

Multi4D: High-Fidelity Dynamic Gaussian Splatting via Multi-Level Competitive Allocation

Rui Wang^{✉*}, Quentin Lohmeyer[✉], Siyu Tang[✉], and Mirko Meboldt[✉]

ETH Zürich, Zurich ZH 8092, Switzerland

{ruiwang46, qlohmeyer, meboldt}@ethz.ch siyu.tang@inf.ethz.ch
<https://batfacewayne.github.io/Multi4D.io/>

Abstract. Dynamic 3D Gaussian splatting faces a fundamental tension between motion consistency and visual fidelity. Deformation-based approaches preserve temporal correspondence but suffer from motion over-factorization, oversmoothing high-frequency dynamics. In contrast, 4D-primitive methods capture fine visual details yet incur temporal over-parameterization, breaking object identity and leading to severe storage overhead. To resolve this, we introduce Multi4D, a framework for high-fidelity dynamic Gaussian Splatting based on multi-level competitive allocation. Instead of a monolithic representation, we distribute modeling capacity across three structured levels: static structure, persistent dynamic geometry, and transient appearance primitives. Through shared rasterization and residual-driven optimization, these levels dynamically compete to explain photometric error, enabling adaptive specialization without pre-assigned decomposition. This allocation preserves long-term motion consistency while capturing fine dynamic detail, achieving state-of-the-art rendering quality and real-time performance with significantly fewer dynamic primitives. Furthermore, because our representation explicitly tracks compact persistent Gaussians over time, semantic features can be embedded afterward, enabling Multi4D to achieve state-of-the-art 4D segmentation accuracy with an order-of-magnitude speedup.

Keywords: Dynamic Gaussian Splatting · 4D Scene Reconstruction · Novel View Synthesis · 4D Segmentation

1 Introduction

3D Gaussian Splatting (3DGS) [10] has revolutionized novel view synthesis by combining explicit 3D primitives with real-time differentiable rasterization. However, extending this representation to *dynamic* scenes exposes a fundamental conflict between physical plausibility (tracking and correspondence) and visual fidelity (rendering quality). Current dynamic 3DGS frameworks generally fall into two opposing domains. The first, comprising *deformation-based* approaches, maintains a fixed set of canonical Gaussians and warps them over time via neural deformation networks [38], explicit trajectory modeling [7, 20], or feature

* Corresponding author.

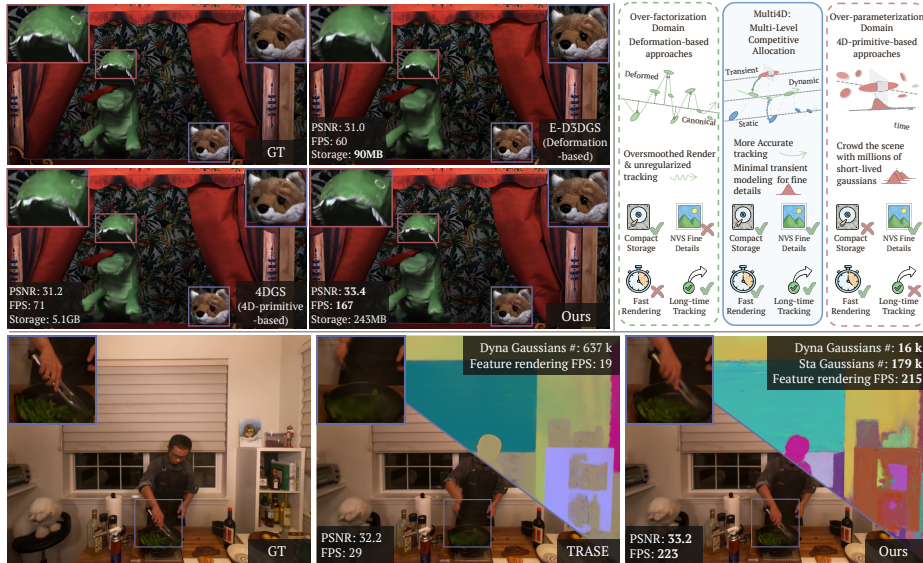


Fig. 1: Multi4D enables (1) high-quality, efficient dynamic scene reconstruction via competitive multi-level specialization, and (2) compact, high-accuracy 4D segmentation with fast inference.

grids [32, 38]. By enforcing a strict temporal correspondence, these methods inherently preserve Gaussian identity, making them suited for downstream tasks like semantic embedding and tracking [8, 18]. However, this formulation often leads to severe *motion over-factorization*, where the deformation field groups nearby motion and over-smooths high-frequency dynamics. While recent works attempt to alleviate this over-smoothing via coarse-to-fine deformation [12], frequency-aware fields [21], or spline-based trajectories [40], they remain fundamentally constrained by two major drawbacks: Because the representation is dominated by motion modeling, these methods often interpret complex appearance changes (e.g., specularities or lighting shifts) as physical motion, causing spurious geometric warping to minimize photometric error. Moreover, they face a non-trivial computational bottleneck: querying the deformation network for *every* primitive at *every* timestamp incurs an overhead that scales strictly with the total number of Gaussians. Consequently, increasing the primitive count to capture fine dynamic details directly compromises real-time rendering capacity and memory efficiency [29, 30].

Conversely, the second domain comprises *4D-primitive* [5, 37], which conceptualize scene dynamics as 4D spatiotemporal Gaussian hyper-cylinders. At any given timestamp, these primitives are sliced along the temporal axis to yield 3D Gaussians with time-varying opacities and linear local position shifts. By modulating temporal existence, these methods seamlessly capture complex appearance changes and transient geometry. Furthermore, this time-dependent formulation

enables temporal pre-filtering to render only active Gaussians, accelerating rendering speed. However, while effectively modeling the dynamic scene with fine details by piece-wise linear approximation, these methods lead to severe *temporal over-parameterization*. They are prone to modeling physical motion as transient appearance changes, where the optimization tends to exploit temporal scaling to minimize photometric error, hallucinating millions of primitives that exist for extremely short lifespans rather than capturing physical movement, leading to broken geometry in the fast motion region. Most recent methods [15, 41, 42] attempt to reduce the number of Gaussian primitives via temporal score pruning or sensitivity analysis, primarily for static regions to alleviate the storage overhead at the cost of slightly reducing rendering quality. However, these approaches are generally limited by the local piece-wise motion approximation assumption and cannot further efficiently simplify dynamic regions. Moreover, lacking holistic modeling and canonical geometric constraints limits these methods to only dense camera input, and generalizes poorly to sparse camera inputs and monocular settings, compared to deformation-based approaches.

We therefore challenge the monolithic assumption that one representation must explain both physical kinematics and transient appearance. Instead, we formulate dynamic reconstruction as a *competitive multi-level optimization problem*, in which models with distinct inductive biases dynamically compete to explain photometric residuals. By explicitly separating statics from dynamics, and geometric deformation from transient appearance, we enable residual-driven allocation across levels under a unified differentiable renderer. To this end, we introduce **Multi4D**, a competitive multi-level framework that jointly optimizes three functionally specialized Gaussian subsets: (1) **Static Gaussians** ($\mathcal{G}_s$), which provide a time-invariant structural backbone; (2) **Persistent Dynamic Gaussians** ($\mathcal{G}_d$), deformable primitives governed by a holistic deformation module to maintain long-term identity and trackability; and (3) **Transient Gaussians** ($\mathcal{G}_t$), short-lived 4D primitives dedicated exclusively to modeling high-frequency appearance residuals. A high-level overview of our proposed method is shown in Fig. 1.

