓	#GS (K)
N3DV (6 scenes) (Li et al., 2022)										
Input Host	31.5059	–	0.9376	0.0579	124.3	32.0151	–	0.9450	0.0470	307.3
Continued (fixed count)	31.2502	−0.2557	0.9378	0.0554	124.3	31.8295	−0.1855	0.9440	0.0455	307.3
Continued (density control)	31.2324	−0.2735	0.9414	0.0489	275.9	31.8511	−0.1640	0.9442	0.0452	328.1
Frozen-field (fixed count)	31.2924	−0.2135	0.9381	0.0561	124.3	31.8053	−0.2097	0.9438	0.0460	307.3
Frozen-field (density control)	31.4594	−0.0465	0.9403	0.0501	269.9	31.8120	−0.2030	0.9442	0.0459	328.1
Same-Child Joint	31.5135	+0.0076	0.9421	**0.0478**	198.2	31.8835	−0.1316	0.9447	**0.0442**	329.0
P+C-only	31.7403	+0.2344	**0.9429**	0.0481	197.8	32.1519	+0.1369	**0.9463**	0.0445	328.5
Ours	**31.7516**	+0.2457	**0.9429**	0.0479	197.8	**32.1520**	+0.1369	0.9462	0.0444	328.5
Technicolor (5 scenes) (Sabater et al., 2017)										
Input Host	29.1747	–	0.8169	0.2179	291.3	31.5621	–	0.8930	0.1139	332.7
Continued (fixed count)	29.5049	+0.3302	0.8329	0.1965	291.3	31.9460	+0.3839	0.8976	0.1066	332.7
Continued (density control)	28.7537	−0.4210	0.8316	0.2179	289.1	30.5447	−1.0173	0.8794	0.1298	388.6
Frozen-field (fixed count)	29.3880	+0.2132	0.8283	0.2051	291.3	31.8200	+0.2579	0.8961	0.1088	332.7
Frozen-field (density control)	28.1252	−1.0495	0.8194	0.2363	277.5	30.3373	−1.2248	0.8777	0.1310	390.3
Same-Child Joint	**31.6538**	+2.4791	**0.8809**	**0.1328**	478.7	**32.4566**	+0.8945	**0.9021**	**0.1006**	385.2
P+C-only	31.3447	+2.1700	0.8748	0.1410	478.6	32.1436	+0.5816	0.8987	0.1062	383.7
Ours	31.5274	+2.3527	0.8792	0.1359	478.6	32.4491	+0.8870	0.9013	0.1030	383.7
Panoptic (6 scenes) (Joo et al., 2015; Luiten et al., 2024)										
Input Host	27.2443	–	0.9015	0.1471	189.0	27.7380	–	0.9237	0.0710	238.0
Continued (fixed count)	27.8988	+0.6544	0.9113	0.1029	189.0	27.0052	−0.7328	0.9150	0.0841	238.0
Continued (density control)	27.9464	+0.7021	0.9148	**0.0922**	195.5	27.6286	−0.1094	0.9193	0.0747	245.0
Frozen-field (fixed count)	27.6862	+0.4419	0.9070	0.1155	189.0	27.6135	−0.1245	0.9184	0.0771	238.0
Frozen-field (density control)	27.7249	+0.4806	0.9113	0.1028	196.2	27.5201	−0.2180	0.9178	0.0766	244.3
Same-Child Joint	27.8439	+0.5996	0.9123	0.1041	279.9	27.7786	+0.0405	0.9207	0.0737	279.6
P+C-only	27.7352	+0.4908	0.9127	0.1251	271.5	27.9993	+0.2613	**0.9270**	**0.0650**	276.7
Ours	**28.1050**	+0.8606	**0.9174**	0.0992	271.5	**28.0009**	+0.2629	0.9253	0.0672	276.7

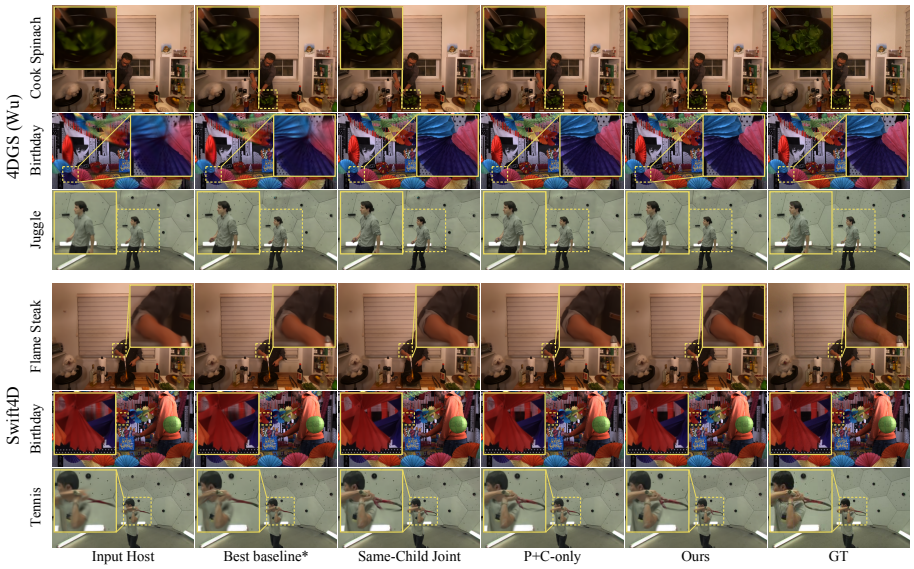


Figure 3: **Qualitative comparisons across two Hosts and three datasets.** Table 1 checkpoints are rendered at the same Test camera and time, with the same dashed regions enlarged. Best baseline* has the highest mean PSNR among the four Continued/Frozen-field conditions for each Host–dataset combination and is fixed across all scenes and frames. Appendix F shows all conditions.

Qualitative comparison. In Figure 3, Ours reconstructs local structures more clearly, including the boundaries of spinach leaves and a sleeve, folds in paper decorations, and the rim of a racket. In particular, although the enlarged region of Swift4D Flame Steak occupies only about 1.2% of the image, Ours achieves region PSNR that is 3.76 dB higher than the input Host and 0.44 dB higher than

Same-Child Joint. This shows that gains that appear small in the overall average can be pronounced in the correction of local residual errors.

4.3 Ablation Studies and Analysis

To address the temporal and spatial coupling in §3.2, we analyze time-selective correction and a progressive joint optimization scope. We first separate their effects on improvement and damage in final quality, then examine Child generation locations and joint optimization of source Parents. Finally, we separately change time-specific position corrections and Parent outputs and rerender to test their contributions to the final reconstruction.

Table 2: **Time-specific correction and initial P+C restriction.** Test PSNR and total Gaussian count (K). For initial P+C restriction, ✓/– indicate enabled/disabled; for time-specific correction, ✓/– indicate time-specific/time-shared values. Ours enables both elements. All conditions allow Child densification during the first 10,000 iterations. The **best** and second-best PSNR (dB) are highlighted for each Host–scene combination. A detailed improvement/damage analysis is provided in Appendix C.

