MAPo : Motion-Aware Partitioning of Deformable 3D Gaussian Splatting for High-Fidelity Dynamic Scene Reconstruction

Supplementary Material

1. Appendix Overview

This appendix provides comprehensive supplementary material to support and expand upon the methods and experiments presented in the main paper. Our goal is to offer full transparency and facilitate the reproducibility of our work. The appendix is organized as follows:

- **Extended Experiments and Analysis (Section 2):** We present a suite of additional experiments and in-depth analyses that rigorously validate our core contributions. This includes a study on the generalization of our partitioning strategy to different backbones, a deeper analysis of our core components' functions, and an exploration of our method's capabilities in modeling pure appearance variations and long video sequences.
- **Comparisons with Methods Under Specific Setups (Section 3):** We provide a transparent discussion and direct comparison with recent methods like SWinGS, DN-4DGS, and ST-GS, whose implementations are not publicly available or require specific experimental protocols that differ from our main evaluation setup.
- **More Quantitative and Qualitative Results (Section 4):** To further demonstrate the generalization of our method, we extend our evaluation to the **Technicolor** dataset. Besides, we offer exhaustive qualitative results to further demonstrate the visual fidelity of our approach.
- **Implementation and Experiment Details (Section 5):** We provide a complete breakdown of our implementation, including dataset processing, general experimental configurations, algorithmic specifics of our partitioning strategy, scene-specific hyperparameters, and the full loss function.
- **Discussions and Limitations (Section 6):** We conclude with a discussion on potential limitations of our current method and suggest promising directions for future work.

2. Extended Experiments and Analysis

2.1. Generalization to Different Deformation Backbones

Our partitioning strategy is designed to be highly adaptable and can be seamlessly integrated with other deformation-based approaches. To demonstrate this, we applied our method to the 4DGaussians [10] framework. As shown in Tab. 1, Tab. 2 and Fig. 1, this integration leads to significant performance gains.

Table 1. **Quantitative comparison of integrating MAPo with the 4DGaussians backbone on the N3DV dataset.** Quality metrics (PSNR, SSIM, LPIPS) are averaged across all scenes of the N3DV dataset, while computational overhead (Storage, Training Time, FPS) is measured on the *flame_salmon_frag1* scene.

Method	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
4DGaussians	30.19	0.917	0.061	53 MB	1 hour 13 mins	78.28
4DGaussians (80k)	30.51	0.919	0.059	55MB	6 hours 22 mins	75.09
Ous+4DGaussians	30.82	0.935	0.053	181MB	58 mins	79.79

Table 2. **Quantitative comparison of integrating MAPo with the 4DGaussians backbone on the MeetRoom dataset.** Quality metrics (PSNR, SSIM, LPIPS) are averaged across all scenes of the MeetRoom dataset, while computational overhead (Storage, Training Time, FPS) are measured on the *discussion* scene.

Method	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
4DGaussians	26.16	0.894	0.081	51 MB	1 hour 3 mins	77.26
4DGaussians (80k)	26.20	0.895	0.079	52MB	5 hours 20 mins	76.82
Ours+4DGaussians	26.43	0.899	0.074	175MB	51 mins	70.95

To ensure a fair comparison, we note that our integrated approach was trained for 80,000 iterations. Therefore, we benchmark against both the standard 4DGaussians configuration and an extended version trained for the same 80k iterations (denoted as 4DGaussians (80k)). The results reveal that while the standard 4DGaussians already achieves strong performance, extending its training yields only marginal improvements in visual quality (e.g., a big PSNR increase but negligible changes in SSIM and LPIPS) at the cost of a significantly longer training time. In contrast, our method comprehensively outperforms both versions of 4DGaussians on all evaluation metrics, with the sole exception of storage cost.

The increased storage is an expected consequence of the 4DGaussians deformation network, which is inherently larger due to its use of a HexPlane. However, we believe this can be managed by partitioning the HexPlane temporally during a split, rather than duplicating it, ensuring the cumulative size remains roughly constant.

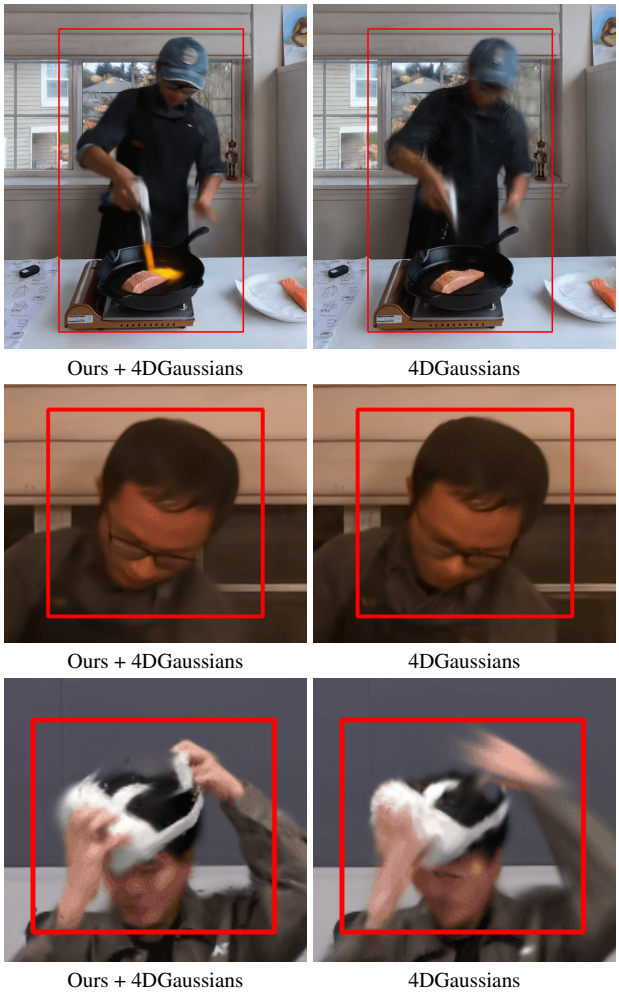


Figure 1. **Quanlitative results of our method on 4DGaussians.**

2.2. In-depth Analysis of Core Components

2.2.1. How Temporal Partitioning Mitigates Temporal Averaging

Our temporal partitioning significantly mitigates the temporal averaging effect inherent in methods relying on a single, unified model. By recursively partitioning high-dynamic 3DGs, we replace one global modeling task with multiple, simpler, localized ones. This decouples and specializes the spatio-temporal representation, allowing for a more faithful reconstruction of complex motions.

