# TRAJECTORY-ANCHORED COMPACT GAUSSIANS FOR FEED-FORWARD 4D RECONSTRUCTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Dynamic scene reconstruction from monocular video requires a representation that supports both novel-view rendering and correspondence through time. Feed-forward methods learn reconstruction priors across scenes, enabling inference on unseen videos without costly per-scene optimization. However, current feed-forward methods rely heavily on dense pixel-aligned architecture which produces redundant primitives. On the other hand, recent compact query-based methods can struggle to maintain accurate temporal correspondence. We present a feed-forward framework for compact, structured 4D Gaussian reconstruction without input camera poses. Our central idea is to use dense correspondence to organize Gaussians around a sparse set of explicit motion trajectories. We first predict dense B-spline trajectories with a geometry-anchored parameterization that preserves each trajectory's source point position. Grouping trajectories by spatial proximity, motion, and appearance yields representative anchors. Given learnable tokens paired with the anchor trajectories, a query-based decoder aggregates image evidence to generate child Gaussians, whose local offsets capture geometry and motion beyond each anchor. This hierarchy supports rendering and dense 3D correspondence with a compact Gaussian representation whose size is independent of the number of input pixels. Across the evaluated benchmarks, our compact model outperforms the evaluated pixel-aligned baselines in most rendering metrics while using significantly fewer Gaussians. The same compact representation outperforms the evaluated query-based baselines across all reported rendering and Gaussian-tracking metrics.

## 1 INTRODUCTION

Reconstructing dynamic scenes from monocular video underpins applications in robotics, entertainment, and graphics. These applications not only ask for photorealistic rendering from new viewpoints, but also require a *structured* representation, in which scene elements persist through time and remain in dense correspondence, so that a surface observed in one frame can be located in every other. This reflects object permanence: a moving object remains the same object, and should be represented once and transported through time rather than re-instantiated at every timestamp. A structured representation should also be *compact*, avoiding duplicate primitives for the same surface across space and time. Per-scene optimization can meet parts of these requirements, but it fits each video from scratch at considerable cost of time and computation, and the optimization system tends to be specific to that scene rather than being generalizable.

Feed-forward methods learn reconstruction across diverse videos and produce a 4D scene in a single forward pass, which removes per-scene optimization and generalizes to unseen inputs. Two families dominate in the feed-forward methods. Pixel-aligned methods predict dynamic representations such as 4D Gaussians (Yang et al., 2024a; Luiten et al., 2024) for every input pixel (Xu et al., 2025; Lin et al., 2026a; Yang et al., 2026), retaining direct access to image evidence but inheriting the dense input sampling: a surface observed in several frames produces several primitives that need not agree, so the output carries redundancy across space and time and exhibits ghosting. Query-based decoders instead allocate a fixed budget of tokens (Ren et al., 2026; Kim et al., 2026), which removes this duplication and keeps the output representation compact. However, they do not sufficiently exploit the redundancy they remove: without explicit tracking of moving elements, correspondence over time is left implicit and unsupervised, leading to blurry reconstruction with ambiguous motion.



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.




Figure 1: Our method predicts structured 4D reconstruction given monocular video input. Child Gaussians move with anchor trajectories (middle), yielding a structured 4DGS (bottom).

Our key observation is that temporal correspondence enables compact reconstruction: repeated observations of the same surface can share a persistent representation that follows its motion. This allows information to accumulate across frames without duplicating geometry at each timestamp. We build on this insight with a hierarchical 4D representation that separates motion tracking from rendering detail. A compact set of *anchor trajectories* captures scene motion, with each anchor deforming source-frame geometry through a trajectory in time. *Child Gaussians* are then generated around these anchor trajectories to achieve fine geometry and appearance reconstruction. This hierarchical design thus enables a compact set of primitives to achieve detailed and globally coherent tracking across time.