Through shared rasterization, gradients are coupled across subsets: once stable geometry explains a region, residual-driven allocation suppresses redundant modeling in other subsets. This structural decoupling prevents deformation fields from absorbing transient photometric noise while allowing high-capacity 4D primitives to model irreducible appearance variation. As a result, Multi4D preserves long-term motion consistency and fine-grained dynamic detail simultaneously. By explicitly separating structural persistence from transient modeling, dynamic novel view synthesis benefits from coherent geometry with significantly fewer modeled primitives, yielding higher fidelity, faster rendering, and improved generalization under sparse or monocular supervision. For 4D segmentation, we restrict optimization to the persistent subset $\mathcal{G}_s \cup \mathcal{G}_d$, which maintains stable temporal identity while discarding transient appearance noise. This produces compact feature rendering, stronger motion consistency, and state-of-the-art

tracking accuracy with an order-of-magnitude speedup. **Our contributions are summarized as follows:**

- We propose **Multi4D**, a multi-level Gaussian decomposition framework that separates static structure, persistent dynamic geometry, and transient appearance modeling, reconciling motion consistency with photometric fidelity.
- We introduce a bottom-up, self-regularized training strategy with velocity-aware lifting and mask-aware pruning, enabling structured specialization and compact representations.
- We demonstrate that Multi4D achieves state-of-the-art dynamic novel view synthesis, delivering superior rendering fidelity with significantly fewer dynamic primitives and improved runtime performance.
- We further show that Multi4D naturally supports state-of-the-art 4D segmentation with significantly faster inference.

2 Related Work

Deformation-based dynamic Gaussian methods track a canonical set of 3D Gaussians using time-conditioned deformation modules, including explicit trajectories [7, 20], neural networks [38], and grids [3, 32]. Extensions introduce finer deformation models such as multi-grid refinement, spline trajectories, and frequency-aware networks [6, 21, 35, 40]. However, per-frame querying deformation fields for every primitive incurs significant computational overhead and limits modeling capacity. Efficiency-oriented variants decouple static and dynamic [30, 33], yet complex time-varying appearance modeling remains challenging.

4D-primitive-based methods represent dynamic scenes using spatiotemporal Gaussians sliced at each timestamp [5, 37]. Although capable of high-fidelity rendering, this formulation often leads to temporal over-parameterization, spawning large numbers of short-lived primitives. Extensions improve compactness and temporal modeling via feature-based rendering, explicit velocity parameters, or static–dynamic factorization [14, 19, 22, 31]. Temporal pruning and sensitivity analysis further reduce redundancy [15, 41, 42]. However, the lack of persistent geometric tracking limits compression in highly dynamic regions and reduces robustness under sparse or monocular supervision.

Semantic Embedding with Gaussian Splatting combines Gaussian representations with foundation models [4, 11] for 3D scene understanding. Extensions to dynamic settings build upon deformation-based frameworks to maintain temporal identity [8, 13, 17, 18]. However, these approaches inherit the computational overhead of deformation architectures, as high-dimensional semantic features must be evaluated and rendered for every primitive at each frame.

These limitations motivate representations that preserve persistent geometry, maintain sparse parameterization, and enable high-fidelity dynamic rendering.

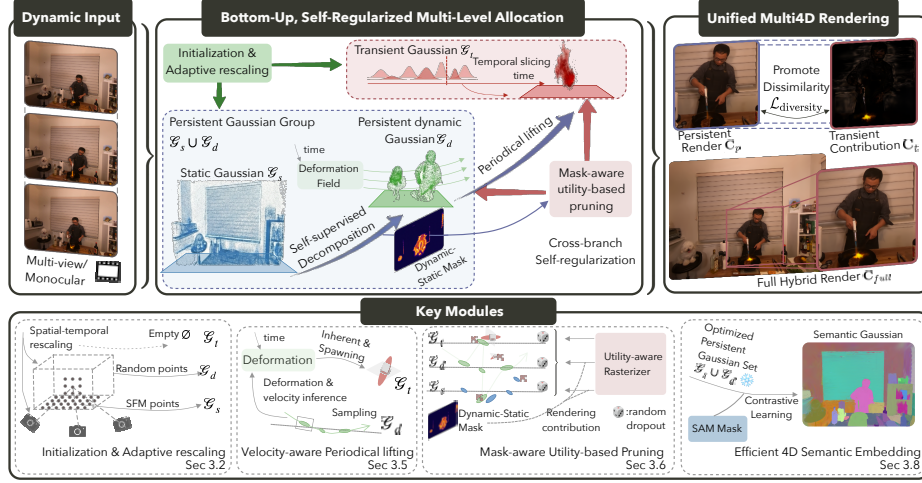


Fig. 2: Overview of the Multi4D pipeline. We employ a bottom-up training scheme that enables competitive allocation across multi-level Gaussian subsets through cross-set self-supervision. After optimization, the persistent subset $\mathcal{G}_p$ is frozen and used for efficient downstream 4D segmentation.

3 Method

We formulate dynamic reconstruction over $\mathcal{S} = \{I_t\}_{t=1}^T$ using a bottom-up multi-level competitive allocation framework. Instead of prescribing fixed roles, modeling responsibility emerges through optimization among three specialized Gaussian subsets that compete under a shared photometric objective.

We represent the scene as $\mathcal{G} = \mathcal{G}_s \cup \mathcal{G}_d \cup \mathcal{G}_t$: **Static** ($\mathcal{G}_s$): time-invariant 3D Gaussians anchoring stable structure. **Persistent Dynamic** ($\mathcal{G}_d$): canonical Gaussians driven by a geometric-only deformation field Φ_g based on HexPlane [3, 32] that predicts rigid motion: $(\mu_t, \mathbf{r}_t) = (\mu, \mathbf{r}) + \Phi_g(\mu, t)$, preserving temporal identity without appearance drift. **Transient Dynamic** ($\mathcal{G}_t$): 4D spatiotemporal Gaussians modeling high-frequency appearance changes and transient geometry.

An overview of the pipeline is shown in Fig. 2. All subsets are jointly rendered within a unified differentiable rasterizer, where shared transmittance couples their gradients and naturally induces competition across subsets. Once one subset explains a region, residual-driven densification in the remaining subsets is suppressed. The following sections detail the resulting bottom-up multi-level optimization strategy.