This effect is visualized in Fig. 2, where the canonical field becomes sharper due to the specialized modeling. To further validate this specialization at the parameter level, we analyze the pairwise cosine similarity between the weights of all deformation networks. The resulting heatmap in Fig. 3 reveals a clear hierarchical block structure: sibling networks handling adjacent time segments exhibit the highest similarity, whereas temporally distant networks show significantly lower similarity. This provides strong quantitative evidence that our partitioning fosters genuine model specialization adapted to local motion patterns, rather than simply duplicating parameters.

2.2.2. Visualizing Hierarchical Contributions

Our recursive partitioning results in a hierarchical structure where different levels of partition contribute to the final rendering. Fig. 4 breaks down a final rendered frame, visualizing the distinct contributions from 3DGs at each partition level (from level 0 to 3). This illustrates how our method composes the scene by combining the general representation from coarser levels with the specialized, fine-grained details captured by deeper levels.

2.2.3. Detailed Ablation of Dynamic Score Design

To validate our design of the dynamic score, we conduct a detailed ablation study on its key components and fusion methods, with results on the MeetRoom dataset shown in Tab. 3.

First, to demonstrate the effectiveness of using a dynamic score-based partitioning strategy itself, we compare against a baseline where 3DGs are partitioned **randomly** instead of being guided by our dynamic score. The significant performance drop of this random variant (Baseline (random)) confirms that an informed, score-based selection is crucial for allocating modeling capacity effectively.

Next, the data clearly indicates that combining both motion cues (displacement and variance) is essential. Relying on a single metric leads to a noticeable degradation in performance compared to our full approach. Qualitative results in Fig. 5 provide a visual counterpart to this analysis, showing that using either displacement-only or variance-only scores fails to capture the full motion complexity, resulting in artifacts or blurred details.

Finally, we compare different fusion strategies for the combined score. While all fusion methods yield comparable and strong results, the harmonic mean consistently achieves the best performance across all metrics, albeit by a small margin. We ultimately select the harmonic mean not only for its slightly superior empirical results but also for its intuitive theoretical property: it requires both scores to be high to produce a high final score, making it a robust criterion for identifying genuinely dynamic regions.

2.2.4. Qualitative Comparison with Segmental Training.

In addition to the quantitative results presented in the main paper, we provide a qualitative comparison against the segmental training baseline (Baseline (seg)) in Fig. 6. This baseline trains a separate model for each segment of the video, a common strategy for handling long dynamic sequences. As demonstrated in Fig. 6, our approach yields superior image quality.

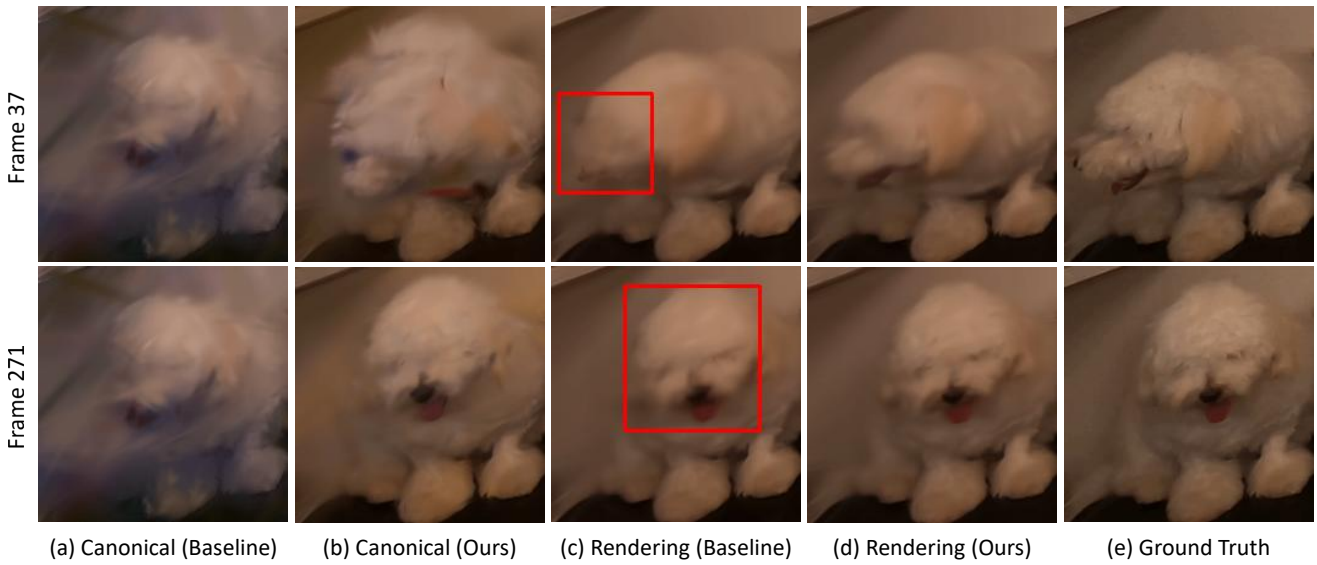


Figure 2. **Effect of our temporal partition strategy on the canonical field and final rendering.** We visualize results at two different timestamps (Frame 37 and 271). The baseline uses a single, shared canonical field (a) for all frames, resulting in a temporally averaged and blurry rendering (c) (e.g., the dog's face). Our method, by using specialized models per temporal segment, produces multiple canonical representations specific to that time segment (b), leading to a sharper and more accurate final rendering (d). "Canonical" refers to the rendering of the 3D Gaussians without applying deformation.