We propose a hierarchical framework with three components. First, each trajectory is expressed as a source point supplied by a geometry backbone and a spline-based displacement trajectory. We adopt the continuous B-spline trajectory representation of Trace Anything (Liu et al., 2026), but change its parameterization: its absolute control points determine the source position and the motion jointly, which causes shift from the source-frame geometry. We instead predict displacement controls with a null-space-constrained formulation, which makes the displacement exactly zero there for any predicted coefficients by construction. Second, we extract key trajectories from the dense pixel-aligned trajectories by image-local grouping over 3D proximity, relative-distance variation over time, and appearance, yielding explicit representative trajectories. Third, a query-based decoder conditions a learnable token on each key trajectory, reads image features, and emits a local group of child Gaussians whose time-dependent offsets express motion details beyond the anchor trajectory

This hierarchical design allows our method to achieve both coherent global motion and accurate local details. Our method consumes no input camera poses, yet attains the highest PSNR and SSIM among external baselines, from $65K$ Gaussians, one to two orders of magnitude fewer than pixel-aligned feed-forward baselines. The same representation supports dense tracking: our compact model improves on every query-based baselines closing most of the gap to dense pixel-aligned tracking that query-based decoders had previously conceded.

Our contributions are threefold:

- We introduce a hierarchical representation and a query-based architecture that achieves both accurate and compact 4D reconstruction and tracking.
- We introduce a geometry-anchored B-spline parameterization that decouples source geometry from motion by construction, and a key-trajectory extraction procedure that turns dense predictions into a compact motion scaffold for query-based decoding.

- We demonstrate state-of-the-art dynamic novel-view synthesis without input poses at a fraction of the primitive budget, together with tracking accuracy that closes most of the gap between query-based and pixel-aligned representations.

## 2 Related Work

**Per-scene Dynamic Gaussian Splatting.** Dynamic extensions of 3DGS (Kerbl et al., 2023) optimize Gaussians per video. Early methods give each Gaussian its own motion (Luiten et al., 2024; Yang et al., 2024a; Li et al., 2024; Wu et al., 2024; Yang et al., 2024b) and rely on multi-view captures or monocular benchmarks with teleporting cameras (Pumarola et al., 2021; Park et al., 2021a;b; Yoon et al., 2020; Gao et al., 2022); later methods handle casual monocular videos with pretrained depth and 2D tracks, representing motion with SE(3) bases, motion scaffolds, or per-Gaussian splines (Stearns et al., 2024; Liu et al., 2025; Wang et al., 2025b; Zhang et al., 2026b; Lei et al., 2025; Park et al., 2025; Litman et al., 2026). Many share motion across Gaussians through control points, motion nodes, shared trajectories, or rigid groups (Huang et al., 2024; Kwak et al., 2025; Liang et al., 2025b; Kratimenos et al., 2024; Tu et al., 2026). These methods take minutes to hours per video; we keep the idea of shared motion but predict trajectories and Gaussians in a single feed-forward pass.

**Feed-forward Reconstruction.** DUSt3R (Wang et al., 2024) and its multi-view (Wang et al., 2025a; 2026a;b; Keetha et al., 2026; Lin et al., 2026b; Burzio et al., 2026) and dynamic (Zhang et al., 2025b; Wang et al., 2025c; Sucar et al., 2025; 2026; Feng et al., 2025; Zhang et al., 2025c; Karhade et al., 2026) successors predict per-pixel point maps but no renderable scene. Feed-forward Gaussian splatting fills this gap. Pixel-aligned methods place Gaussians on the pixel grid of each input image, in static (Charatan et al., 2024; Zhang et al., 2024; Chen et al., 2024; Ye et al., 2025; Lin et al., 2026b) and dynamic scenes (Liang et al., 2025a; Yang et al., 2025; Xu et al., 2025; Lin et al., 2025; 2026a; Yang et al., 2026; Hur et al., 2026; Balice et al., 2026; NVIDIA, 2026), duplicating surfaces seen in several frames, while query-based methods decode a fixed Gaussian budget from learnable queries (Ren et al., 2026; Li et al., 2026a;b; Kim et al., 2026) but leave the motion implicit. We also decode a fixed set of Gaussians from queries, but attach each query to a predicted anchor trajectory.

**Point Tracking.** Point trackers follow query pixels in 2D (Doersch et al., 2023; Karaev et al., 2024) or 3D (Xiao et al., 2024; 2025; Zhang et al., 2025a), or query any point at any time from a video encoding (Zhang et al., 2026a; Jeon et al., 2026; Chen et al., 2026; Luo et al., 2026). Dense methods predict per-pixel 3D tracks or scene flow (Ngo et al., 2025; Nam et al., 2026; Liang et al., 2025c; Lu et al., 2026; Zhu et al., 2026b); TraceAnything (Liu et al., 2026) fits a clip-long B-spline to every pixel of every frame, and OmniX (Jiang et al., 2026) and SM4RT (Lin et al., 2026c) compress motion into dynamic tokens or rigid bases. None of them models appearance. We turn the all-to-all field of TraceAnything into a renderable scene: we anchor each spline to the predicted source geometry so that it models only motion, and keep a small set of trajectories as anchors for child Gaussians, yielding a compact renderable scene with dense correspondence.