		4DGS (Wu)						Swift4D					
		Flame Steak		Painter		Basketball		Flame Steak		Painter		Basketball	
Initial P+C restriction	Time-specific correction	PSNR↑	#GS (K)	PSNR↑	#GS (K)	PSNR↑	#GS (K)	PSNR↑	#GS (K)	PSNR↑	#GS (K)	PSNR↑	#GS (K)
–	–	32.4224	170.9	35.9719	389.7	26.8101	305.5	32.7730	281.5	34.0902	542.0	27.1046	291.7
–	✓	32.5944	174.7	36.6006	391.2	27.4405	321.6	32.8251	292.1	33.9357	532.2	27.5613	303.2
✓	–	32.7254	172.4	35.9934	385.3	27.1919	290.4	32.7514	282.1	33.8938	543.9	27.2427	287.4
✓	✓	**32.9323**	175.3	**36.6098**	384.2	**27.6630**	307.8	**33.6939**	254.5	**34.2337**	531.7	**27.7363**	298.1

Roles of temporal selectivity and the initial refinement scope. Table 2 cross-compares time-specific correction and the initial P+C restriction. Without time-specific correction, the additional Child correction is shared across all times; without the initial P+C restriction, Non-parents are also optimized from the outset. Figure 4 aggregates decreases and increases in pixel squared error relative to the Host as improvement and damage for the same models.

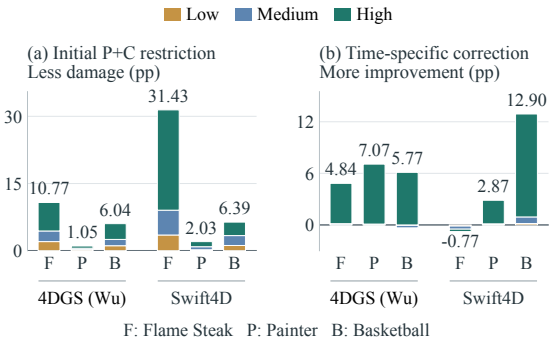


Figure 4: **Changes in improvement and damage.** (a) Damage reduction from initial P+C restriction with time-specific correction retained. (b) Additional improvement from time-specific correction with initial P+C restriction retained. Colors denote initial Host error bins (low/medium/high); numbers are sums across bins. Differences are percentage points (pp) relative to total Host SSE; positive values are favorable.

When comparing time-specific correction under the same initial refinement scope, Test PSNR improves in 11 of the 12 comparisons across the two Hosts and three scenes. In particular, with the initial P+C restriction, improvement in initially high-error regions increases in five of the six combinations (Figure 4(b)). Conversely, when comparing the initial P+C restriction while retaining time-specific correction, damage to the existing reconstruction decreases in all six combinations (Figure 4(a)). For 4DGS (Wu) Flame Steak and Basketball, final Test quality improves even when Train PSNR decreases because the reduction in damage exceeds the reduction in improvement. These results show complementary roles: time-specific correction primarily contributes to further correction of remaining errors, whereas the initial P+C restriction reduces unnecessary damage to the existing reconstruction.

The initial P+C restriction improves quality with similar Gaussian counts on 4DGS (Wu) Flame Steak and fewer Gaussians on Basketball for both Hosts, so increased counts alone cannot explain the gain. In cases such as Swift4D Flame Steak, the gain from time-specific correction manifests as reduced damage rather than additional improvement, showing that both must be considered to explain final quality. In a controlled experiment matching the amount of correction at the target frame, time-specific edits could also concentrate their effects within a limited time interval (Appendix A).

Choice of Child generation locations and joint optimization targets. Table 3 tests two Parent roles. (a) initializes equal numbers of Children from Gaussians with large screen-space position gradients or random Gaussians; selected sources are initially optimized together as Parents. Gradient source yields higher quality in all six combinations of two Hosts and three scenes, with particularly large Painter gains of 0.80/1.14 dB for 4DGS (Wu)/Swift4D, respectively. (b) retains identical initial Children and varies only the jointly optimized existing Gaussians. Controls freeze actual source Parents during the first stage and optimize equal numbers of random Non-parents or Non-parents matched in position, scale, opacity, and visibility. Source Parent outperforms both controls in five of the six combinations. The Child-only comparison also shows Test quality gains from joint source-Parent optimization on 4DGS (Wu) Flame Steak and Painter (Appendix B). These results show that gradient-selected Parents provide useful starting points for both Child generation and joint adjustment of existing Gaussians with the new correction.

Table 3: **Starting points for correction and joint optimization targets.** The **best** PSNR (dB) is highlighted for each panel, Host, and scene; (b) also highlights the second-best. Final #Child is the final Child count after densification and pruning.

		Flame Steak		Painter		Basketball	
Input model	Selection strategy	PSNR↑	Final #Child (K)	PSNR↑	Final #Child (K)	PSNR↑	Final #Child (K)
(a) Child generation source							
4DGS (Wu)	Random source	32.8552	42.0	35.8114	157.2	27.4436	53.4
	Gradient source (Ours)	**32.9323**	57.4	**36.6098**	184.4	**27.6630**	117.1
Swift4D	Random source	33.1563	43.2	33.0959	66.2	27.5449	26.4
	Gradient source (Ours)	**33.6939**	19.5	**34.2337**	104.8	**27.7363**	72.2
(b) Existing Gaussians for joint optimization							
4DGS (Wu)	Random Non-parent	32.8474	49.9	36.3633	181.0	27.5709	90.7
	Matched Non-parent	32.8815	50.3	36.3309	176.7	27.4745	90.9
	Source Parent (Ours)	**32.9323**	57.4	**36.6098**	184.4	**27.6630**	117.1
Swift4D	Random Non-parent	33.1431	46.3	33.8358	89.7	**27.7873**	51.8
	Matched Non-parent	33.0597	46.4	33.7807	89.0	27.7287	53.1
	Source Parent (Ours)	**33.6939**	19.5	**34.2337**	104.8	27.7363	72.2

Reconstruction contributions of time-specific correction and joint Parent adjustment. Without retraining the parameters of the final model, we change only selected outputs to examine whether the two elements actually contribute to the rendering results. All other parameters and the Gaussian count remain unchanged.

For the Basketball example in Appendix B (Figure 6), replacing the time-specific Child position correction with its temporal mean over all frames increases region error; Ours has 44.7/70.4% lower error than this control for 4DGS (Wu)/Swift4D, respectively. Restoring only the Parents to the input Host state also increases error, with Ours achieving 16.7/6.9% lower error than the Parent-restored control, respectively. Additional rerenderings show that the effect of Parent state also depends on whether Children are included (Appendix B). This shows that the final reconstruction uses not only the time-specific Child corrections but also the Parent state that has adapted together with the Children.

Limitations. Under the specified additional training budget, the background of 4DGS (Wu) Boxes improves, but small dot patterns are still missing and the walls remain blurred (Appendix D). Furthermore, even with storage compression (Appendix E), the storage burden of the additional representation may grow for long sequences or large-scale scenes such as cities.

5 Conclusion

We have proposed targeted post-training refinement for trained dynamic Gaussian scenes, jointly designing time-specific correction capacity and the scope of joint adjustment of the existing representation. We add time-selective residual corrections to Children that reuse learned motion, and progressively expand the adjustment scope from joint optimization with their source Parents to all existing Gaussians. Across two deformation-based Hosts and three datasets, we improve average novel-view reconstruction quality over the input models and examine the roles of time-specific correction and joint Parent adjustment through ablation studies and rerendering diagnostics. These results show the importance of jointly designing additional correction capacity and co-adaptation of the existing representation for post-training quality improvement of learned scenes.

AI USE STATEMENT

In this work, we used generative AI tools for translation. Additionally, we used generative AI tools to assist in writing scripts for preparing qualitative comparison videos for the supplementary material and to clean up and package code for supplementary submission. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The methods and experimental setups are described in §3 and §4.1, respectively. The training code required for reproducibility is provided as supplementary material.

REFERENCES

Hans-Joachim Bungartz and Michael Griebel. Sparse grids. *Acta Numerica*, 13:147–269, May 2004. ISSN 1474-0508. doi: 10.1017/s0962492904000182.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4220–4230, June 2024.

Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic Studio: A Massively Multiview System for Social Motion Capture. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 3334–3342, December 2015.