3.1 Preliminaries: 3D & 4D Gaussian Splatting

3D Gaussian Splatting [10] provides an explicit representation of a static 3D scene using a set of anisotropic Gaussian primitives $\mathcal{G}$. Each primitive $g \in \mathcal{G}$ is

defined by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$, formulated as:

$$g(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right), \quad \text{s.t.} \quad \boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^T\mathbf{R}^T, \quad (1)$$

where $\mathbf{R}$ is the rotation matrix (derived from a quaternion $\mathbf{r}$) and $\mathbf{S} = \text{diag}(\mathbf{s})$ is the scale matrix, respectively. Differentiable splatting [39] is used to render these Gaussians onto the image plane. The final color $\mathbf{C}$ at pixel $\mathbf{u}$ is computed by blending the contributions of all N depth-sorted Gaussians:

$$\mathbf{C}(\mathbf{u}) = \sum_{i=1}^N \mathbf{c}_i(\mathbf{d}) \sigma_i \mathcal{P}_i(g_i, \mathbf{u}) \prod_{j=1}^{i-1} (1 - \sigma_j \mathcal{P}_j(g_j, \mathbf{u})), \quad (2)$$

with $\mathbf{c}_i(\mathbf{d}) \in \mathbb{R}^k$ denoting spherical harmonic (SH) coefficients evaluated at the view direction $\mathbf{d}$, σ_i denoting the opacity, and $\mathcal{P}_i$ denoting the 2D projection of each Gaussian g_i .

4D Gaussian Splatting. To capture temporal appearances without explicit deformation fields, native 4D approaches [37] extend Gaussian primitives into spacetime using 4D ellipsoids defined by $\boldsymbol{\mu}_{4D} = (\boldsymbol{\mu}_{xyz}, \mu_t) \in \mathbb{R}^4$ and a 4D covariance matrix $\boldsymbol{\Sigma}_{4D}$. At a given timestamp t , the time-conditioned 3D Gaussian geometry is analytically derived via slicing:

$$\begin{aligned} \boldsymbol{\mu}_{xyz|t} &= \boldsymbol{\mu}_{1:3} + \boldsymbol{\Sigma}_{1:3,4} \boldsymbol{\Sigma}_{4,4}^{-1} (t - \mu_t), \\ \boldsymbol{\Sigma}_{xyz|t} &= \boldsymbol{\Sigma}_{1:3,1:3} - \boldsymbol{\Sigma}_{1:3,4} \boldsymbol{\Sigma}_{4,4}^{-1} \boldsymbol{\Sigma}_{4,1:3}. \end{aligned} \quad (3)$$

Moreover, the view-dependent color is extended to 4D spherical harmonic (4DSH) [37], capturing time-varying appearance with $\mathbf{c}_{4D}(\mathbf{d}, t - \mu_t)$.

3.2 Initialization and Adaptive Rescaling

Initialization via Inverse Expressiveness. To enable bottom-up specialization during optimization, we impose an inductive bias hierarchy by initializing each Gaussian subset inversely proportional to its expressive capacity. The constrained static subset $\mathcal{G}_s$ is densely initialized using COLMAP [26] points to capture stable scene structure, preventing static elements from being incorrectly modeled as dynamic motion during optimization. The persistent dynamic subset $\mathcal{G}_d$ is initialized with sparse, randomly generated points, allowing gradual dynamic geometric refinement under self-supervision. The highly expressive transient subset $\mathcal{G}_t$ is initialized empty and instantiated only through periodic lifting from $\mathcal{G}_d$ in later training stages, preventing overfitting to reconstruction noise.

Adaptive Spatial-Temporal Rescaling. Jointly optimizing these three subsets introduces numerical challenges due to inconsistent scene scales. Uncalibrated spatial and temporal extents can lead to gradient imbalance and ill-conditioned 4D covariance matrices, causing severe instabilities (e.g., singularities when inverting $\boldsymbol{\Sigma}_{4,4}$ in Eq. 3). To stabilize optimization, we linearly normalize the spatial and temporal domains prior to training using scaling factors derived from the camera distribution and the total video duration.

3.3 Unified Multi4D Rendering

To ensure consistent depth ordering and coupled optimization, we render all primitives in a single differentiable pass at timestamp t . Each primitive $i \in \mathcal{G}$ is projected into an instantaneous 3D state $\Theta_{t,i} = \{\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \sigma_t, \mathbf{c}_t\}$:

$$\Theta_{t,i} = \begin{cases} (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma, \mathbf{c}_{3D}) & \text{if } g_i \in \mathcal{G}_s, \\ (\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \sigma, \mathbf{c}_{3D}) & \text{if } g_i \in \mathcal{G}_d, \\ (\boldsymbol{\mu}_{xyz|t}, \boldsymbol{\Sigma}_{xyz|t}, \sigma_t, \mathbf{c}_{4D}) & \text{if } g_i \in \mathcal{G}_t. \end{cases} \quad (4)$$

Static and persistent geometries are defined explicitly (via $\Phi_g(\boldsymbol{\mu}, t)$ for $\mathcal{G}_d$), while transient geometry ($\mathcal{G}_t$) is obtained through 4D slicing (Eq. 3). Transient opacity $\sigma_t = \sigma \exp(-(t - \mu_t)^2 / 2\boldsymbol{\Sigma}_{4,4})$, enables discontinuous temporal modeling, while 4DSH ($\mathbf{c}_{4D}$) captures time-varying appearance.

Unified Blending & Decoupled Outputs. Unlike prior binary static-dynamic designs [14, 22, 33], we base on [22] and further depth-sort all three subsets jointly within a single rasterization. The shared transmittance accumulation couples their gradients and forms the basis of cross-set competition. During the same pass, we extract three outputs without additional sorting: (1) *Full Render* ($\mathbf{C}_{full}$), which blends all $\mathcal{G}$ for photometric supervision; (2) *Persistent Render* ($\mathbf{C}_p$), which evaluates $\mathcal{G}_s \cup \mathcal{G}_d$ while ignoring $\mathcal{G}_t$ transmittance; and (3) *Transient Contribution* ($\mathbf{C}_t$), which accumulates $\mathcal{G}_t$ using the global transmittance of $\mathcal{G}$. This decoupling isolates transient visibility while preserving correct cross-set occlusion reasoning. We further apply an SSIM-based diversity loss $\mathcal{L}_{\text{diversity}}$ between $\mathbf{C}_t$ and $\mathbf{C}_p$ to discourage redundant modeling and encourage subset specialization.

3.4 Self-Supervised Dynamic-Static Decomposition

Following the self-supervised dynamic-static separation strategy introduced in [30], we isolate persistent actors from the background without relying on ground-truth tracking labels. We augment $\mathcal{G}_d$ with a base mask logit m_i and predict a time-dependent offset using an MLP $\mathcal{D}_m$:

$$m'_i(t) = m_i + \mathcal{D}_m(\mathcal{H}(\boldsymbol{\mu}_i, t)).$$

By substituting SH colors with $m'_i(t)$ during rasterization, we obtain a continuous 2D dynamic mask $\mathbf{M}_d \in [0, 1]^{H \times W}$. We then render color images independently from the dynamic ($\mathbf{C}_d$) and static ($\mathbf{C}_s$) subsets, and compute the composite image used for early photometric supervision as

$$\mathbf{C}_{\text{comp}} = \mathbf{M}_d \odot \mathbf{C}_d + (1 - \mathbf{M}_d) \odot \mathbf{C}_s.$$

Densely initializing $\mathcal{G}_s$ from COLMAP [26, 27] while initializing $\mathcal{G}_d$ with sparse random points introduces a structural asymmetry that biases the optimization toward assigning static content to $\mathcal{G}_s$. As a result, the dynamic mask $\mathbf{M}_d$ naturally contracts to regions exhibiting temporal variation.