## 3 Method

Given a monocular video $\mathcal{V} = \{\mathbf{I}_i\}_{i=1}^{T}$ of $T$ frames of size $H \times W$ observed at times $\{t_i\}$, with no camera poses provided, our goal is to reconstruct the dynamic scene as a set of time-varying Gaussians $\mathcal{G}(t)$ that can be rendered from a novel viewpoint at any query time $t$ and maintain dense correspondences.

Our method builds this representation in three stages, shown in Fig. 2. First, through the encoder and the spline-based trajectory decoder, our method predicts pixel-aligned B-spline trajectory as an offset on its corresponding point map at its own source time (Sect. 3.2). Second, these pixel-aligned trajectories are grouped by proximity, motion, and appearance, and reduced to a compact set of key trajectories (Sect. 3.3). Finally, the query-based Gaussian decoder uses the anchor trajectories and learnable query tokens as input to generate child Gaussians, producing the final time-varying Gaussians $\mathcal{G}(t)$ (Sect. 3.4).

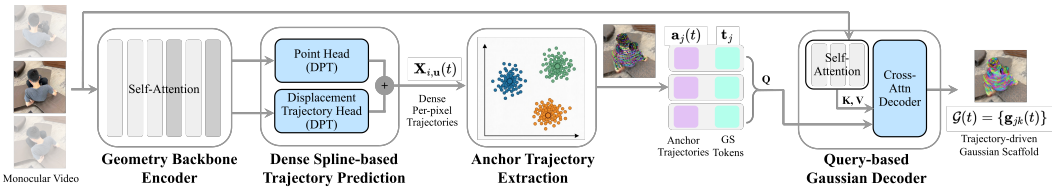


Figure 2: **Overview of our method.** Given a monocular video, a shared encoder and two DPT heads predict source point maps and B-spline displacements, yielding dense geometry-anchored $\mathbf{X}_{i,\mathbf{u}}(t)$. Anchor trajectory extraction selects compact motion anchors $\mathbf{a}_j(t)$, each paired with a learnable token $\mathbf{t}_j$. Conditioned on these anchors Given these anchors and learnable Gaussian tokens as input queries, a query-based decoder aggregates features from a separate image encoder and generates child Gaussians $\mathbf{g}_{jk}(t)$, forming a structured 4D representation for novel-view synthesis and dense correspondence.

### 3.1 Preliminary: 4D Gaussian Splatting

Kerbl et al. (2023) proposed 3D Gaussian Splatting (3DGS), which models a 3D scene by a collection of Gaussian primitives $\mathcal{G} = \{\mathbf{g}_j\}$. Each Gaussian $\mathbf{g}_j$ contains the position, scale, rotation (quaternion), opacity and spherical harmonic coefficients for view-dependent appearance. 4D Gaussian Splatting (4DGS) extends this representation to dynamic scenes by allowing the primitives vary with time (Wu et al., 2024). Each Gaussian thus becomes a function $\mathbf{g}_j(t)$, and the scene at time $t$ is $\mathcal{G}(t) = \{\mathbf{g}_j(t)\}$. The time-varying Gaussians can be efficiently rendered through tile-based rasterizers from arbitrary cameras parametrized by $\boldsymbol{\pi}$, $\mathbf{I}(t) = \mathcal{R}(\mathcal{G}(t), \boldsymbol{\pi})$.

Instantiations of 4DGS differ in which attributes are allowed to vary and how that variation is parameterized. We use a hierarchical representation in which a compact set of anchor trajectories captures scene motion, while child Gaussians attached to each anchor represent local geometry, appearance and dynamics.

### 3.2 Dense Spline-based Trajectory Prediction

**Geometry Backbone.** Given input video $\mathcal{V} = \{\mathbf{I}_i\}_{i=1}^{T}$, we firstly utilize a self-attention encoder similar to DepthAnythingV3 (Lin et al., 2026b) to produce image tokens and estimate camera intrinsic and extrinsic parameters. We simultaneously incorporate a DPT-based point map head to predict pixel-aligned point map $\mathbf{p}_{i,\mathbf{u}}$ for each pixel $\mathbf{u}$ at each timestamp $i$ (associated with time $t_i$).