Taeho Kang, Jaeyeon Park, Kyungjin Lee, and Youngki Lee. Clustered Error Correction with Grouped 4D Gaussian Splatting. In *SIGGRAPH Asia 2025 Conference Papers*, pp. 1–12, New York, NY, USA, December 2025. Association for Computing Machinery. doi: 10.1145/3757377.3763858.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

Seung-gyeom Kim, Areum Kim, Yongjae Yoo, and Sukmin Yun. Implicit 4D Gaussian Splatting for Fast Motion with Large Inter-Frame Displacements. In *The Fourteenth International Conference on Learning Representations*, 2026.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. In *The Third International Conference on Learning Representations*, 2015.

Sangwoon Kwak, Joonsoo Kim, Jun Young Jeong, Won-Sik Cheong, Jihyong Oh, and Munchurl Kim. MoDec-GS: Global-to-Local Motion Decomposition and Temporal Interval Adjustment for Compact Dynamic 3D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11338–11348, June 2025.

Junoh Lee, Changyeon Won, Hyunjun Jung, Inhwan Bae, and Hae-Gon Jeon. Fully Explicit Dynamic Gaussian Splatting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 5384–5409. Curran Associates, Inc., 2024. doi: 10.52202/079017-0174.

Tianye Li, Mira Slavcheva, Michael Zollhöfer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3D Video Synthesis From Multi-View Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5521–5531, June 2022.

Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8508–8520, June 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 (3DV)*, 2024.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising Densification in Gaussian Splatting. In *Computer Vision – ECCV 2024*, pp. 347–362. Springer Nature Switzerland, November 2024. ISBN 9783031730368. doi: 10.1007/978-3-031-73036-8_20.

Neus Sabater, Guillaume Boisson, Benoit Vandame, Paul Kerbiriou, Frederic Babon, Matthieu Hog, Remy Gendrot, Tristan Langlois, Olivier Bureller, Arno Schubert, and Valerie Allie. Dataset and Pipeline for Multi-View Light-Field Video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 30–40, July 2017.

Vikram Sandu, Mayurdeep Pathak, and Rajiv Soundararajan. Temporally Aware Densification for Dynamic 3D Gaussian Splatting. In *European Conference on Computer Vision (ECCV)*, 2026.

Meng Wang, Changqun Xia, Yuze Wang, Junyi Wang, Wantong Duan, Xinxiong Xie, and Yue Qi. Urban-GS: A Unified 3D Gaussian Splatting Framework for Compact and High-Fidelity Aerial-to-Street Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33207–33216, June 2026.

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, April 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20310–20320, June 2024.

Jiahao Wu, Rui Peng, Zhiyan Wang, Lu Xiao, Luyang Tang, Jinbo Yan, Kaiqiang Xiong, and Ronggang Wang. Swift4D: Adaptive divide-and-conquer Gaussian Splatting for compact and efficient reconstruction of dynamic scene. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jiawei Xu, Zexin Fan, Jian Yang, and Jin Xie. Grid4D: 4D Decomposed Hash Encoding for High-Fidelity Dynamic Gaussian Splatting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 123787–123811. Curran Associates, Inc., 2024. doi: 10.52202/079017-3934.

Haosen Yang, Chenhao Zhang, Wenqing Wang, Marco Volino, Adrian Hilton, Li Zhang, and Xiatian Zhu. Improving Gaussian Splatting with Localized Points Management. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21696–21705, June 2025.

Yixin Yang, Bojian Wu, Yang Zhou, and Hui Huang. Augmented Radiance Field: A General Framework for Enhanced Gaussian Splatting. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zongxin Ye, Wenyu Li, Sidun Liu, Peng Qiao, and Yong Dou. AbsGS: Recovering Fine Details in 3D Gaussian Splatting. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, pp. 1053–1061. ACM, October 2024. doi: 10.1145/3664647.3681361.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, June 2018.

Zheng Zhou, Yu-Jie Xiong, Jia-Chen Zhang, Chun-Ming Xia, Xihe Qiu, and Hongjian Zhan. Gradient-direction-aware density control for 3d gaussian splatting. In *The Fourteenth International Conference on Learning Representations*, 2026.

A Effect of Editing One Frame on Other Frames

We examine how much a position edit intended to correct one frame also changes other frames. Using the two degrees of freedom jointly optimized by Ours (Equation (7)), we compare a *shared-position edit*, which changes the reference position shared across times, with a *time-restricted edit*, which applies the same change only near the target frame. The key is to match the positions and rendered images at the target frame, then examine changes in the remaining frames. Both edits start from the same model obtained by jointly optimizing Parents and Children for 10,000 iterations on 4DGS (Wu) Flame Steak and Painter. Target frames at 1/4, 1/2, and 3/4 of each sequence are specified in advance, and all six cases are evaluated.

First, we obtain a position change that better fits a single target frame t_0. We select the 1,024 Children with the largest position-gradient magnitudes computed across all Train cameras at that frame, and adjust their reference query coordinates for up to 64 steps (fresh Adam, learning rate 3×10^{-5}). At each step, we measure the mean squared error on the target frame's Train images and retain the coordinate change from the step with the lowest error. For the *shared-position edit*, we apply this coordinate change to field queries at all times and measure how much positions and rendered images change at non-target frames. The Host and field weights, Gaussian set, and other attributes are fixed; only the changes in position outputs caused by adjusting the query coordinates are reflected in rendering. Only Train observations are used to select the target time, Children, and edit. Test observations are subsequently used to evaluate the fixed edit.

Next, for the *time-restricted edit*, we take the final position change $\Delta_{\mathrm{can}}(t_0)$ that the coordinate edit produces at the target frame and apply it only near that frame through the additional Child time correction. At M equally spaced knots including both endpoints, we set only the values on either side of the target time to 1 (the rest to 0), and use $b(t)$ expressed in the basis of Equation (8):

$$\Delta_{\mathrm{time}}(t) = b(t)\Delta_{\mathrm{can}}(t_0), \qquad b(t_0) = 1, \qquad \Delta_{\mathrm{time}}(t_0) = \Delta_{\mathrm{can}}(t_0). \tag{10}$$

The two edits move the same Gaussians to the same positions at the target frame and produce identical rendered images from all cameras. The temporal support of the time-restricted edit is constructed within the existing basis for this diagnosis; in actual Ours training, the reference positions and temporal correction coefficients are jointly optimized.

Table 4: **Changes in other frames after identically correcting the same frame.** All values are percentage changes (%) relative to the pre-edit error (SSE) of the corresponding observation set. Negative values indicate error reduction (improvement), and positive values indicate error increase (damage). The target frame aggregates all Train cameras; the remaining frames aggregate all Test observations except the target.

		Edited frame (Train)	Other frames (Test)	
Scene	Target frame	Same for both edits	Shared-position edit	Time-restricted edit
Flame Steak	75	-0.6070	$+0.0564$	$+0.0011$
	150	-0.4065	$+0.0255$	$+0.0012$
	225	-0.4313	$+0.0178$	$+0.0009$
Painter	12	-1.1000	-0.2611	-0.0364
	25	-1.0490	-0.1908	-0.0265
	37	-0.8881	-0.1946	-0.0253

For Flame Steak, frame 150 in Table 4, for example, both edits reduce Train error at that frame by 0.4065%. At other Test frames, the shared-position edit increases error by 0.0255%, while the time-restricted edit increases it by 0.0012%. Thus, the amount corrected at the target frame is identical, but the damage to other frames differs. In contrast, the shared-position edit also improves other frames on Painter. In all six cases, the rendered images at the target frame are exactly identical between the two edits, and the time-restricted edit affects only 6 frames. Positions and rendered images outside this support remain unchanged. Target Train error decreases in all cases, but target Test error increases for Flame Steak at target frames 150 and 225. These results show that the additional Child temporal representation can separate the amount of correction at the target frame from the temporal extent over which it is applied. Quality gains from actual training are evaluated through the ablation studies in the main text.