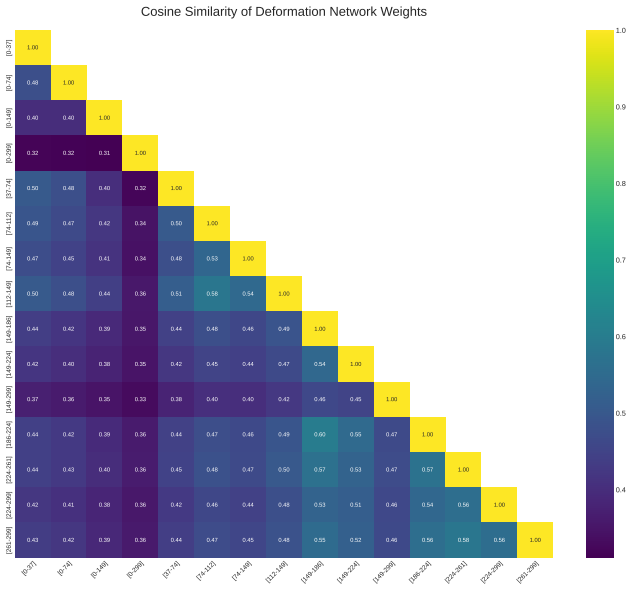


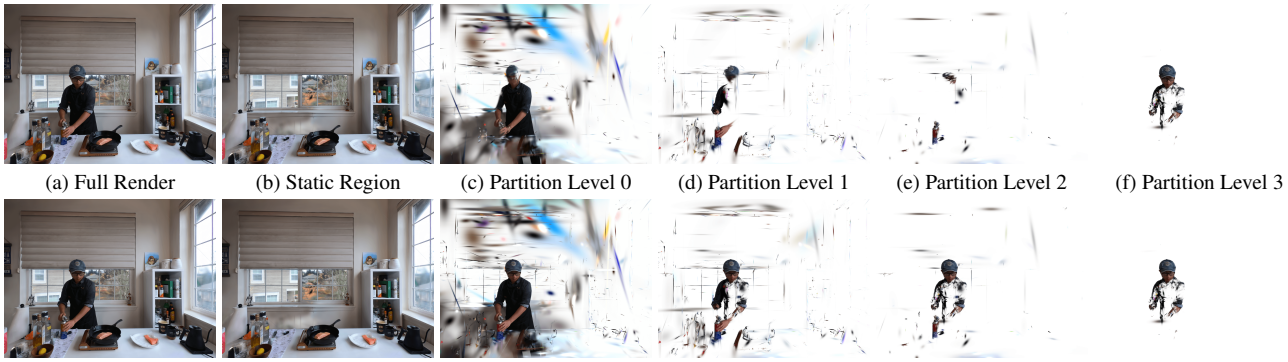
Figure 3. **Analysis of deformation network specialization.** Heatmap of pairwise cosine similarity between the weights of all deformation networks (root and sub-networks) after recursive partitioning on the *flame_salmon_frag3* scene. Our partitioning strategy yields specialized models adapted to different temporal segments.

Table 3. **Ablation on the dynamic score design.** We compare different components and fusion methods on the MeetRoom dataset.

Dynamic Score Variant	PSNR↑	SSIM↑	LPIPS↓
Baseline	26.24	0.896	0.081
Baseline (seg)	26.31	0.900	0.073
Baseline (random)	26.28	0.897	0.078
Only Max Displacement	26.52	0.901	0.070
Only Variance	26.41	0.899	0.072
Addition ($\tilde{r} + \tilde{v}$)	26.58	0.902	0.067
Multiplication ($\tilde{r} \times \tilde{v}$)	26.54	0.902	0.069
Harmonic Mean (Ours)	**26.63**	**0.903**	**0.067**

2.2.5. Sensitivity to Key Hyperparameters

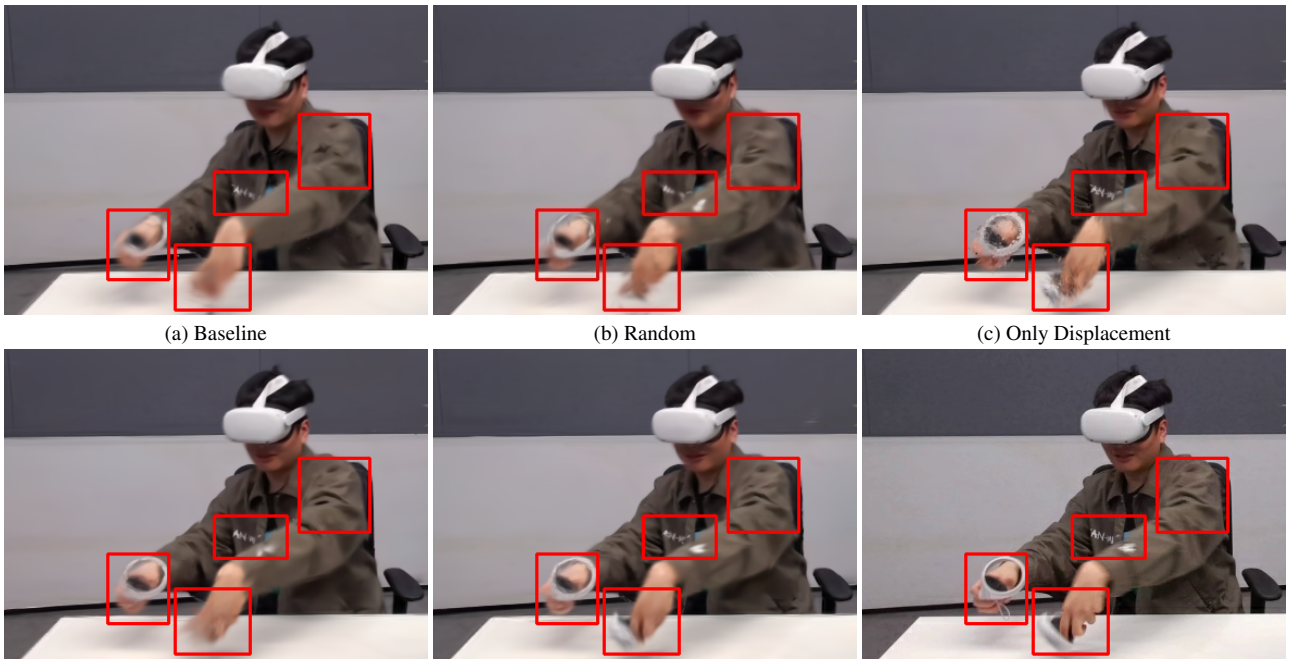
Our method is guided by several key hyperparameters that influence both the scene partitioning. We analyze our method's sensitivity to three crucial ones: the `dynamic_ratio`, which controls the initial seeding of high-dynamic 3DGs for temporal partitioning; the `static_ratio`, which determines the proportion of low-dynamic 3DGs to be treated as static; and the history length m, which defines the number of recorded historical positions for dynamic score estimation. To analyze the sensitivity to these parameters, we conduct an ablation study on the *flame_salmon_frag3* scene.