To further enforce this separation, we introduce a spatially-aware opacity penalty $\mathcal{L}_\alpha$ that uses the inferred mask $\mathbf{M}_d$ as a spatial template. It penalizes persistent dynamic Gaussians ($\mathcal{G}_d$) projecting into static regions ($\mathbf{M}_d \approx 0$), driving redundant primitives toward zero opacity and enabling their removal during subsequent densification and pruning.

3.5 Velocity-Aware Periodical Lifting

Following our bottom-up training strategy, we introduce **Velocity-Aware Periodic Lifting**, which periodically promotes a small set of *active* persistent Gaussians into the transient subset. Once dynamic-static separation stabilizes, we use the deformed mask logit $m'_i(t)$ (Sec. 3.4) as an activity score at timestamp t . We then sample K candidates from the active set $\{g_i \in \mathcal{G}_d \mid m'_i(t) > \tau\}$ and lift them to $\mathcal{G}_t$. This sampling remains efficient since $\mathcal{G}_d$ is kept sparse throughout optimization, allowing lifting to operate on a compact candidate set.

We lift these candidates into $\mathcal{G}_t$ using **Momentum Inheritance**. The parent’s instantaneous velocity is estimated via finite differences $\mathbf{v}_i = (\Phi_g(\boldsymbol{\mu}_i, t + \Delta t) - \Phi_g(\boldsymbol{\mu}_i, t))/\Delta t$, and used to initialize the new 4D primitive:

$$\boldsymbol{\mu}_{4D}^{(new)} = [\boldsymbol{\mu}_i(t) + \epsilon, t]^T, \quad \mathbf{r}_{4D}^{(new)} \leftarrow \text{Align}(\mathbf{v}_i). \quad (5)$$

A small offset ϵ is added towards the camera center to prevent immediate occlusion by parent gaussians, while Align orients the 4D principal axis along the spatiotemporal trajectory $[\mathbf{v}_i^T, 1]^T$. Momentum inheritance provides a strong motion prior for the high-capacity transient subset, mitigating the instability of unconstrained 4D optimization under sparse or monocular supervision. Once promoted, transient primitives autonomously densify to model high-frequency appearance residuals, while mask-aware pruning removes mutually redundant primitives across $\mathcal{G}_d$ and $\mathcal{G}_t$. This controlled handover allows $\mathcal{G}_t$ to absorb non-rigid residual phenomena, while the geometric-only deformation field Φ_g models rigid motion with stable temporal correspondence.

3.6 Mask-Aware Utility-Based Pruning

To maintain subset specialization and suppress redundant cross-set modeling, we introduce mask-aware pruning based on each Gaussian’s contribution to the final rendering, overcoming the limitations of opacity-based pruning. For each Gaussian $g_i \in \mathcal{G}$ and view I , we define its peak visible contribution:

$$w_{i,I} = \max_{\mathbf{u} \in I} \left(\sigma_i \mathcal{P}_i(g_i, \mathbf{u}) \prod_{j=1}^{i-1} (1 - \sigma_j \mathcal{P}_j(g_j, \mathbf{u})) \right) \cdot M(\mathbf{u}), \quad (6)$$

where the gating mask $M(\mathbf{u})$ applies $\mathbf{M}_d$ for $\mathcal{G}_d$, $(1 - \mathbf{M}_d)$ for $\mathcal{G}_s$, and 1 for $\mathcal{G}_t$. This foreground-background-aware gating ensures that persistent and static primitives contribute only within their assigned regions, preventing cross-set overlap. For the transient subset $\mathcal{G}_t$, the accumulated transmittance spans the

entire depth-sorted union, allowing $w_{i,I}$ to incorporate cross-set occlusion reasoning and suppress transient noise hidden behind solid geometry. We aggregate contributions over a window $\mathcal{I}_s$ to obtain a final score:

$$s_i = \beta \cdot \max_{I \in \mathcal{I}_s} (w_{i,I}) + (1 - \beta) \cdot \frac{1}{|\mathcal{I}_s|} \sum_{I \in \mathcal{I}_s} w_{i,I}. \quad (7)$$

Here, β balances preserving occasionally important high-contribution primitives against removing consistently unused ones. Primitives with $s_i < \tau_{\text{prune}}$ are removed, eliminating low-utility primitives while preserving essential scene structure. To further discourage view-dependent overfitting, we apply Stochastic Primitive Dropout [23], randomly disabling primitives during training to encourage cooperative modeling across subsets.

3.7 Training Strategy and Objectives

We optimize Multi4D end-to-end using a two-stage training schedule that first establishes subset specialization and then focuses on rendering refinement. The overall objective is

$$\mathcal{L}_{total} = \mathcal{L}_{color} + \lambda_{sep} \mathcal{L}_{sep} + \lambda_{reg} \mathcal{L}_{reg} + \lambda_{div} \mathcal{L}_{diversity}. \quad (8)$$

Here, $\mathcal{L}_{color}$ denotes photometric supervision, while $\mathcal{L}_{sep}$ denotes dynamic-static separation loss, and $\mathcal{L}_{reg}$ stands for regularization losses.

Phase I (Subset Formation). During the early stage, explicit cross-set mechanisms are activated, including dynamic-static separation Sec. 3.4 and velocity-aware periodic lifting Sec. 3.5. Together with the unified Multi4D renderer Sec. 3.3 and mask-aware pruning Sec. 3.6, these mechanisms allow the representation to progressively reorganize across $\mathcal{G}_s$, $\mathcal{G}_d$, and $\mathcal{G}_t$.

Phase II (Rendering Refinement). In contrast to Phase I, dynamic-static decomposition and lifting are disabled once subset specialization stabilizes. Optimization then focuses on refining geometry and appearance using the unified Multi4D renderer, while mask-aware pruning remains active to remove redundant primitives and maintain a compact representation.

Losses and training details are provided in the supplementary material.

3.8 Downstream Application: Efficient 4D Segmentation

To demonstrate the downstream utility of our decomposed representation, we perform 4D segmentation by adapting the contrastive feature distillation framework of TRASE [18]. We define the semantic scope on the persistent subset $\mathcal{G}_p = \mathcal{G}_s \cup \mathcal{G}_d$, excluding the transient set $\mathcal{G}_t$.

Following appearance convergence, we freeze all geometric parameters and append a learnable semantic feature $\mathbf{f}_i \in \mathbb{R}^{32}$ to each $i \in \mathcal{G}_p$. We render the semantic map $\hat{S}$ via splatting and supervise it using a soft-mined contrastive objective [18] against 2D SAM masks M_{sam} : $\mathcal{L}_{sem} = \mathcal{L}_{pos}(\hat{S}, M_{sam}) + \mathcal{L}_{neg}(\hat{S}, M_{sam})$.