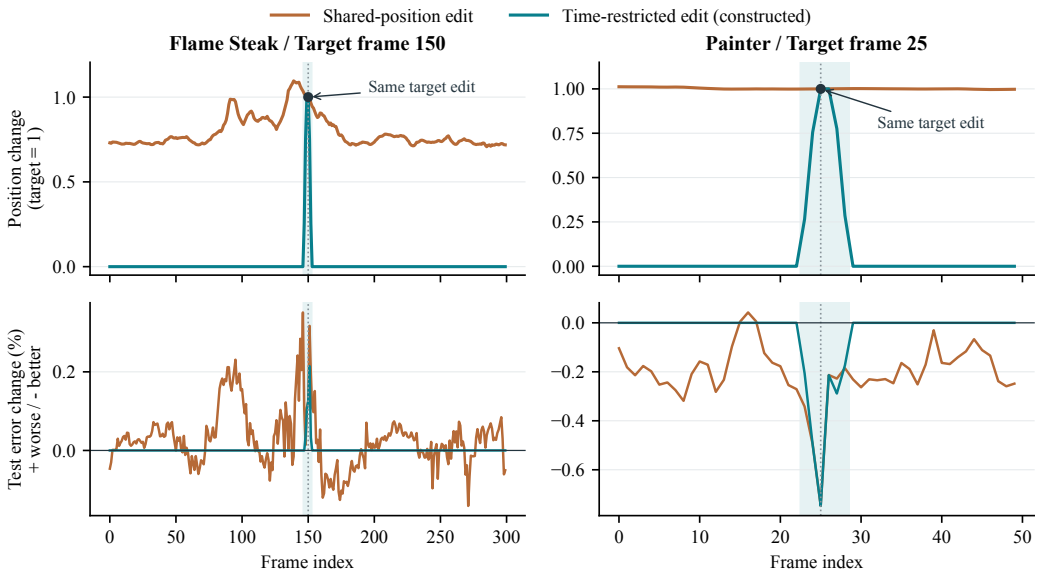


Figure 5: **Identical corrections at the target frame, followed by a comparison of changes at other frames.** Orange denotes the shared-position edit, and teal denotes the time-restricted edit constructed from the existing temporal basis. Dotted lines and markers indicate the matched target correction, while shading indicates the support of the time-restricted edit. The top row shows the mean magnitude of the position changes of the selected Children (normalized to 1 at the target frame); the bottom row shows the change in error relative to the pre-edit SSE at each Test frame. Positive values indicate degradation and negative values indicate improvement.

B Learned Corrections and Parent–Child Compositing

This section further examines how learned Child corrections and joint Parent adjustment contribute to the final reconstruction. In the Basketball example in Figure 6, replacing time-specific Child position corrections with their temporal means or restoring Parent outputs to the input Host state increases region error. We then examine the effect of initial joint Parent optimization and how the learned Parent state affects compositing with Children.

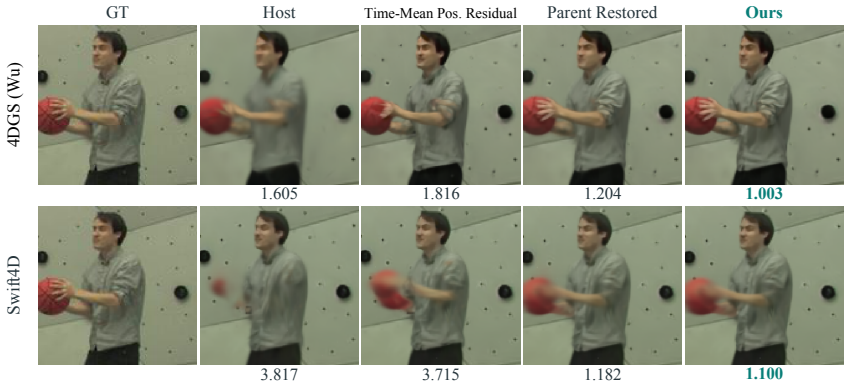


Figure 6: **Rerendering diagnosis of the learned correction.** Basketball, Test camera 15, frame 76. Values are region MSE $\times 10^3$. Time-Mean Pos. Residual replaces each Child's additional position correction with its mean vector over all frames, while Parent Restored restores only Parent outputs to the input Host state. All other parameters and the Gaussian count are retained, without retraining.

Effect of initial joint Parent optimization. On 4DGS (Wu) Flame Steak and Painter, we use the same input model and initial Children to compare a control that starts by optimizing only Children

while freezing all existing Gaussians with Ours, which starts by jointly optimizing their source Parents as well.

Table 5: **Effect of initial joint Parent optimization.** We compare 4DGS (Wu) models with different initial optimization targets on the full Test set. Improvement B and damage D are percentages (%) relative to Host SSE, as defined in Equation (14); Child counts are final counts including densification and pruning.

Scene	Initial targets	Test PSNR (dB)	Improvement B	Damage D	Child count
Flame Steak	Child-only	32.8652	14.03	9.62	50,762
	Parent+Child (Ours)	32.9323	20.56	15.00	57,389
Painter	Child-only	36.3102	49.41	11.10	174,845
	Parent+Child (Ours)	36.6098	52.80	10.39	184,408

In Table 5, Ours achieves greater net error reduction and higher Test quality in both scenes. On Flame Steak, the additional improvement exceeds the accompanying increase in damage; on Painter, improvement increases while damage decreases. This shows the benefit of jointly adjusting source Parents rather than starting refinement with Children alone.

Effect of Parent state on compositing with the same Children. Using Ours from Table 5, we retain the Children and restore only the Parents to their pre-refinement state to examine changes in image quality and Child compositing weights. We compare images rendered by the learned model as is with images in which only the original Parent outputs are replaced by outputs from the input model at the same time. We retain the Children, learned Non-parents, Gaussian count, field, cameras, and renderer, and evaluate all Test observations without additional training.

Because pixel colors are weighted sums of multiple Gaussian colors, changes in the positions and opacities of preceding Parents can change the compositing weights of the same Children. In Table 6, PSNR measures reconstruction quality, while D_A measures the *magnitude of the change in Child compositing weights*. $D_A = 0$ indicates no change in these weights.

Table 6: **Effect of restoring Parent outputs with identical Children.** We evaluate the same 4DGS (Wu) Ours models as in Table 5 on the full Test set.

	Test PSNR(dB)↑		Change in Child compositing weight
Scene	Ours	Parent restored	D_A
Flame Steak	32.9323	32.1587	0.0079
Painter	36.6098	33.7247	0.0344

On Painter, restoring Parents reduces PSNR from 36.6098 to 33.7247 dB and changes the Child compositing weight by an average of 0.0344.

At each pixel, A_C is the sum of the compositing weights of all Children. D_A is the absolute difference before and after Parent restoration, averaged first over pixels within each observation and then over all Test observations:

$$A_C^o(\mathbf{p}) = \sum_{j \in C} T_j^o(\mathbf{p}) \alpha_j^o(\mathbf{p}), \tag{11}$$

$$D_A = \frac{1}{|\mathcal{O}_{\text{test}}|} \sum_{o \in \mathcal{O}_{\text{test}}} \frac{1}{|\Omega_o|} \sum_{\mathbf{p} \in \Omega_o} \left| A_{C,\text{Ours}}^o(\mathbf{p}) - A_{C,\text{restore}}^o(\mathbf{p}) \right|, \tag{12}$$

Here, o denotes an observation, $\mathbf{p}$ a pixel, T_j^o the accumulated transmittance before the Child, α_j^o its per-pixel alpha, and Ω_o the full set of pixels in an observation. We measure A_C using group-specific color tagging while retaining all Gaussians, and verify agreement between the sum of weights and total alpha as well as reproduction of the reference Ours full Test PSNR. The correction effect of the same Children therefore also depends on the compositing state with their Parents. This supports the design of jointly optimizing Children and their source Parents to adjust how the new correction is reflected in the image.