(a) Full Render (b) Static Region (c) Partition Level 0 (d) Partition Level 1 (e) Partition Level 2 (f) Partition Level 3

(g) Full Render (h) Static Region (i) Partition Level 0+ (j) Partition Level 1+ (k) Partition Level 2+ (l) Partition Level 3+

Figure 4. **Visualization of Partition Level and Static Region** on the *flame_salmon_frag4* sequence at frame 260. "Full Render" refers to the rendered result of our method at the current frame. "Static Region" refers to the rendered result of our method's static 3DGs. "Partition Level k" refers to the rendered result of all dynamic 3DGs at partition level k that participate in the rendering of the current frame. "Partition Level k+" refers to the rendered result of all dynamic 3DGs at partition levels greater than or equal to k that participate in the rendering of the current frame.



(a) Baseline (b) Random (c) Only Displacement

(d) Only Variance (e) Displacement + Variance (f) GT

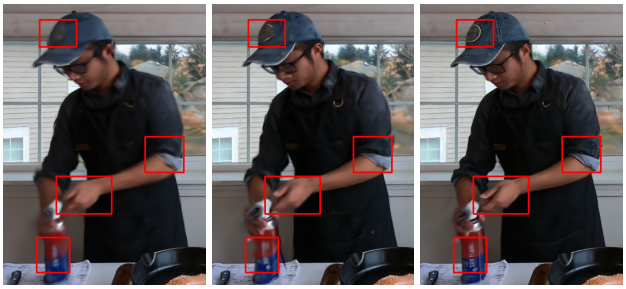
Figure 5. **Observation of dynamic partition on *Vrheadset*.**

Table 4. **Sensitivity analysis of `dynamic_ratio`.** We report results on *flame_salmon_frag3*.

`dynamic_ratio`	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
0.01	30.12	0.929	0.058	55 MB	1 hour 42 mins	86.55
0.05	30.26	0.933	0.054	59 MB	1 hour 48 mins	83.12
0.10	30.30	0.934	0.052	63 MB	1 hour 50 mins	79.64
0.20	30.32	0.935	0.052	82 MB	2 hours 16 mins	66.58
0.50	30.28	0.934	0.053	121 MB	2 hours 47 mins	50.19

Table 5. **Sensitivity analysis of `static_ratio`.** We report results on *flame_salmon_frag3*.

`static_ratio`	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
0 (None)	30.34	0.936	0.049	102 MB	2 hours 34 mins	48.32
0.50	30.27	0.933	0.053	80 MB	2 hours 19 mins	63.50
0.75	30.33	0.935	0.050	71 MB	2 hours 6 mins	73.81
0.95	30.30	0.934	0.052	63 MB	1 hours 50 mins	79.64
1.0	30.12	0.928	0.056	51 MB	1 hours 23 mins	90.12



(a) E-D3DGS (b) E-D3DGS(seg) (c) Ours

Figure 6. **Qualitative comparison on the *flame_salmon_frag3* in the N3DV dataset.**

Dynamic Ratio. As shown in Tab. 4, our method can achieve competitive results across a wide range of values, exhibiting considerable robustness to the initial `dynamic_ratio`. An extremely low ratio (e.g., 0.01) may lead to poor initial reconstruction by failing to adequately partition dynamic areas. However, moderately low values can be compensated for by the densification process, which adds new 3DGs in dynamic regions, ultimately achieving competitive results. Conversely, an excessively high ratio (e.g., 0.5) may create redundant 3DGs. We chose a value of 0.1 for our experiments, as it provides a balanced and efficient starting point that robustly converges to a high-quality result.

It is worth noting that selecting a suitable `dynamic_ratio` is an efficient process. Since the partitioning occurs early in training, one can set multiple ratio values and observe the initial partitioning results within a single run. This allows for the selection of an appropriate value in just a few minutes.

Static Ratio. With a well-chosen `dynamic_ratio`, we further investigate the impact of different `static_ratio` values on reconstruction performance. As presented in Tab. 5, under a good initial dynamic partitioning, the choice of `static_ratio` has a minor impact on the final reconstruction quality but significantly affects computational costs. As the ratio increases, rendering speed (FPS) and training efficiency improve substantially due to the reduced overhead from deformation computations. Concurrently, as more dynamic 3DGs are converted to static, their per-Gaussian embeddings are discarded, leading to a marked reduction in storage.

The Number of Recorded Positions m. We analyze the impact of the number of recorded historical positions, m, which is used to compute the dynamic score. As shown in Tab. 6, the choice of m affects the model's ability to infer motion intensity. A very small value ($m = 2$) is insufficient to capture enough historical context, leading to suboptimal reconstruction quality. Performance improves significantly when m is increased to 10, as more motion information becomes available.

Further increasing m from 10 to 300 yields only marginal improvements in quality, indicating that a relatively small number of historical points is already sufficient for a robust estimation of motion. We chose $m = 300$ in our experiments primarily to align with the sequence length of the dataset, ensuring that the dynamic score can capture motion intensity across the entire video duration. The results suggest that our method is not highly sensitive to this parameter, and a moderately reduced number of historical records would likely not significantly impact performance.

(a) Full Render (b) Static Region

Figure 7. **Observation of Static Partition** on More Scenes.

2.2.6. The Role of Static 3D Gaussian Partitioning

It is crucial to clarify that the primary objective of our static partitioning is **computational efficiency**, not semantic segmentation. By identifying 3DGs with negligible motion and exempting them from expensive deformation network computations, our approach drastically reduces computational load, as demonstrated in our main paper's ablation study. While its main goal is efficiency, our dynamic-score-based method still achieves a plausible separation of static scene components, as qualitatively shown in Fig. 7.