**Dense Trajectory Prediction.** We predict a continuous 4D trajectory for every pixel in every input frame. Trace Anything (Liu et al., 2026) represent trajectories with cubic B-splines. For pixel $\mathbf{u}$ in source frame $i$, the absolute-control formulation is

$$\mathbf{X}^{\mathrm{TA}}_{i,\mathbf{u}}(t) = \sum_{k=0}^{K-1} B^{(k)}(t)\, \mathbf{P}^{(k)}_{i,\mathbf{u}} = \mathbf{b}(t)\mathbf{P}_{i,\mathbf{u}}, \tag{1}$$

where $\mathbf{b}(t) = [B^{(0)}(t), \ldots, B^{(K-1)}(t)] \in \mathbb{R}^{1\times K}$ contains the cubic basis functions with clamped knots, and $\mathbf{P}_{i,\mathbf{u}} \in \mathbb{R}^{K\times 3}$ represents stacked the 3D control points as rows, which is predicted by a DPT head in (Liu et al., 2026). The splines, predicted independently from the source points, tend to shift from the source points, causing inconsistency.

We instead propose to anchor each trajectory at the world-coordinate point $\mathbf{p}_{i,\mathbf{u}}$ predicted by our geometry backbone at time $t_i$. We define the offset control points $\mathbf{O}_{i,\mathbf{u}} \in \mathbb{R}^{K\times 3}$, which satisfies $\mathbf{P}_{i,\mathbf{u}} = \mathbf{1}_K \mathbf{p}_{i,\mathbf{u}} + \mathbf{O}_{i,\mathbf{u}}$. This gives $\mathbf{X}_{i,\mathbf{u}}(t) = \mathbf{p}_{i,\mathbf{u}} + \mathbf{b}(t)\mathbf{O}_{i,\mathbf{u}}$. Preserving the source point $\mathbf{X}_{i,\mathbf{u}}(t_i) = \mathbf{p}_{i,\mathbf{u}}$ requires linear constraint $\mathbf{b}(t_i)\mathbf{O}_{i,\mathbf{u}} = 0$.

We enforce this constraint by setting $\mathbf{O}_{i,\mathbf{u}} = \mathbf{N}_i\boldsymbol{\Theta}_{i,\mathbf{u}}$, where $\mathbf{N}_i \in \mathbb{R}^{K\times(K-1)}$ is an orthonormal basis of the null space of $\mathbf{b}(t_i)$. The resulting *null-space-constrained (NSC)* trajectory is

$$\mathbf{X}_{i,\mathbf{u}}(t) = \mathbf{p}_{i,\mathbf{u}} + \mathbf{b}(t)\mathbf{N}_i\boldsymbol{\Theta}_{i,\mathbf{u}}, \qquad \text{where } \mathbf{b}(t_i)\mathbf{N}_i = \mathbf{0}. \tag{2}$$

We design a DPT head that predicts the $K-1$ free 3D coefficient vectors $\mathbf{\Theta}_{i,\mathbf{u}} \in \mathbb{R}^{(K-1)\times 3}$ for each pixel $\mathbf{u}$ at each timestamp $i$. We compute $\mathbf{N}_i$ once per source frame from the trailing right singular vectors of $\mathbf{b}(t_i)$ and share it across pixels (Appendix A.3). This guarantees $\mathbf{X}_{i,\mathbf{u}}(t_i) = \mathbf{p}_{i,\mathbf{u}}$ for any predicted coefficients while retaining every spline satisfying the anchor constraint.

### 3.3 Anchor Trajectory Extraction

The dense trajectory field contains repeated observations of the same surfaces and neighboring points with correlated motion. Similar to optimization-based 4DGS methods that start from a sparse set of 3D tracks (Lei et al., 2025; Wang et al., 2025b), we select $M$ anchor trajectories to provide compact motion support for structured Gaussian decoding. Candidate trajectories are sampled on a pixel grid from input frames at a fixed temporal stride. For brevity, let $n$ index a candidate's source timestamp-pixel pair $(i, \mathbf{u})$, with trajectory $\mathbf{X}_n(t)$ and source point $\mathbf{p}_n = \mathbf{p}_{i,\mathbf{u}}$.