Effect of Parent adjustment with and without Children. We examine whether the effect of Parent restoration depends on compositing with Children. For both Hosts on Painter and Basketball, we retain the learned Parent outputs or restore them to the input Host outputs, and rerender each case with Children included or excluded, yielding four conditions. Non-parents and the attributes of the remaining Gaussians are fixed, and no additional optimization is performed.

Table 7: **Rerendering with crossed Parent states and Child inclusion.** For Ours on Painter and Basketball, only Parent outputs are restored or Children are excluded from rendering. Values are full Test PSNR (dB).

		Painter (PSNR)		Basketball (PSNR)	
Parent output	Child	4DGS (Wu)	Swift4D	4DGS (Wu)	Swift4D
Learned state	Included	36.6098	34.2337	27.6630	27.7363
Restored to input Host	Included	33.7247	30.4248	27.0670	27.4236
Learned state	Excluded	24.3429	22.5638	25.0743	26.3256
Restored to input Host	Excluded	33.6295	30.7103	26.6943	27.1399

In Table 7, retaining the learned Parents is favorable in all four combinations when Children are included, whereas restoring Parents yields higher quality when Children are excluded. Thus, the final model uses a compositing state adapted together with the Children, rather than only the individual changes in Parents.

For the same observation, let $I_{P,C}, I_{P_0,C}$ denote the renders with learned and restored Parents, respectively, and let $I_{P,\emptyset}, I_{P_0,\emptyset}$ denote the corresponding renders without Children. The difference between these two effects of Parent restoration is

$$\Xi = (I_{P,C} - I_{P_0,C}) - (I_{P,\emptyset} - I_{P_0,\emptyset}). \tag{13}$$

Averaging $|\Xi|$ over the pixels and channels of each image and then over all Test observations gives 0.01675/0.02327 for 4DGS (Wu)/Swift4D on Painter and 0.00750/0.00318 on Basketball. This value directly measures how the image effect of Parent adjustment changes with the presence of Children. Its benefit to quality is evaluated jointly using the GT errors of the four renders above.

C Detailed Analysis of Improvement and Damage

We separate the quality differences in Section 4.3 into the effects of correcting more errors and causing less damage.

Let $e_H(p), e_M(p)$ be the mean squared RGB errors of the starting Host and the refined model at pixel p. Improvement B and damage D are defined as

$$B = \frac{100}{Z_H} \sum_p [e_H(p) - e_M(p)]_+, \qquad D = \frac{100}{Z_H} \sum_p [e_M(p) - e_H(p)]_+, \qquad Z_H = \sum_p e_H(p), \tag{14}$$

where $[x]_+ = \max(x, 0)$. Both values are percentages (%) with the common denominator of total Host SSE. The net error change is $D - B$. Figure 4(a) shows damage reduction as $D_{-/\checkmark} - D_{\checkmark/\checkmark}$, and (b) shows additional improvement as $B_{\checkmark/\checkmark} - B_{\checkmark/-}$. The first symbol indicates the initial P+C restriction; the second indicates time-specific (✓)/time-shared (–) correction. For each test camera and frame, we sort pixels by input Host error and divide them into three equal-sized low-, medium-, and high-error bins. Ties are resolved using a fixed pixel order, and the same bins are used for all conditions. Improvement and damage within each bin sum over its pixels while retaining the denominator of total Host SSE, so the sum across the three bins equals the full-image value. Differences between conditions are measured in percentage points (pp), and the bins are used only for post-training diagnosis.

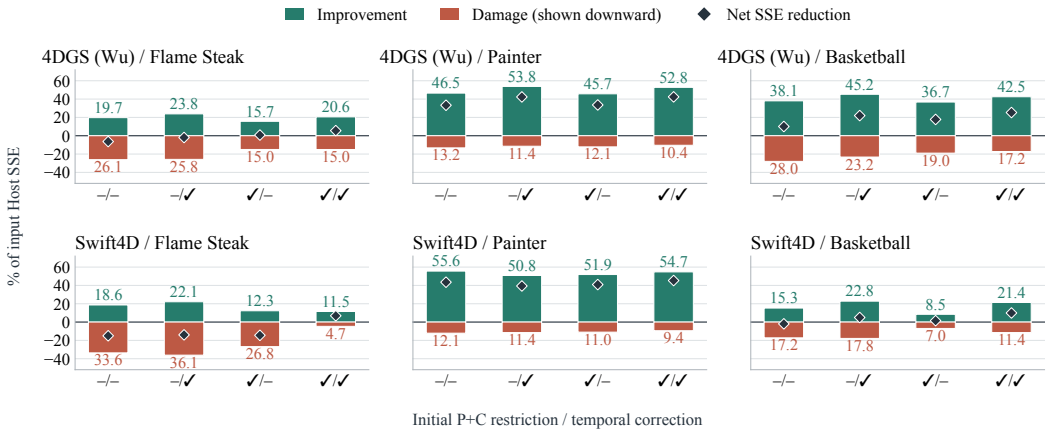


Figure 7: **Improvement/damage decomposition for all 24 conditions.** Full Test results for the same models as in Table 2. Green denotes B, downward red denotes D, and black markers denote $B - D$. The first symbol indicates the initial P+C restriction; the second indicates time-specific (✓)/time-shared (–) correction.

In the 4DGS (Wu) comparisons with the initial P+C restriction retained, changing time-shared correction to time-specific correction increases improvement on Flame Steak (15.72% → 20.56%), Painter (45.72% → 52.80%), and Basketball (36.75% → 42.52%). Damage is nearly unchanged on Flame Steak (14.98% → 15.00%) and decreases on the other scenes. Conversely, freezing Non-parents initially while retaining time-specific correction reduces improvement on 4DGS (Wu) Flame Steak from 23.83% to 20.56%, but reduces damage more substantially, from 25.77% to 15.00%. On Basketball as well, the reduction in damage (23.24% → 17.20%) exceeds the reduction in improvement (45.23% → 42.52%), favoring the initial restriction.

D Residual Reconstruction Errors

For 4DGS (Wu) Boxes in Figure 8, region quality improves over the Host, but small dot patterns are still missing and boundaries remain blurred in (a), while blotches and blurring remain on the wall in (b). These cases show background residual errors that are not fully removed within the specified additional training budget.

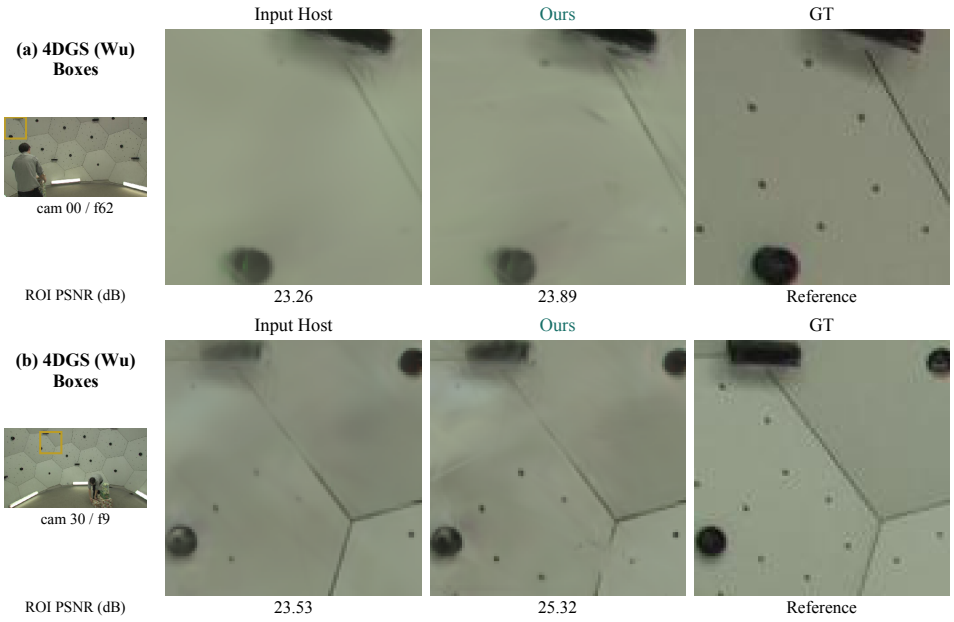


Figure 8: **Background errors remaining after improvement.** 4DGS (Wu) Boxes, Test (a) camera 00, frame 62 and (b) camera 30, frame 9. The same 96×96 region is compared within each case; values are region PSNR (dB).