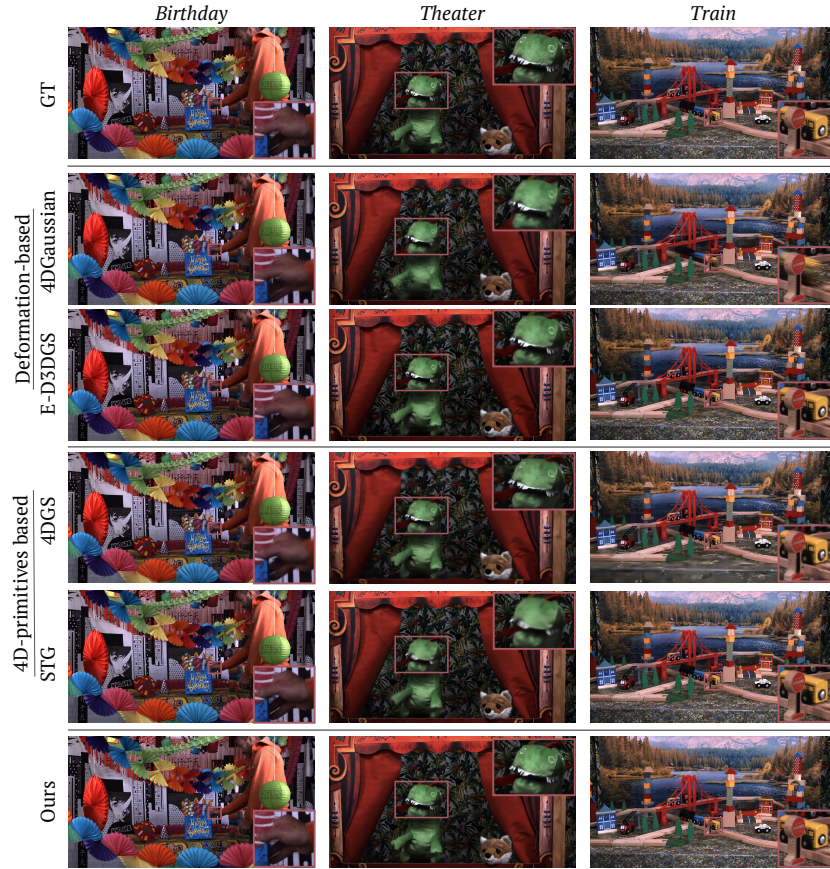


Fig. 3: Qualitative comparisons with baseline methods [2, 19, 32, 37] on Technicolor dataset [25]. Our method shows consistently superior dynamic details modeling.

Finally, to enforce internal spatial coherence within actors, we apply KNN feature smoothing directly in canonical 3D space: $\mathbf{f}_i \leftarrow \text{Normalize}\left(\frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \mathbf{f}_j\right)$. Additional details are provided in the supplementary material.

4 Experiments

To evaluate both rendering fidelity and tracking capability, we benchmark Multi4D on three tasks: (1) multi-view dynamic novel view synthesis, (2) monocular dynamic novel view synthesis, and (3) downstream 4D segmentation.

4.1 Experimental Setup

Datasets. We evaluate on three dynamic-scene benchmarks. For multi-view reconstruction, we use **Technicolor** [25] (five scenes at 2048×1088 , excluding

Table 1: Quantitative comparisons with SOTA methods on the Technicolor Dataset [25]. The **best**, **second best**, and **third best** results are highlighted.

Method	Birthday		Fabien		Painter		Theater		Train		Mean		
	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	FPS $\uparrow$
DyNeRF [16]	29.20	-	32.76	-	35.95	-	29.53	-	31.58	-	31.80	-	0.02
HyperReel [1]	29.99	0.039	34.70	0.053	35.91	0.039	33.32	0.053	29.74	0.053	32.70	0.047	4.00
4DGaussians [32]	29.80	0.050	33.36	0.068	34.52	0.051	30.26	0.076	26.39	0.113	30.86	0.071	35
Def-3DGS [38]	30.68	0.044	33.33	0.067	34.71	0.050	29.65	0.077	26.39	0.110	30.95	0.070	76
E-D3DGS [2]	31.88	0.033	34.69	0.061	35.97	0.036	31.04	0.064	30.87	0.053	32.89	0.049	79
4DGS [37]	31.00	0.038	33.57	0.058	35.73	0.042	31.29	0.070	28.79	0.059	32.07	0.054	55
STG [19]	31.65	0.029	35.61	0.047	35.73	0.037	31.16	0.060	32.61	0.030	33.35	0.040	86
Ours	31.88	0.033	36.34	0.046	36.62	0.035	33.49	0.062	33.20	0.026	34.30	0.037	161

Table 2: Quantitative comparisons with SOTA methods on the Neu3D dataset [16]. The **best**, **second best**, and **third best** results are highlighted.

Method	Cut Beef		Cook Spinach		Sear Steak		Flame Steak		Flame Salmon		Coffee Martini		Mean		
	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	FPS $\uparrow$
NeRFPlayer ² [28]	29.35	0.046	30.56	0.036	29.13	0.046	31.93	0.025	31.65	0.030	31.53	0.025	30.69	0.034	0.05
HyperReel [1]	32.92	0.028	32.30	0.030	32.57	0.024	32.20	0.026	28.26	0.059	28.37	0.054	31.10	0.036	2.00
HexPlane ^{1,2} [3]	32.55	-	32.04	-	32.39	-	32.08	-	29.47	-	-	-	31.71	-	0.56
Def-3DGS [38]	31.43	0.033	33.06	0.027	33.01	0.024	31.83	0.025	28.70	0.043	27.88	0.047	30.98	0.033	29
4DGaussian [32]	32.66	0.027	32.46	0.026	32.49	0.022	32.75	0.023	29.00	0.044	27.34	0.048	31.12	0.032	53
DeGauss [30]	32.56	0.022	32.61	0.025	33.20	0.022	32.75	0.023	29.23	0.042	28.80	0.042	31.52	0.029	157
E-D3DGS [2]	33.02	0.021	32.71	0.022	31.91	0.020	30.23	0.024	29.79	0.036	29.56	0.032	31.20	0.026	70
4DGS [37]	33.23	0.023	32.73	0.025	33.44	0.020	33.19	0.020	28.86	0.043	27.98	0.044	31.57	0.029	114
STG ² [19]	33.55	0.021	33.18	0.022	33.89	0.017	33.59	0.018	29.48	0.038	28.55	0.042	32.04	0.026	140
Ours	34.02	0.021	33.30	0.022	34.19	0.018	34.17	0.018	29.33	0.038	28.80	0.042	32.30	0.026	217

¹ Excludes the *Coffee Martini* scene

² Due to severe memory overflow, evaluated by training six independent models on 50-frame sequence chunks.

camera row 2 column 2 following [1]) and **Neu3D** [16] (300-frame sequences at 1352×1014). For monocular reconstruction, we use **NeRF-DS** [36]. For 4D segmentation, we adopt the **Neu3D-Mask** [18] benchmark. All experiments run on a single NVIDIA RTX 4090 GPU.