**Trajectory Grouping.** We adapt the image-local Density-Peak Clustering (DPC) of Super-Trajectory (Wang et al., 2017), using a dissimilarity metric that combines 3D proximity, relative-distance variation, and appearance. Starting from an image grid, local assignments and density-peak updates form groups whose centers are actual candidate trajectories. For candidates $n$ and $m$, define $d_{nm}(t) = \|\mathbf{X}_n(t) - \mathbf{X}_m(t)\|_2$ over their jointly observed times $\Omega_{nm}$. We then define the criterion inspired by GroupFlow (Tu et al., 2026), augmented with color:

$$D_{nm} = \lambda_{\mathrm{x}} \operatorname{mean}_{t\in\Omega_{nm}} d_{nm}(t) + \lambda_{\mathrm{m}} \operatorname{std}_{t\in\Omega_{nm}} d_{nm}(t) + \|\mathbf{c}_n - \mathbf{c}_m\|_2, \tag{3}$$

where $\lambda_{\mathrm{x}} = 1/(2s_{\mathrm{x}})$ and $\lambda_{\mathrm{m}} = 1/(2s_{\mathrm{m}})$ normalize the spatial and motion terms, with $s_{\mathrm{x}}$ and $s_{\mathrm{m}}$ the medians of the two statistics over neighboring candidates. The color $\mathbf{c}_n$ is sampled at the trajectory's source pixel and normalized to $[0, 1]$. The distance statistics favor nearby trajectories with stable relative separation, as expected under shared rigid motion even when point velocities differ. Finally, we fill separate static and dynamic budgets using the predicted motion mask: dynamic groups contribute their centers in descending group size, and static anchors, fixed at their source points, are chosen to cover the input views, yielding the anchor set $\mathcal{A} = \{\mathbf{a}_j(\cdot)\}_{j=1}^{M}$.

### 3.4 Query-Based Gaussian Decoder

The extracted anchors (Sect. 3.3) form a trajectory-driven scaffold of the dynamic scene. To enrich this scaffold with detailed motion, geometry, and appearance, we use trajectory-conditioned queries to predict a local group of child Gaussians for each anchor, which forms the final structured dynamic 4DGS $\mathcal{G}(t)$.

**Trajectory-driven Gaussian Scaffold.** We use the extracted anchor trajectories from Sect. 3.3 as space-time anchors to generate a group of $N$ child Gaussians. At time $t$, the position of child $k$ associated with anchor $j$ is

$$\boldsymbol{\mu}_{jk}(t) = \mathbf{a}_j(t) + r_j(t)\,\mathbf{u}_{jk}(t), \qquad k = 1, \ldots, N, \tag{4}$$

where $\mathbf{u}_{jk}(t)$ is a predicted offset and $r_j(t)$ scales the offsets. This anchor–offset hierarchy forms a *trajectory-driven Gaussian scaffold*, similar to dynamic scaffold representations (Kwak et al., 2025). The trajectories provide a shared motion reference for each group, while time-dependent offsets accommodate local geometric and motion variations.

**Trajectory-conditioned Queries.** Generating child Gaussians requires image evidence for local placement, shape, and appearance. We therefore pair each anchor trajectory $\mathbf{a}_j(\cdot)$ with a learnable Gaussian token $\mathbf{t}_j$, shared across time, and a learnable initial radius $r_j^{(0)}$, and form its query at time $t$ as

$$\mathbf{q}_j^{(0)}(t) = \mathbf{t}_j + f_{\mathrm{t}}(\gamma_t(t)) + f_{\mathrm{p}}(\gamma_{\mathrm{p}}(\mathbf{a}_j(t))) + f_{\mathrm{r}}\left(\gamma_{\mathrm{r}}(r_j^{(0)})\right), \tag{5}$$

where $\gamma_t$, $\gamma_{\mathrm{p}}$, and $\gamma_{\mathrm{r}}$ are sinusoidal encodings and $f_{\mathrm{t}}$, $f_{\mathrm{p}}$, and $f_{\mathrm{r}}$ are learned projections to the token dimension. Following TokenGS (Ren et al., 2026), a separate ViT encodes the input frames together with their times and Plücker rays, yielding the image memory $\mathbf{F}$; the rays are computed from the cameras predicted by our camera head, so they share the coordinate frame of the anchors.