E Storage Cost and Compression

This paper focuses on improving reconstruction quality, but post-training compression can mitigate the storage burden of the additional representation. We apply low-rank decomposition and quantization to Child coefficients after training, preserve sensitive coefficients at their original precision, and then apply lossless coding. Compression strength is selected using Train observations, while the Host and Gaussian count are retained. Across the six Host–dataset combinations in Table 8, the decrease in mean Test PSNR is at most 0.0171 dB, whereas additional Child storage is reduced by 43.2–90.5%. This shows that the storage cost of the correction representation can be substantially reduced with a small average quality loss.

Table 8: **Storage and quality before and after compression.** Dataset averages before and after compressing identical checkpoints; storage is in MB. Host includes the field and existing GS, while Total includes the Host, Children, and reconstruction metadata. ΔPSNR is the value after compression minus that before compression.

Model	Dataset	Host	Child		Total		ΔPSNR
			Before	After	Before	After	(dB)
4DGS (Wu)	N3DV	39.32	313.53	29.90	352.86	69.22	−0.0171
Swift4D	N3DV	142.50	90.61	28.84	233.12	171.34	−0.0008
4DGS (Wu)	Technicolor	212.12	189.96	107.91	402.08	320.03	−0.0021
Swift4D	Technicolor	207.89	51.72	14.31	259.61	222.19	−0.0076
4DGS (Wu)	Panoptic	54.67	190.96	35.24	245.64	89.91	−0.0082
Swift4D	Panoptic	106.05	89.78	27.50	195.84	133.56	−0.0029

On 4DGS (Wu) Birthday, Painter, Train, and Basketball, compression alone takes 3.08–5.42 seconds per scene and model reconstruction takes 0.87–1.90 seconds on an Intel Core i9-10900X (2 CPU threads).

F Additional Qualitative Comparisons

We present all comparison conditions from Table 1 for the six Host–scene combinations in Figure 3. We compare the input Host, the fixed count and density control conditions of both Continued and Frozen-field, Same-Child Joint, P+C-only, and Ours against GT. All conditions use the same test camera, frame, crop, and display scale; values are PSNR within the displayed region.

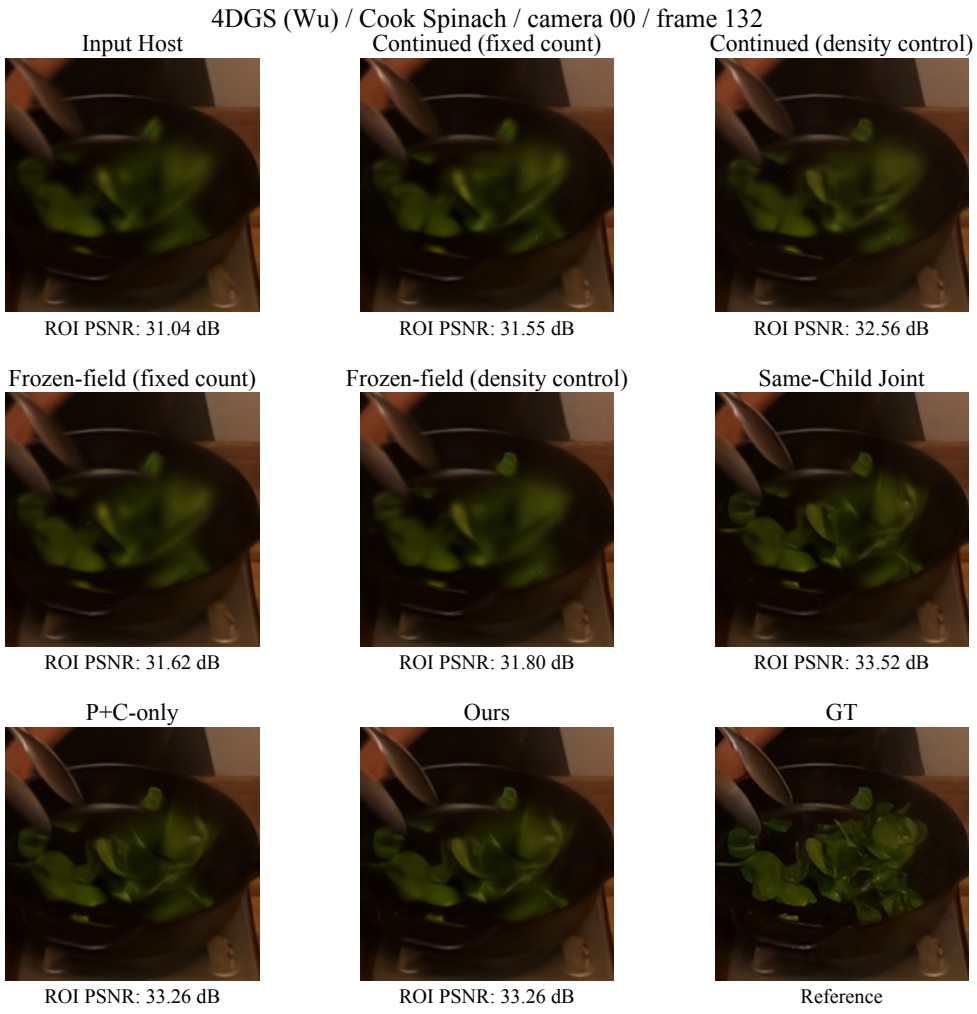


Figure 9: **4DGS (Wu), Cook Spinach.** The same spinach region at Test camera 00, frame 132.