2.3. Modeling of Pure Appearance Variations

While our dynamic score is designed based on historical positions, our experiments reveal that the framework can still effectively model dynamic regions dominated by appearance changes, such as cast shadows. As shown in Fig. 8, our

Table 6. **Sensitivity analysis on the number of historical positions recorded m.** We report results on *flame_salmon_frag3*.

m	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
2	30.12	0.926	0.057	53 MB	1 hours 37 mins	85.74
10	30.20	0.929	0.056	60 MB	1 hours 45 mins	80.87
100	30.32	0.933	0.054	65 MB	1 hours 58 min	75.91
300	30.30	0.934	0.052	63 MB	1 hours 50 mins	79.64

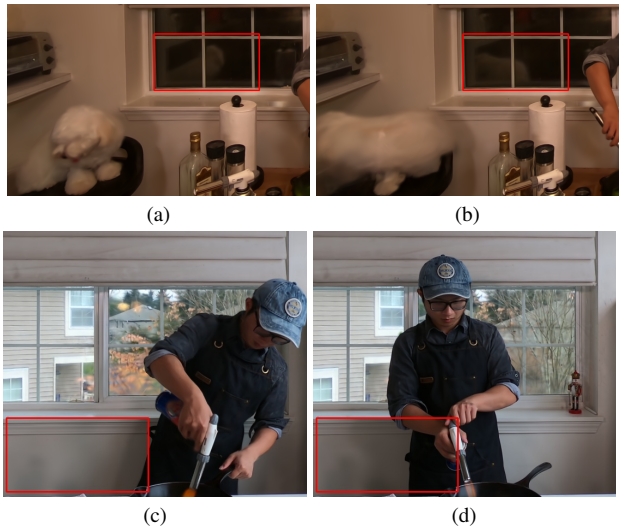


Figure 8. **Modeling of pure appearance variations in scenes.** Our method effectively captures appearance changes even when geometric motion is negligible. **Top Row:** In the *cook_spinach* scene, our model successfully reconstructs the time-varying reflections on the window surface between frame 0 (a) and frame 300 (b). **Bottom Row:** In the *flame_salmon_frag3* scene, it accurately captures the moving cast shadow on the wall, again between frame 0 (c) and frame 155 (d).

method successfully reconstructs a person's shadow moving across a wall. This capability arises from the dual nature of fitting appearance changes in deformable Gaussian Splatting: the optimization can either (1) directly modify the color (SH coefficients) of 3DGs at the corresponding location, or (2) move existing 3DGs from distant locations that possess the target color to fit the appearance change. The latter approach is a common phenomenon, often considered an ambiguity that many human reconstruction methods try to suppress by fixing positions before optimizing appearance.

However, our method is highly compatible with this second behavior. 3DGs that are moved over long distances to model appearance changes—such as shadows—are treated as highly dynamic objects by our position-based dynamic score. Consequently, these 3DGs are correctly identified and assigned to specialized temporal partitions, allowing them to be modeled with high fidelity. This demonstrates that our dynamic score, while based on geometry, can effectively detect and enable the modeling of significant appearance variations in the optimization process.

2.4. Performance on Long Video Sequences

Table 7. **Quantitative comparison on the full 1200-frame *flame_salmon* sequence.** Our method maintains high fidelity on long-duration videos compared to baselines.

Method	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
4DGaussians	28.37	0.904	0.090	53 MB	2 hours 1 min	76.15
Swift4D	28.21	0.910	0.076	113 MB	1 hour 14 mins	97.27
E-D3DGS	29.40	0.915	0.072	83 MB	4 hours 50 mins	37.69
Ours	29.82	0.924	0.059	113 MB	3 housr 2 mins	55.63

To evaluate the scalability of our method on long video sequences, we conducted an experiment on the complete *flame_salmon* sequence from the N3DV dataset. This sequence is composed of four consecutive fragments, totaling 40 seconds (1200 frames). We compare our full MAPo model against other baselines.

As shown in Tab. 7 and qualitatively visualized in Fig. 9, our method demonstrates a significant advantage in handling long-term dynamic scenes. While the performance of the baselines degrades noticeably when tasked with modeling the full 40-second sequence, our approach maintains high fidelity.

3. Comparisons with Methods Under Specific Setups

In our main paper, we excluded SWinGS [8], DN-4DGS [6], and ST-GS [5] from our primary comparison tables to maintain a unified and fair experimental protocol. These methods present specific challenges for a direct, holistic comparison:

- **SWinGS** [8] and **DN-4DGS** [6] only report quantitative results on a subset of the N3DV scenes we evaluate, and their implementations are not publicly available, preventing a comprehensive and reproducible comparison.
- **ST-GS**[5] employs a significantly different experimental setup, including: (1) partitioning long sequences into shorter clips for independent training, (2) loading image data directly to GPU memory, which affects training cost measurement, and (3) initializing from a dense, multi-frame SfM point cloud. These protocol discrepancies prevent a direct apples-to-apples comparison.

Despite these challenges, to provide as broad a context as possible, we present a comparison against their reported results in Tab. 8. MAPo demonstrates highly competitive performance. Compared to SWinGS's coarse, window-level partitioning based on 2D optical flow, our fine-grained, per-3DG partitioning achieves significantly higher rendering quality by adaptively allocating resources precisely where needed. Moreover, even against ST-GS, which leverages a