**Anchor-Guided Gaussian Decoding.** For each target time $t$, a stack of $L$ decoder blocks extracts the final structured 4DGS output from memory $\mathbf{F}$ given query $\mathbf{q}_j^{(0)}(t)$. The blocks follow the anchor-guided cross-attention of LocusGS (Li et al., 2026a), with its static anchor and radius replaced by the time-dependent $\mathbf{a}_j(t)$ and $r_j(t)$. For image patch $p$ with ray $\boldsymbol{\ell}_p$, the attention weights at layer $\ell$ are

$$w_{jp}^{(\ell)}(t) = \mathrm{softmax}_p\left[\frac{\bar{\mathbf{q}}_j^{(\ell)}(t)^\top \mathbf{k}_p}{\sqrt{d}} + b_{\mathrm{geo}}\Big(\mathbf{a}_j(t), \boldsymbol{\ell}_p, r_j^{(\ell)}(t)\Big)\right], \tag{6}$$

where $\bar{\mathbf{q}}, \mathbf{k}$ are projected attention queries and keys, and $d$ is the per-head dimension. The geometric bias $b_{\mathrm{geo}}$ penalizes anchor-to-ray distance similar to (Li et al., 2026a), with the radius controlling its bandwidth. The weighted image values update the queries, followed by self-attention and an MLP. After each block, the updated query predicts a residual update to the radius, applied before a softplus activation that keeps it positive. We use $L = 3$ blocks. Unlike LocusGS (Li et al., 2026a), which also refines the anchor centers, we keep $\mathbf{a}_j(t)$ fixed and update only the query features and radii: moving the anchors would pull them toward the camera rays favored by the attention bias and break the temporal smoothness of the predicted trajectories. Each query jointly predicts positional offsets, opacity, scale, rotation quaternions, and RGB colors for its $N$ children; opacity, scale, and color are shared across time, while positions and orientations vary with $t$. Their centers follow Eq. 4 with $r_j(t) = r_j^{(L)}(t)$. Offsets are unbounded, so the radius is neither a hard spatial bound nor the Gaussian scale. The resulting $MN$ Gaussians form the final $\mathcal{G}(t) = \{\mathbf{g}_{jk}(t)\}$, which support dynamic novel view synthesis and dynamic 3D tracking.

### 3.5 Training

Our method comprises two stages of training. The first stage focuses on learning pixel-aligned tracking as in Sect. 3.2. The second stage trains the query-based Gaussian decoder (Sec. 3.4) to extract structured 4DGS, with compact output size and high-quality dense correspondence.

**Stage I: Pixel-aligned Decoder Training.** We first train the encoder and the pixel-aligned decoders with

$$\mathcal{L}_{\mathrm{I}} = \lambda_g \mathcal{L}_{\mathrm{geo}} + \lambda_t \mathcal{L}_{\mathrm{traj}} + \lambda_p \mathcal{L}_{\mathrm{reproj}} + \lambda_s \mathcal{L}_{\mathrm{static}} + \lambda_r \mathcal{L}_{\mathrm{render}}. \tag{7}$$

Here, $\mathcal{L}_{\mathrm{geo}}$ supervises depth and camera prediction. $\mathcal{L}_{\mathrm{traj}}$ and $\mathcal{L}_{\mathrm{reproj}}$ supervise trajectories at 3D position and 2D projection, respectively. The two trajectory losses utilize a confidence weighting as in Lu et al. (2026). $L_{\mathrm{static}}$ encourage static region to stay static, following Liu et al. (2026). We add additional RGB rendering loss (L1, D-SSIM) $\mathcal{L}_{\mathrm{render}}$ between the rendering of Gaussians and the ground truth images. The 4D Gaussian non-positional attributes are provided by another DPT head, which we also train at this stage.

**Stage II: Query-based Decoder Training.** We then train the query-based Gaussian decoder using ground-truth trajectories as fixed anchors, without jointly optimizing the spline predictor:

$$\mathcal{L}_{\mathrm{II}} = \mathcal{L}_{\mathrm{RGB}} + \lambda_d \mathcal{L}_{\mathrm{depth}} + \lambda_m \mathcal{L}_{\mathrm{motion}}. \tag{8}$$

The RGB rendering loss is pixel-weighted L1, D-SSIM and LPIPS losses, with more emphasis on dynamic region by a factor of 4. We only enable the LPIPS rendering loss at the end of training for stability. The depth loss supervises Gaussian-rendered depth with the same emphasis on dynamic region. As our output 4DGS are no longer pixel-aligned, it is nontrivial to directly supervise their motion against the pixel-aligned ground truth. We therefore associate each ground-truth track with the child Gaussians that render its source pixel, weighted by their detached alpha-compositing contributions, and penalize the displacement error of each contributing Gaussian rather than that of their blended motion. All supervisions are applied on the final decoder outputs. The formal definitions of the trajectory and motion losses and their schedules are provided in Appendix B.