Baselines. For novel view synthesis, we compare with NeRF-based methods [1, 16, 24, 28, 36] and Gaussian-based approaches from two domains: deformation-based methods (Def-3DGS [38], 4DGaussian [32], DeGauss [30], E-D3DGS [2]) and 4D-primitive-based methods (4DGS [37], STG [19]). For 4D segmentation, we compare with OpenGaussian [34], SA4D [8], and TRASE [18].

Metrics. For novel view synthesis, we report PSNR $\uparrow$, DSSIM $\downarrow$, and rendering speed (FPS $\uparrow$). DSSIM follows the DSSIM₁ formulation with data range 1.0 as in prior work [19, 32]. For 4D segmentation, we report mIoU $\uparrow$ and mAcc $\uparrow$.

4.2 Experimental Results

We evaluate Multi4D on the Technicolor dataset [25]. Quantitative and qualitative results are shown in Tab. 1 and Fig. 3. Our method improves PSNR by 0.95 dB

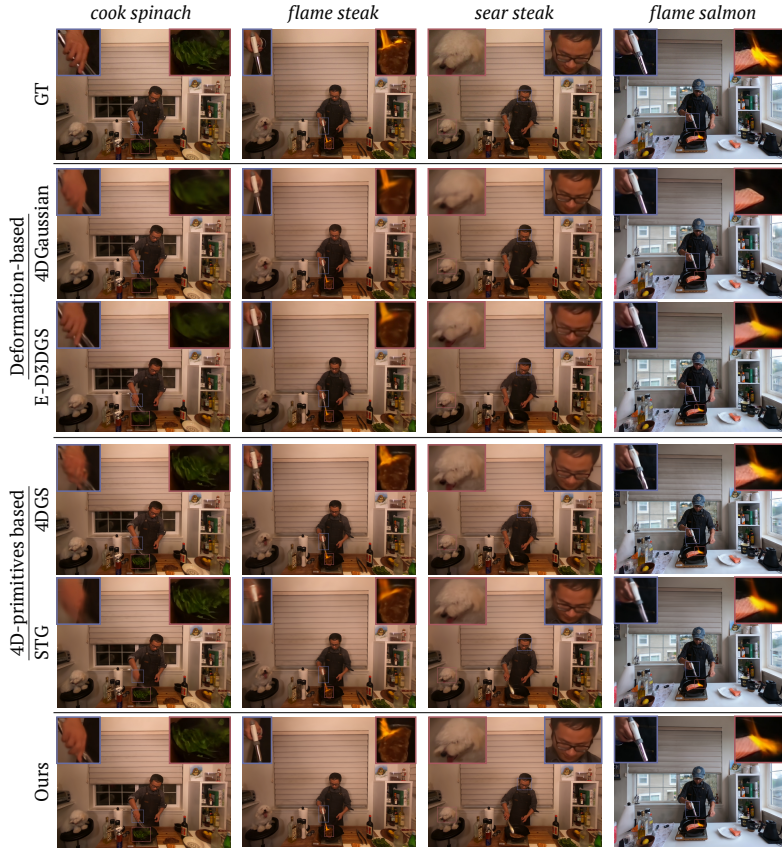


Fig. 4: Qualitative comparisons on the Neu3D dataset [16]. State-of-the-art deformation-based methods [2, 32] suffer from over-smoothing in regions with complex appearance changes (highlighted in red), while 4D-primitive approaches [19, 37] exhibit broken geometry in fast-moving areas (highlighted in blue). Multi4D successfully mitigates both artifacts and consistently achieves the highest visual fidelity.

over the strongest Gaussian baseline [2, 19, 37] while achieving real-time rendering at 161 FPS. Multi4D also achieves state-of-the-art results on Neu3D with superior reconstruction quality and 217 FPS, as shown in Fig. 4 and Tab. 2. As illustrated in Figs. 3 and 4, Multi4D better preserves high-frequency dynamic details, whereas deformation-based methods exhibit temporal blurring and 4D-primitive approaches produce fragmented geometry.

We further evaluate Multi4D on the monocular NeRF-DS dataset [36], which exposes limitations of 4D-primitive frameworks [19, 37] under sparse supervision. As shown in Tab. 3, these methods lacking a holistic motion prior degrade significantly and often produce floating artifacts (see Fig. 5). In contrast, Multi4D leverages persistent Gaussians $\mathcal{G}_d$ for coherent motion modeling and transient

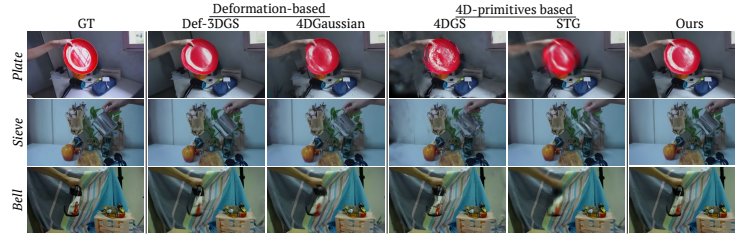


Fig. 5: Qualitative results on NeRF-DS [36]. 4D-primitive-based methods [19,37] degrade heavily under monocular supervision, while Multi4D preserves coherent geometry and fine specular details via decoupled motion and appearance modeling.

Table 3: Quantitative comparison with competitive Nerf methods [24,36], deformation-based gaussian methods [32,38], and 4d primitive-based methods on Monocular NeRF-DS [36] dataset. The **best**, **second best**, and **third best** results are highlighted.

	As		Basin		Bell		Cup		Plate		Press		Sieve		Mean	
Method	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$	PSNR $\uparrow$	DSSIM $\downarrow$
NeRF-DS [36]	25.34	0.060	20.23	0.097	22.57	0.109	24.51	0.060	19.70	0.109	25.35	0.065	24.99	0.065	23.24	0.081
HyperNeRF [24]	17.59	0.074	22.58	0.092	19.80	0.118	15.45	0.085	21.22	0.109	16.54	0.090	19.92	0.074	19.01	0.092
Def-3DGS [38]	26.04	0.060	19.53	0.107	23.96	0.103	24.49	0.059	19.07	0.132	25.52	0.070	25.37	0.069	23.43	0.086
4DGaussian [32]	24.85	0.068	19.26	0.117	22.86	0.099	23.82	0.065	18.77	0.115	24.82	0.082	25.16	0.072	22.79	0.088
4DCS [37]	23.44	0.089	18.66	0.135	22.84	0.106	22.21	0.082	19.06	0.123	22.52	0.118	21.87	0.108	21.51	0.108
STG [19]	24.57	0.072	19.17	0.121	22.82	0.097	23.33	0.068	17.90	0.120	24.89	0.079	25.16	0.072	22.54	0.089
Ours	26.19	0.058	19.50	0.107	24.74	0.081	24.01	0.057	20.36	0.101	25.28	0.069	25.76	0.066	23.69	0.077

Gaussians $\mathcal{G}_t$ for localized specular highlights, resulting in consistently stronger reconstructions.