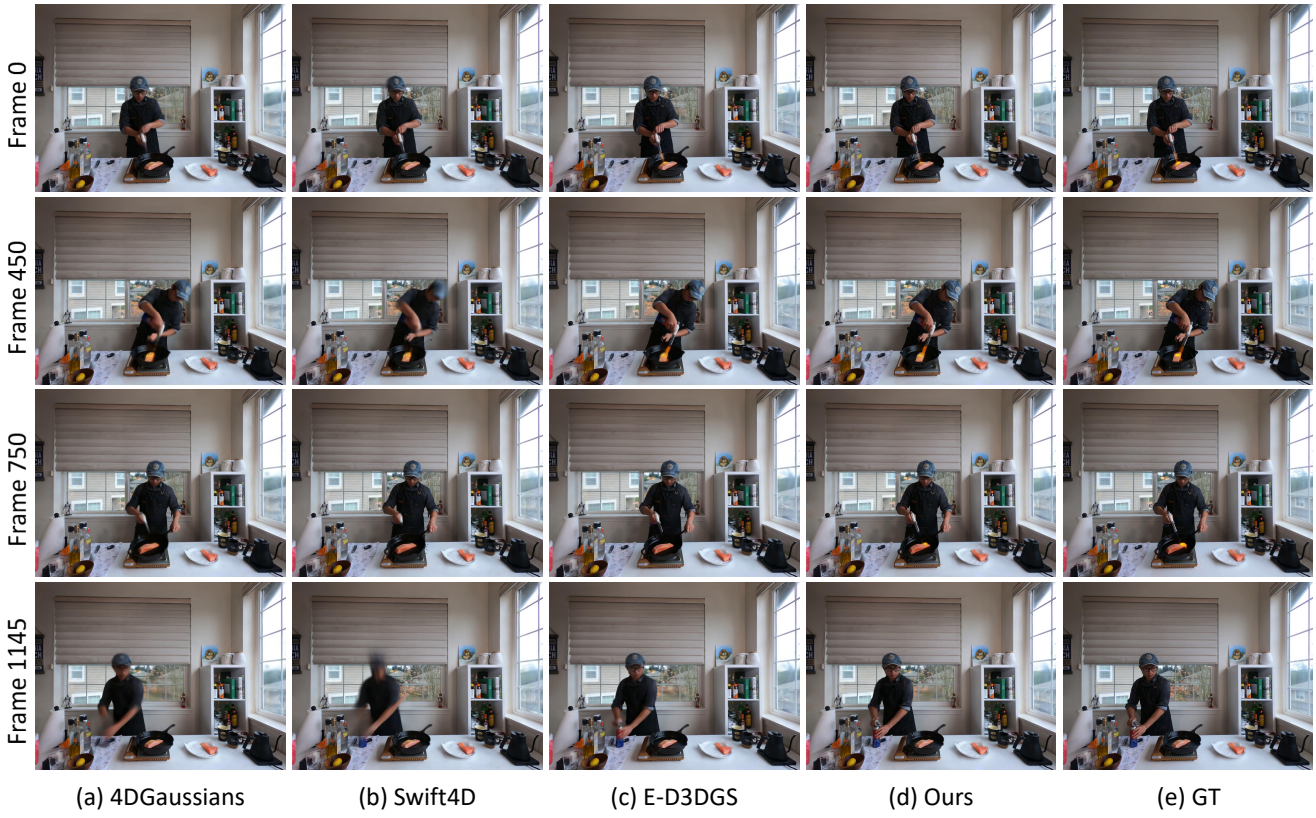


Figure 9. **Qualitative comparison on the long-duration (1200-frame) *flame_salmon* sequence.**

Table 8. **Quantitative comparison** on the N3DV dataset. This comparison excludes the *flame_salmon_frag2*, *flame_salmon_frag3*, and *flame_salmon_frag4* sequences.

Method	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
SWinGS	31.10	0.940	0.096	-	-	71.51
DN-4DGS	32.02	0.944	0.043	112 MB	50 mins	15.00
ST-GS	32.05	0.948	0.044	200MB	4 hours 30 mins	140.00
Ours	31.93	0.958	0.040	65 MB	1 hour 52 mins	75.64

more complex setup, our method remains highly competitive. Our advantages are also pronounced when compared to DN-4DGS; while achieving comparable PSNR, MAPo is superior on perceptual metrics (SSIM/LPIPS) and is far more efficient, requiring nearly half the storage and achieving 5x faster rendering speeds.

4. More Quantitative and Qualitative Results

4.1. Technicolor Dataset

To further validate the generalization and effectiveness of our method, we conduct additional experiments on the public Technicolor dataset [7]. This dataset is captured by 16 fixed cameras and is characterized by scenes with relatively simple motions but complex, fine-grained details. Following standard protocol, we evaluate on 50 frames from five commonly used scenes (Birthday, Fabien, Painter, Theater, Train) and use the camera in the second row and second column for testing.

Table 9. **Quantitative comparison on the Technicolor dataset.** Storage, training time, and FPS are measured on *Painter*.

Method	PSNR↑	SSIM↑	LPIPS↓	Storage↓	Training Time↓	FPS↑
DyNeRF[12]	31.80	-	0.140	0.6MB	-	0.02
HyperReel[12]	32.32	0.899	0.118	289 MB	2 hours 45 mins	0.91
4DGS	29.54	0.873	0.109	-	-	-
4DGaussians	29.62	0.844	0.176	85 MB	27 mins	49.81
Ex4DGS	33.56	0.917	0.086	165 MB	1 hours 7 mins	98.09
E-D3DGS	33.08	0.902	0.110	55 MB	2 hours 35 mins	71.72
Ours	33.69	0.923	0.078	49 MB	1 hour 54 mins	77.15

Quantitative Results. The quantitative comparison against several state-of-the-art methods is presented in Tab. 9. The results demonstrate that our method, MAPo, achieves exceptional performance. It not only secures the top rank in all primary quality metrics (PSNR, SSIM, and LPIPS), significantly outperforming strong baselines like Ex4DGS and E-D3DGS, but also maintains a highly competitive balance of storage, training time, and rendering speed. Notably, our method surpasses others while using

less storage than most competing approaches.

Qualitative Results. Qualitative comparisons, which will be provided in Fig. 10, further highlight the superiority of our approach. In challenging scenes like *Painter*, methods such as E-D3DGS and Ex4DGS exhibit significant drift due to the rapid hand motion, whereas our method reconstructs the fast-moving hand much better. Similarly, for fine details like the moving train in the *Train* scene, our method effectively reconstructs the sharp details of the text on the train's surface, while other approaches often produce noticeable blurring. These visual results corroborate our quantitative findings and underscore MAPo's capability in accurately reconstructing diverse dynamic scenes.

4.2. More Qualitative Results

We provide a rich set of additional visual comparisons across multiple scenes and baselines in Fig. 11 to further showcase the high fidelity of our reconstructions.

5. Implementation and Experiment Details

5.1. Dataset Processing and Experimental Configuration

Dataset Processing. For the N3DV [4] dataset, we adopt the data preprocessing from E-D3DGS [1]. For the Meet Room [3] dataset, we use the script from 4DGaussians [10]. For the Technicolor [7] dataset, we use the script from Ex4DGS [2]. A key aspect of our protocol for fair comparison is loading all image data into system memory (RAM) rather than GPU memory (VRAM).

General Configuration. For experiments on the N3DV, MeetRoom, and Technicolor datasets, we adopt the configuration from E-D3DGS for all common hyperparameters. The entire training is conducted for 80,000 iterations for each scene. Our method is implemented on top of the E-D3DGS codebase.