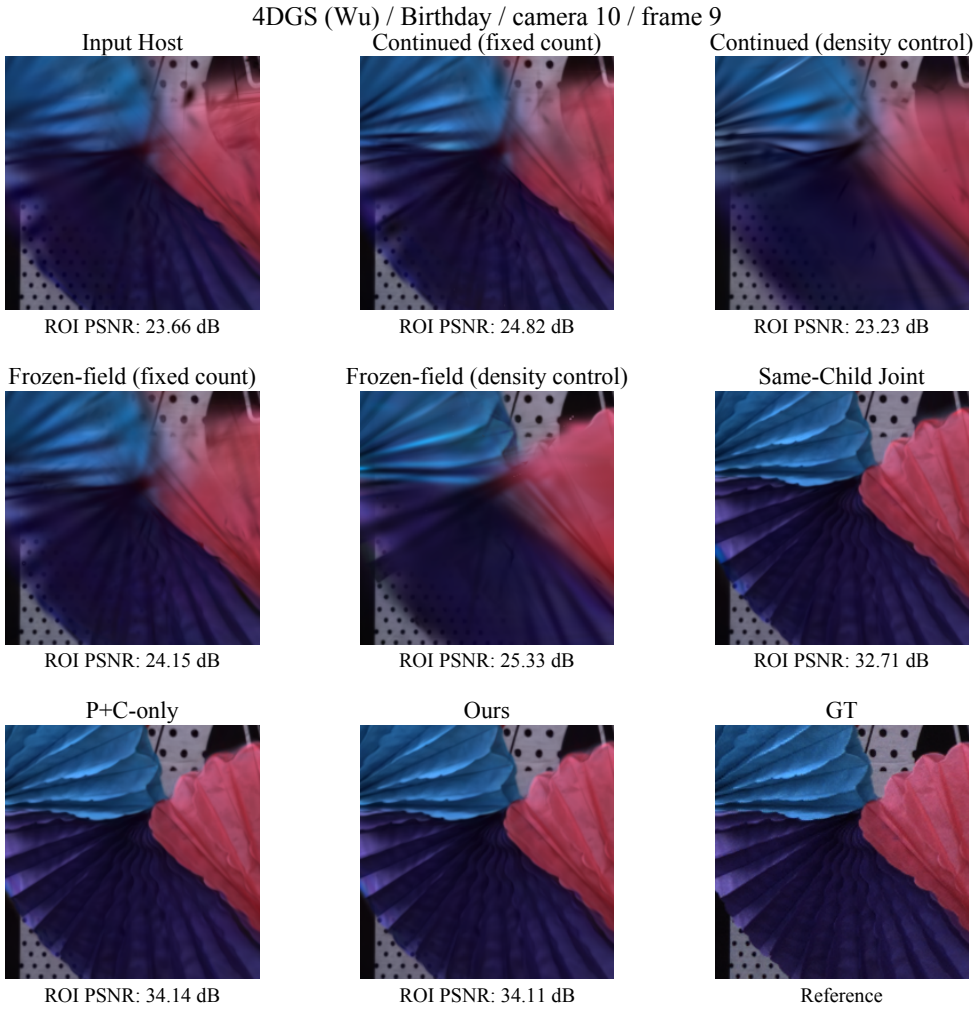


Figure 10: **4DGS (Wu), Birthday.** The same paper-decoration region at Test camera 10, frame 9.

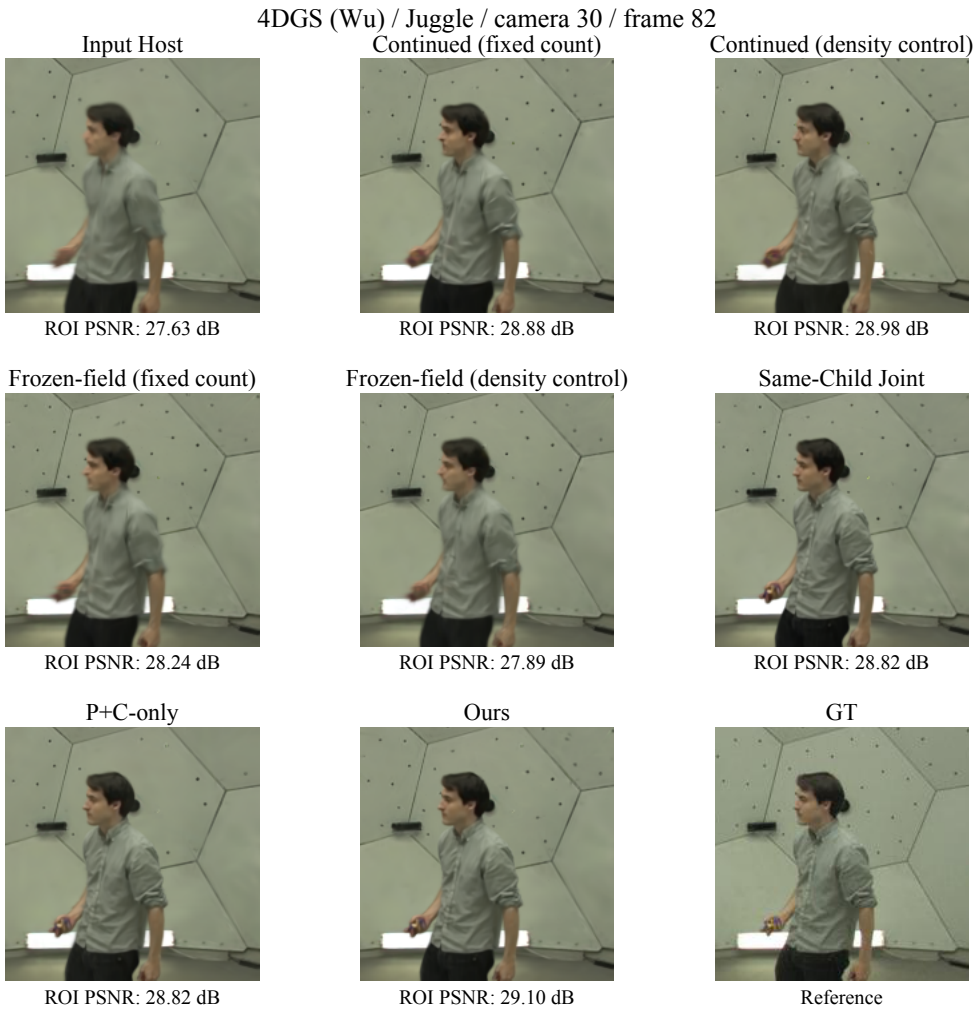


Figure 11: **4DGS (Wu), Juggle.** The same region containing the person at Test camera 30, frame 82.

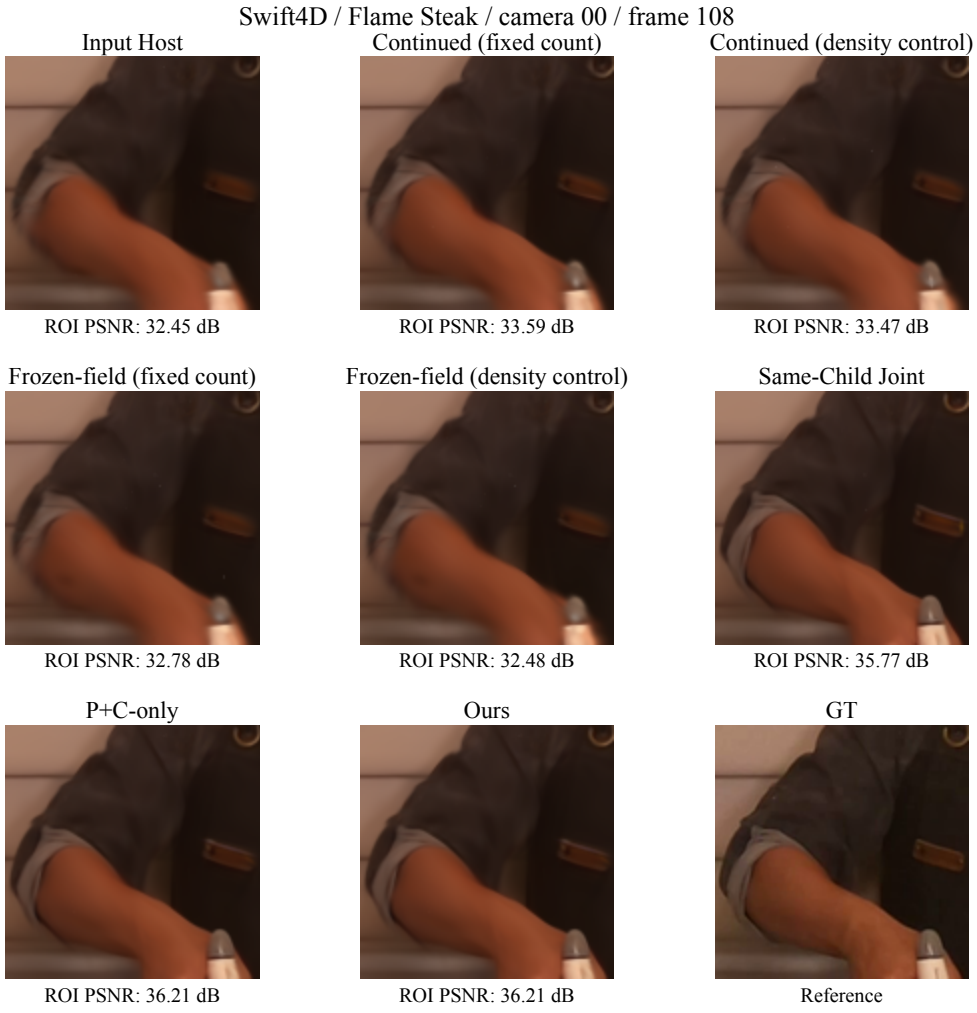


Figure 12: **Swift4D, Flame Steak.** The same folded-sleeve region at Test camera 00, frame 108.