Multi4D also extends naturally to 4D Segmentation, achieving state-of-the-art 0.9142 mIoU on the Neu3D-Mask benchmark (Tab. 4). By restricting semantic optimization to the persistent subset $\mathcal{G}_p$, we avoid transient appearance noise while preserving temporal identity. As shown in Fig. 6, Multi4D produces sharper masks and more stable tracks than TRASE [18]. Our representation uses only 13k dynamic Gaussians (vs. 624k in TRASE), enabling 32-dimensional feature rendering at 204 FPS—nearly a $10\times$ speedup over the 21 FPS baseline.

Additional per-component renderings and qualitative novel view synthesis results are provided in the supplementary material.

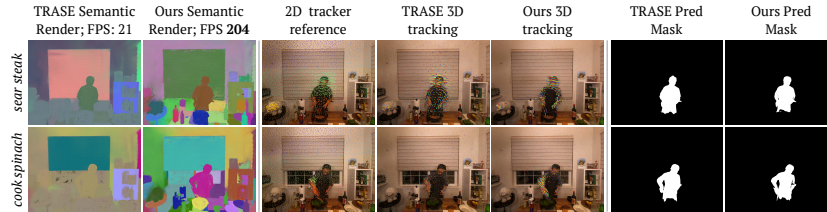


Fig. 6: Qualitative Evaluation of 4D Segmentation with TRASE [18]. From left to right, we visualize high-dimensional semantic feature renderings, 3D motion trajectories (using Co-tracker [9] as a point-based reference), and predicted 2D object masks.

Table 4: Quantitative results for semantic segmentation on the *Neu3D-Mask* benchmark [18]. The **best**, **second best**, and **third best** results are highlighted.

Method	Coffee Martini		Cook Spinach		Cut Beef		Flame Steak		Sear Steak		Average	
	mIOU $\uparrow$	mACC $\uparrow$	mIOU $\uparrow$	mACC $\uparrow$	mIOU $\uparrow$	mACC $\uparrow$	mIOU $\uparrow$	mACC $\uparrow$	mIOU $\uparrow$	mACC $\uparrow$	mIOU $\uparrow$	mACC $\uparrow$
OpenGaussian [34]	0.8254	0.9896	0.6336	0.9798	0.9115	0.9951	0.8199	0.9907	0.8986	0.9943	0.8178	0.9899
SA4D [8]	0.8583	0.9910	0.8987	0.9941	0.8645	0.9914	0.8898	0.9940	0.9047	0.9948	0.8832	0.9931
TRASE [18]	0.9094	0.9946	0.9048	0.9946	0.8962	0.9935	0.8572	0.9920	0.8984	0.9943	0.8932	0.9938
Ours	0.9108	0.9947	0.9161	0.9954	0.9424	0.9967	0.8698	0.9929	0.9321	0.9963	0.9142	0.9952

5 Ablation Study & Efficiency Analysis

Table 5: Ablation of core components on Neu3D dataset [16] (avg. over 4 scenes).

Configuration	PSNR $\uparrow$	DSSIM $\downarrow$	Dyn. GS # ¹ $\downarrow$	Storage ² $\downarrow$
Baseline 4DGS [37]	33.14	0.0219	4215 k	2.6 GB
w/o $\mathcal{G}_d$ (Persistent dynamic subset)	32.78	0.0237	1139 k	727.5 MB
w/o $\mathcal{G}_t$ (Transient dynamic subset)	32.86	0.0217	25 k	105.4 MB
w/o Periodical Lifting (Sec. 3.5)	33.22	0.0216	(13 k + 132 k)	184.84 MB
w/o $\mathcal{L}_{\text{diversity}}$ (Sec. 3.3)	33.66	0.0203	(19 k + 257 k)	263.8 MB
w/o Mask-Aware Pruning (Sec. 3.6)	33.68	0.0199	(70 k + 659 k)	527.9 MB
Multi4D (Full)	33.92	0.0197	(13 k + 152 k)	214.7 MB

¹ Dyn. GS #: Dynamic Gaussians number, denoted as (Persistent $\mathcal{G}_d$ + Transient $\mathcal{G}_t$).

² Storage denotes the combined size of the three Gaussian subsets and the deformation network.

Ablation Study. We evaluate the core components of *Multi4D* on four Neu3D scenes (*Cut Beef*, *Cook Spinach*, *Sear Steak*, *Flame Steak*), summarized in Tab. 5.

Role of Dynamic Subsets $\mathcal{G}_d$, $\mathcal{G}_t$. Removing the persistent dynamic subset ($\mathcal{G}_d$) degrades reconstruction quality (32.78 PSNR), as motion must be approximated by randomly initialized transient primitives. Without stable structural initialization, accurate 4D modeling becomes difficult, leading to fragmented geometry and excessive primitive growth. Conversely, removing the transient subset ($\mathcal{G}_t$) limits high-frequency appearance modeling, reducing PSNR to 32.86 despite a compact representation (25k dynamic Gaussians). This confirms that $\mathcal{G}_d$ models coherent motion while $\mathcal{G}_t$ captures residual appearance changes.

Velocity-Aware Periodical Lifting. Replacing velocity-aware lifting with random initialization reduces PSNR by 0.70 dB (33.92 $\rightarrow$ 33.22), as transient primitives lack inherited motion priors and under-model high-frequency dynamics.

Diversity Loss. Removing $\mathcal{L}_{\text{diversity}}$ weakens subset specialization, causing redundant modeling across persistent and transient sets. This increases dynamic primitives by 67% (276k vs. 165k) and enlarges storage while reducing fidelity.

Mask-Aware Pruning. Without the mask-aware utility-based pruning, temporal over-parameterization re-emerges: the modeled dynamic Gaussian number grows to 729k and storage increases by 145%, while unregularized Gaussians degrade rendering quality.

Efficiency and Compactness. Compared to 4DGS [37], *Multi4D* uses only 165k dynamic Gaussians—a $25\times$ reduction from 4.2M—while improving PSNR

to 33.92. Model size drops from 2.6 GB to 214.7 MB, and training converges in 1.2 hours versus 5.5 hours on a single RTX 4090 ($4.6\times$ faster).

6 Discussion and Conclusion

We presented *Multi4D*, a multi-level framework for dynamic 3D Gaussian Splatting that resolves the tension between motion consistency and photometric fidelity. By competitively allocating modeling capacity across static, persistent, and transient Gaussian subsets under a unified objective, Multi4D enables residual-driven specialization instead of monolithic representations, suppressing motion over-factorization and temporal over-parameterization to produce compact, high-fidelity reconstructions. This structured decomposition preserves coherent motion while capturing high-frequency appearance variation, and performing semantic reasoning on the persistent subset further enables efficient and highly accurate downstream 4D Segmentation.

Limitations and Future Work. Although Multi4D substantially reduces the number of dynamic primitives through optimization-driven compactness, it does not currently incorporate explicit attribute compression. Future work could explore post-training deformation distillation, Gaussian quantization, or lightweight deformation parameterizations to further translate structural compactness into improved storage efficiency.