Baseline Experimental Setup. Our evaluation is against state-of-the-art NeRF-based and 3DGS-based methods. For 3DGS-based baselines, all methods are trained under identical conditions: same initial point cloud, camera poses, and our RAM-based data loading strategy. We use the original authors' recommended hyperparameters (iterations, learning rates) for each baseline to ensure their optimal performance. We report our own locally reproduced metrics for all methods to maintain internal consistency. For Swift4D, we include its time-consuming preprocessing step in its total training time.

5.2. Algorithmic Implementation Details

5.2.1. Dynamic Score Calculation

Starting from iteration 5,700, we record the historical positions of each 3DG into a FIFO queue (capacity: 300 entries). Before each partitioning operation, we compute the dynamic score s for each 3DG using these historical records.

5.2.2. Periodic Temporal Partitioning

This process runs from iteration 6,000 to 20,000, triggered every 2,000 iterations. A 3DG at partition level i with temporal range $[t_{\text{start}}, t_{\text{end}}]$ is split into two children at level $i+1$ covering sub-ranges $[t_{\text{start}}, t_{\text{mid}}]$ and $[t_{\text{mid}}, t_{\text{end}}]$.

- **Initial Pass (6,000 iter):** A portion of level-0 3DGs, determined by `dynamic_ratio`, are partitioned.
- **Subsequent Passes (8,000+ iter):** For levels 0-2, if it's the first time this level is being partitioned, we split the top 50% of 3DGs by dynamic score. For later partitions at the same level, we use a dynamic threshold (the minimum score from the initial pass at that level). Level 3 is the maximum and is not partitioned.

5.2.3. One-time Static 3D Gaussian Partitioning

This is a one-time process at iteration 8,001. A subset of level-0 3DGs with the lowest dynamic scores (proportion determined by `static_ratio`) are converted to static. Their canonical attributes are overwritten by their deformed attributes from the forward pass at that specific iteration's timestep. Their per-Gaussian embeddings are then discarded, and they are subsequently excluded from deformation network computations. The "randomness" of the timestep comes from the standard training data sampling at this specific iteration.

5.2.4. Scene-Specific Hyperparameters

The `dynamic_ratio` and `static_ratio` used are:

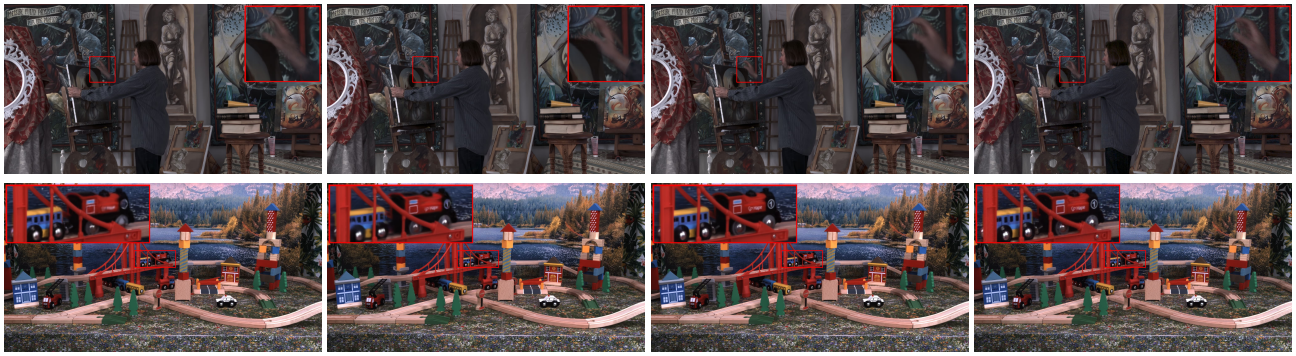
N3DV Dataset:
- For `coffee_martini`, `flame_salmon_frag1/2/3/4`: `dynamic_ratio` = 0.10, `static_ratio` = 0.95.
- For all other N3DV scenes: `dynamic_ratio` = 0.15, `static_ratio` = 0.9.

MeetRoom Dataset:
- For all scenes: `dynamic_ratio` = 0.12, `static_ratio` = 0.85.

Technicolor Dataset:
- For all scenes: `dynamic_ratio` = 0.2, `static_ratio` = 0.9.

5.3. Loss Function

Our total loss combines L1 loss, intermittent DSSIM loss (applied for 5 iterations every 50 iterations after 10K steps), a local smoothness regularization on dynamic 3DGs, and



(a) Ex4DGS (b) E-D3DGS (c) Ours (d) Ground Truth

Figure 10. **Qualitative comparison on the Technicolor dataset.**

our cross-frame consistency loss L_{cross}. The smoothness regularization is:

$$L_{\text{emb\_reg}} = \frac{1}{k|S|} \sum_{i \in S} \sum_{j \in \text{KNN}_{i;k}} (w_{i,j} \| z_{g_i} - z_{g_j} \|_2), \quad (1)$$

where $w_{i,j} = \exp(-\lambda_w \| \mu_j - \mu_i \|_2^2)$. All loss terms have a weight of 1.

6. Discussions and Limitations

Despite our method's strong robustness to hyperparameters for partitioning dynamic and static 3DGs, it still relies on two pre-defined thresholds: `dynamic_ratio` and `static_ratio`. As demonstrated by the sensitivity analysis in Sec . 2.2, the model can converge to high-quality results across a wide range of parameter values, which indicates the method's stability. However, the manual process of setting these parameters, to some extent, reduces the system's level of automation and user-friendliness. For a new scene, users may still need to spend a short amount of time determining the optimal values.

We believe a more ideal solution is to achieve adaptive parameter tuning by combining an analysis of the scene's intrinsic scale with the historical positions of the 3DGs. This way, the system would no longer rely on normalized, relative ratio thresholds, but could instead intelligently identify and partition 3DGs based on the absolute magnitude of their dynamic displacement. We will conduct an in-depth exploration of this parameter-free direction in the future.

Furthermore, our method is specifically designed for multi-view dynamic scene reconstruction, with the core objective of capturing the finest possible dynamic details under the constraints of sufficient viewpoint information. We argue that monocular dynamic scene reconstruction is inherently an ill-posed problem, whose success often relies more heavily on strong priors, such as the implicit smoothness introduced by hash grids or additional depth and geometric regularization losses.