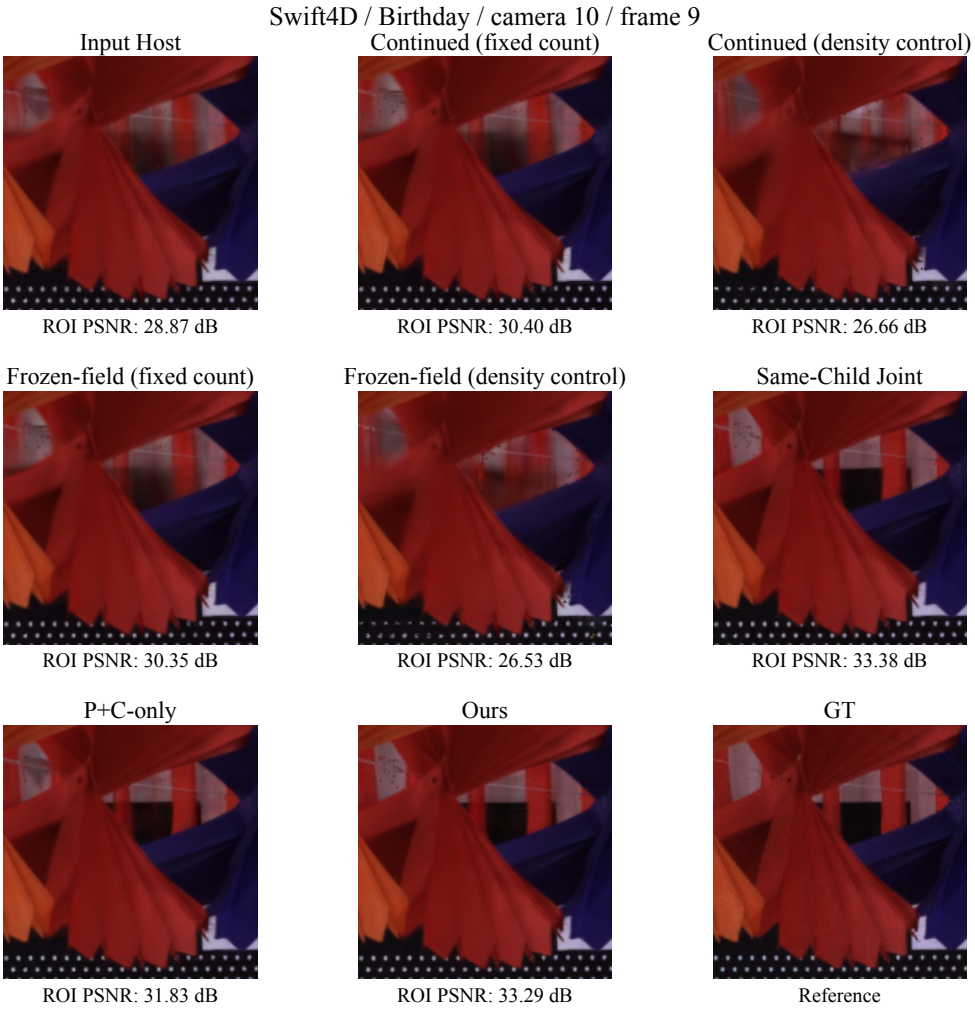


Figure 13: **Swift4D, Birthday.** The same paper-decoration region at Test camera 10, frame 9.

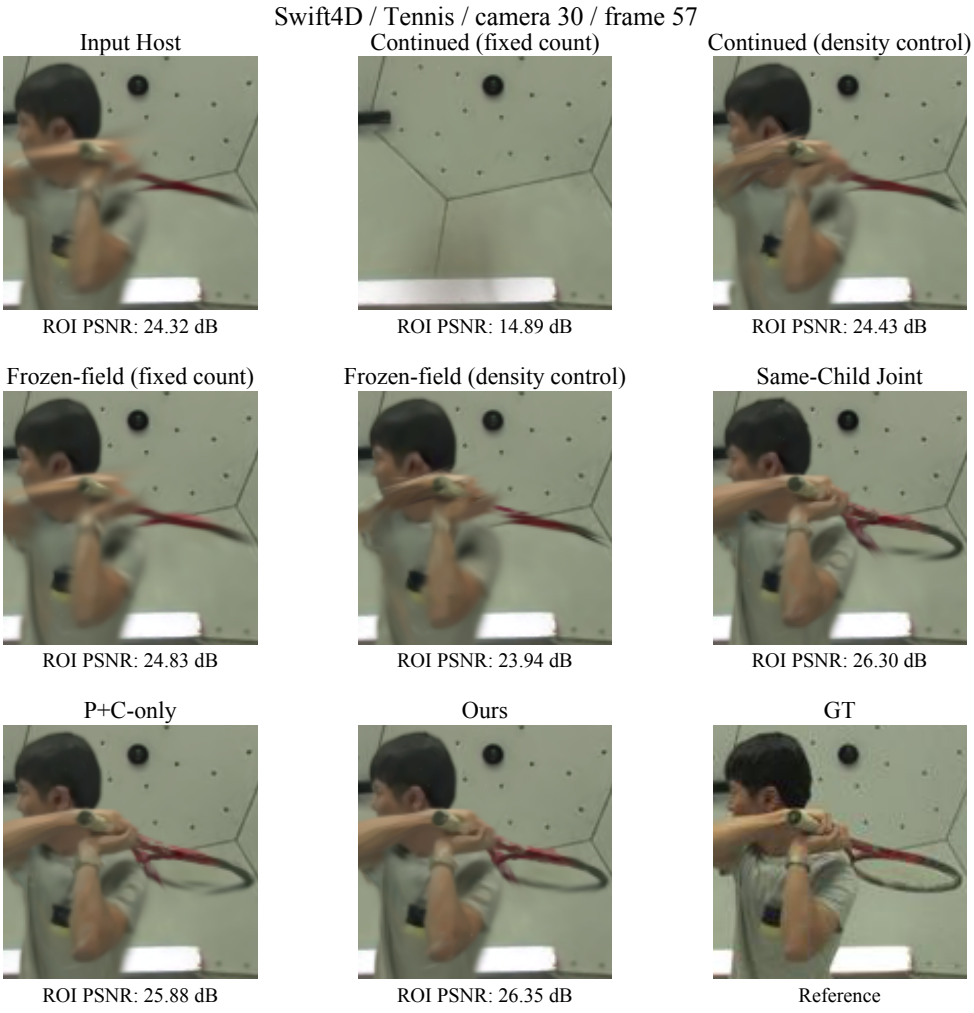


Figure 14: **Swift4D, Tennis.** The same racket region at Test camera 30, frame 57.