## 4 Experiments

### 4.1 Experimental Setup

**Implementation.** Our foundation model builds on DA3 (Lin et al., 2026b) and predicts per-pixel cubic B-spline trajectories with $K = 7$ control points; the query-based decoder uses $M = 1024$ key

Table 1: **Quantitative comparison on dynamic novel-view synthesis.** DyCheck is evaluated with masking over 32 frames at stride=2, the same protocol as C4G and NeoVerse. Kubric follows the TokenGS setting and reports novel view synthesis metrics. Posed denotes required input camera poses. #Primitives indicates the number of predicted Gaussians. We highlight the best, second-best, and third-best results in each metric column.

| | | | DyCheck | | | NVIDIA | | | Kubric | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | Posed | #Primitives | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| 4DGT | Yes | 544,320 | 12.409 | 0.3064 | 0.5214 | 13.991 | 0.2781 | 0.4570 | 20.203 | 0.6261 | 0.4294 |
| MoVieS | Yes | 4,816,896 | 13.983 | 0.3646 | 0.3499 | 19.175 | 0.5302 | 0.2293 | 20.321 | 0.6456 | 0.2769 |
| NeoVerse | No | 4,816,896 | 11.518 | 0.2404 | 0.5117 | 15.196 | 0.3610 | 0.3493 | 20.747 | 0.5910 | 0.3079 |
| C4G | No | 2,048 | 14.591 | 0.3819 | 0.5796 | 15.806 | 0.3839 | 0.6753 | 19.696 | 0.6029 | 0.4463 |
| TokenGS | Yes | 81,920 | 11.272 | 0.3358 | 0.5944 | 18.208 | 0.5354 | 0.5431 | 19.888 | 0.6475 | 0.3753 |
| Ours (dense) | No | 4,816,896 | 15.359 | 0.434 | 0.3017 | 20.730 | 0.5786 | 0.1673 | 25.669 | 0.8431 | 0.1861 |
| Ours (compact) | No | 65,536 | 14.867 | 0.4043 | 0.3752 | 20.392 | 0.5893 | 0.3636 | 25.236 | 0.8196 | 0.2618 |

trajectories with $N = 64$ child Gaussians each. In Stage I, we train the foundation model on eight datasets; in Stage II, we freeze it and train the decoder, first with key trajectories extracted from ground-truth trajectories on Kubric and then with those extracted from the foundation model. Full implementation details are given in Appendix B.1.

**Evaluation.** For novel-view synthesis, we report PSNR, SSIM, and LPIPS on DyCheck (Gao et al., 2022) over co-visible regions of 32-frame windows, on NVIDIA (Yoon et al., 2020) with 12 round-robin input views, and on the Kubric-4D (Greff et al., 2022) validation scenes of TokenGS (Ren et al., 2026). To measure the motion captured by feed-forward 4DGS, we evaluate Gaussian trajectories on DyCheck, Dynamic Replica (Karaev et al., 2023), and PointOdyssey (Zheng et al., 2023), reporting the endpoint error (EPE) of Gaussian displacements and the percentage $\delta_\tau$ of errors below a threshold $\tau$; on DyCheck, we adapt the protocol of Shape of Motion (Wang et al., 2025b) to sliding windows over consecutive annotated frames. We further evaluate 3D point tracking of our dense trajectories on PointOdyssey, Dynamic Replica, and PStudio (Joo et al., 2015) with APD and AJ, following Track4World (Lu et al., 2026). Detailed protocols, co-visibility masks, and pose alignment are described in Appendix B.3 and Appendix B.4.

### 4.2 Dynamic Novel View Synthesis

Tab. 1 compares rendering quality, Gaussian counts, and input-pose requirements for feed-forward reconstruction methods. The baselines are 4DGT (Xu et al., 2025), MoVieS (Lin et al., 2026a), NeoVerse (Yang et al., 2026), TokenGS, and C4G. 4DGT, MoVieS, and TokenGS require input camera poses, whereas NeoVerse, C4G, and our method do not. We report two variants: Ours (dense) renders the pixel-aligned Gaussians predicted by the foundation model, one per input pixel, and Ours (compact) is our complete method, which decodes $MN$ Gaussians with the query-based decoder. Additional DyCheck results at $stride = 1$ are reported in Tab. 4 in Appendix C.