A recent study by Stearns et al. [9] further highlights this challenge through "Marble": in ill-posed settings with insufficient viewpoint information, more expressive and flexible models can paradoxically yield worse results by overfitting to ambiguous observations. This stands in contrast to our design philosophy, which leverages multi-view data to precisely drive a high-degree-of-freedom dynamic representation. Therefore, given the fundamental difference between our method's prerequisites and the monocular setting, we did not evaluate our approach on monocular dynamic scene datasets.

References

[1] Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024. 8

[2] Junoh Lee, Chang-Yeon Won, Hyunjun Jung, Inhwan Bae, and Hae-Gon Jeon. Fully explicit dynamic gaussian splatting, 2024. 8

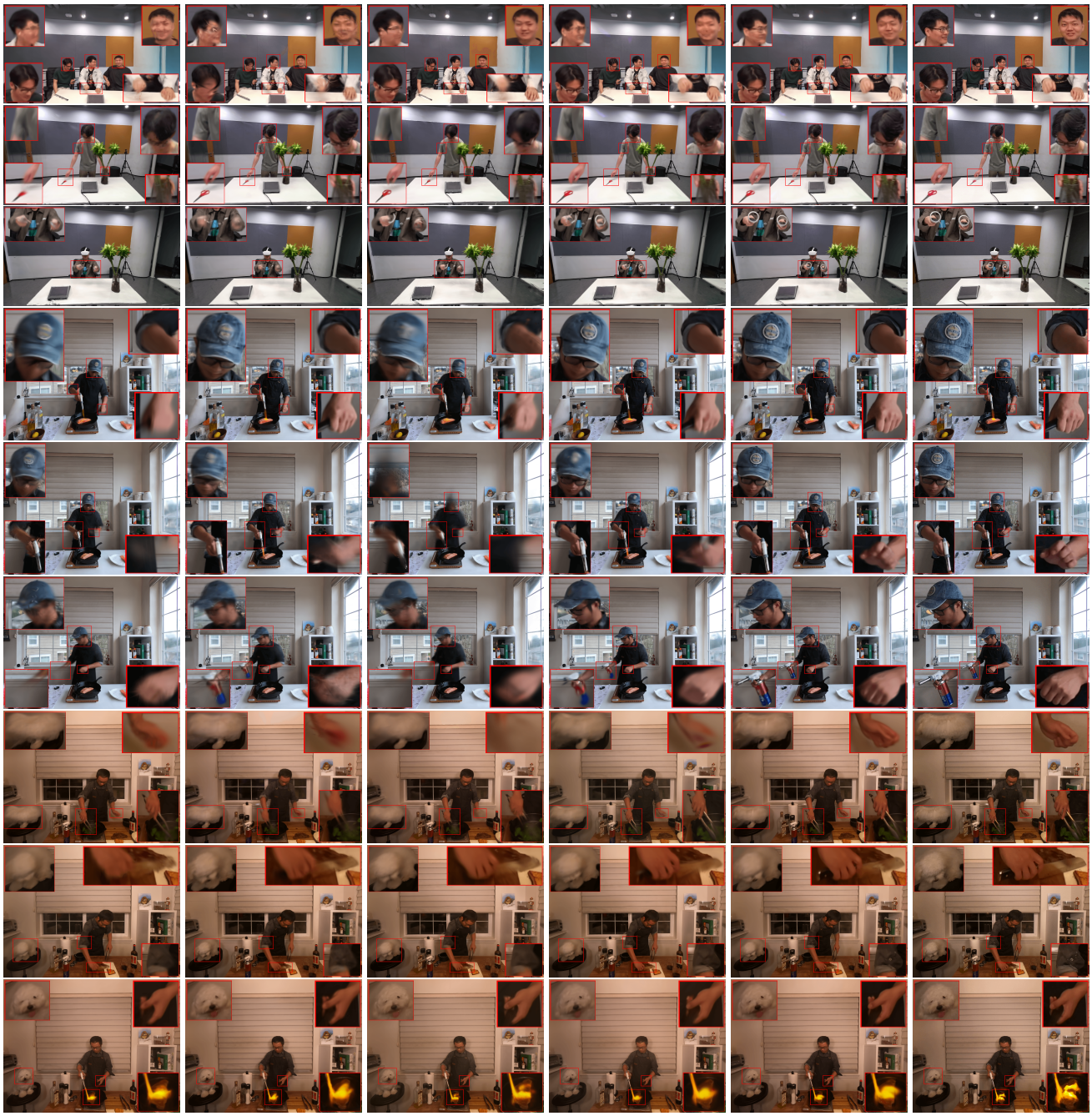
[3] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthesis. *Advances in Neural Information Processing Systems*, 35: 13485–13498, 2022. 8

[4] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 8

[5] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis, 2023. 6

[6] Jiahao Lu, Jiacheng Deng, Ruijie Zhu, Yanzhe Liang, Wenfei Yang, Tianzhu Zhang, and Xu Zhou. Dn-4dgs: Denoised deformable network with temporal-spatial aggregation for dynamic scene rendering, 2024. 6

[7] Neus Sabater, Guillaume Boisson, Benoit Vandame, Paul Kerbiriou, Frederic Babon, Matthieu Hog, Remy Gendrot, Tristan Langlois, Olivier Bureller, Arno Schubert, and Va-



(a) 4DGaussians (b) Ex4DGS (c) Swift4D (d) E-D3DGS (e) Ours (f) Ground Truth

Figure 11. **Qualitative comparisons** on the MeetRoom and N3DV dataset.

lerie Allié. Dataset and pipeline for multi-view light-field video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1743–1753, 2017. 7, 8

[8] Richard Shaw, Michal Nazarczuk, Jifei Song, Arthur Moreau, Sibi Catley-Chandar, Helisa Dhamo, and Eduardo Perez-Pellitero. Swings: Sliding windows for dynamic 3d gaussian splatting, 2024. 6

[9] Colton Stearns, Adam Harley, Mikaela Uy, Florian Dubost, Federico Tombari, Gordon Wetzstein, and Leonidas Guibas. Dynamic gaussian marbles for novel view synthesis of casual monocular videos. In *SIGGRAPH Asia 2024 Conference Papers*, page 1–11. ACM, 2024. 9

[10] 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, 2023. 1, 8