References

1. Attal, B., Huang, J.B., Richardt, C., Zollhoefer, M., Kopf, J., O’Toole, M., Kim, C.: Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16610–16620 (2023)
2. Bae, J., Kim, S., Yun, Y., Lee, H., Bang, G., Uh, Y.: Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. In: European Conference on Computer Vision (ECCV) (2024)
3. Cao, A., Johnson, J.: Hexplane: A fast representation for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 130–141 (2023)
4. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
5. Duan, Y., Wei, F., Dai, Q., He, Y., Chen, W., Chen, B.: 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–11 (2024)
6. Fan, C.D., Chang, C.W., Liu, Y.R., Lee, J.Y., Huang, J.L., Tseng, Y.C., Liu, Y.L.: Spectromotion: Dynamic 3d reconstruction of specular scenes. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 21328–21338 (2025)
7. Huang, Y.H., Sun, Y.T., Yang, Z., Lyu, X., Cao, Y.P., Qi, X.: Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4220–4230 (2024)

8. Ji, S., Wu, G., Fang, J., Cen, J., Yi, T., Liu, W., Tian, Q., Wang, X.: Segment any 4d gaussians. *arXiv preprint arXiv:2407.04504* (2024)
9. Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., Rupprecht, C.: Cotracker: It is better to track together. In: *European conference on computer vision*. pp. 18–35. Springer (2024)
10. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., et al.: 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* **42**(4), 139–1 (2023)
11. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: *Proceedings of the IEEE/CVF international conference on computer vision*. pp. 4015–4026 (2023)
12. Kwak, S., Kim, J., Jeong, J.Y., Cheong, W.S., Oh, J., Kim, M.: 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*. pp. 11338–11348 (2025)
13. Labe, I., Issachar, N., Lang, I., Benaïm, S.: Dgd: Dynamic 3d gaussians distillation. In: *European Conference on Computer Vision*. pp. 361–378. Springer (2024)
14. Lee, J., Won, C., Jung, H., Bae, I., Jeon, H.G.: Fully explicit dynamic gaussian splatting. *Advances in Neural Information Processing Systems* **37**, 5384–5409 (2024)
15. Lee, M., Lee, B., Yunkyu Lee, L., Lee, E., Kim, S., Song, S., Lee, J.C., Ko, J.H., Park, J., Park, E.: Optimized minimal 4d gaussian splatting. *arXiv e-prints* pp. arXiv–2510 (2025)
16. Li, T., Slavcheva, M., Zollhoefer, M., Green, S., Lassner, C., Kim, C., Schmidt, T., Lovegrove, S., Goesele, M., Newcombe, R., et al.: Neural 3d video synthesis from multi-view video. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 5521–5531 (2022)
17. Li, W., Zhou, R., Zhou, J., Song, Y., Herter, J., Qin, M., Huang, G., Pfister, H.: 4d langspat: 4d language gaussian splatting via multimodal large language models. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. pp. 22001–22011 (2025)
18. Li, Y.J., Gladkova, M., Xia, Y., Cremers, D.: Trase: Tracking-free 4d segmentation and editing. *arXiv preprint arXiv:2411.19290* (2024)
19. Li, Z., Chen, Z., Li, Z., Xu, Y.: Spacetime gaussian feature splatting for real-time dynamic view synthesis. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 8508–8520 (2024)
20. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In: *2024 International Conference on 3D Vision (3DV)*. pp. 800–809. IEEE (2024)
21. Miao, Q., Quan, J., Li, K., Xu, Y., Yang, Y., Luo, Y.: Frequency-aware dynamic gaussian splatting. In: *The Fourteenth International Conference on Learning Representations* (2026)
22. Oh, S., Lee, Y., Jeon, H., Park, E.: Hybrid 3d-4d gaussian splatting for fast dynamic scene representation. *arXiv preprint arXiv:2505.13215* (2025)
23. Park, H., Ryu, G., Kim, W.: Dropgaussian: Structural regularization for sparse-view gaussian splatting. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. pp. 21600–21609 (2025)
24. Park, K., Sinha, U., Hedman, P., Barron, J.T., Bouaziz, S., Goldman, D.B., Martin-Brualla, R., Seitz, S.M.: Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228* (2021)
25. Sabater, N., Boisson, G., Vandame, B., Kerbiriou, P., Babon, F., Hog, M., Gendrot, R., Langlois, T., Bureller, O., Schubert, A., et al.: Dataset and pipeline for multi-

- view light-field video. In: Proceedings of the IEEE conference on computer vision and pattern recognition Workshops. pp. 30–40 (2017)
26. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
 27. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV) (2016)
 28. Song, L., Chen, A., Li, Z., Chen, Z., Chen, L., Yuan, J., Xu, Y., Geiger, A.: Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics* **29**(5), 2732–2742 (2023)
 29. Tu, A., Ying, H., Hanson, A., Lee, Y., Goldstein, T., Zwicker, M.: Speede3dgs: Speedy deformable 3d gaussian splatting with temporal pruning and motion grouping. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 26083–26093 (2026)
 30. Wang, R., Lohmeyer, Q., Meboldt, M., Tang, S.: Degauss: Dynamic-static decomposition with gaussian splatting for distractor-free 3d reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6294–6303 (2025)
 31. Wang, Y., Yang, P., Xu, Z., Sun, J., Zhang, Z., Chen, Y., Bao, H., Peng, S., Zhou, X.: Freetimegs: Free gaussian primitives at anytime anywhere for dynamic scene reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21750–21760 (2025)
 32. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 20310–20320 (2024)
 33. Wu, J., Peng, R., Wang, Z., Xiao, L., Tang, L., Yan, J., Xiong, K., Wang, R.: Swift4d: Adaptive divide-and-conquer gaussian splatting for compact and efficient reconstruction of dynamic scene. *arXiv preprint arXiv:2503.12307* (2025)
 34. Wu, Y., Meng, J., Li, H., Wu, C., Shi, Y., Cheng, X., Zhao, C., Feng, H., Ding, E., Wang, J., et al.: Opengaussian: Towards point-level 3d gaussian-based open vocabulary understanding. *Advances in Neural Information Processing Systems* **37**, 19114–19138 (2024)
 35. Xu, J., Fan, Z., Yang, J., Xie, J.: Grid4d: 4d decomposed hash encoding for high-fidelity dynamic gaussian splatting. *Advances in Neural Information Processing Systems* **37**, 123787–123811 (2024)
 36. Yan, Z., Li, C., Lee, G.H.: Nerf-ds: Neural radiance fields for dynamic specular objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8285–8295 (2023)
 37. Yang, Z., Yang, H., Pan, Z., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642* (2023)
 38. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 20331–20341 (2024)
 39. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)* **38**(6), 1–14 (2019)

40. Yoon, J., Han, S., Oh, J., Lee, M.: Splines: Learning smooth trajectories in gaussian splatting for dynamic scene reconstruction. In: The Thirteenth International Conference on Learning Representations (2025)
41. Yuan, Y., Shen, Q., Yang, X., Wang, X.: 1000+ fps 4d gaussian splatting for dynamic scene rendering. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)
42. Zhang, X., Liu, Z., Zhang, Y., Ge, X., He, D., Xu, T., Wang, Y., Lin, Z., Yan, S., Zhang, J.: Mega: Memory-efficient 4d gaussian splatting for dynamic scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 27828–27838 (2025)