**Rendering quality and representation size.** Our two variants rank first and second in PSNR and SSIM on all three benchmarks. Ours (compact) uses $65{,}536$ Gaussians, about $74\times$ fewer than MoVieS, NeoVerse, and Ours (dense) and $20\%$ fewer than TokenGS. C4G uses fewer primitives ($2{,}048$) but is worse than Ours (compact) in all nine metrics. Ours (dense) attains the lowest LPIPS on every benchmark, whereas Ours (compact) stays within $0.5$ dB PSNR of it with $74\times$ fewer Gaussians. The compact representation thus preserves PSNR and SSIM but leaves a perceptual gap relative to Ours (dense). Fig. 3 provides the qualitative comparison. Both our dense and compact methods generate high-quality novel view synthesis results.

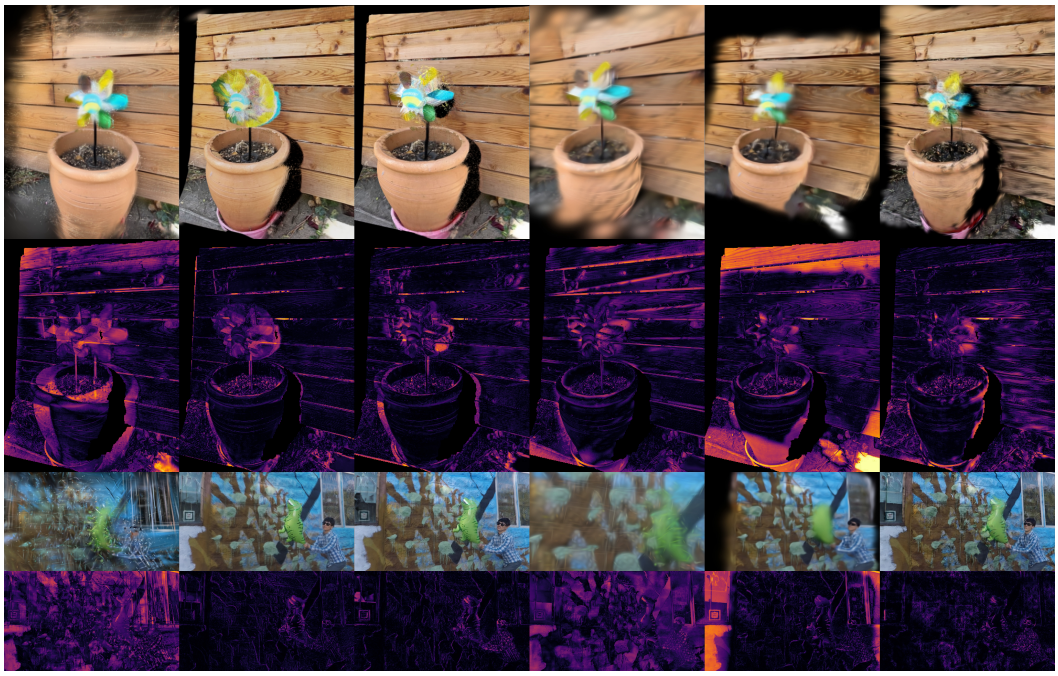


Figure 3: **Qualitative comparisons on dynamic novel-view synthesis.** Row 1 and 3 are novel view rendering results. Row 2 and 4 are colormapped error maps. Note that our method achieves sharper images while better resolving complex motion such as the rotating windmill. We recommend the reader to watch the *supplemental videos* for evaluating the visual results.

Table 2: **Quantitative comparison on dynamic Gaussian tracking.** Results on DyCheck, Dynamic Replica, and PointOdyssey. Lower EPE and higher $\delta$ are better; thresholds are 5 cm and 10 cm for DyCheck, and 0.05 and 0.1 units for Dynamic Replica and PointOdyssey.

| | | DyCheck | | | Dynamic Replica | | | PointOdyssey | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Type | Method | EPE↓ | $\delta_{5\text{cm}}$ ↑ | $\delta_{10\text{cm}}$ ↑ | EPE↓ | $\delta_{0.05}$ ↑ | $\delta_{0.1}$ ↑ | EPE↓ | $\delta_{0.05}$ ↑ | $\delta_{0.1}$ ↑ |
| Pixel-aligned | NeoVerse | 0.1330 | 37.75 | 63.72 | 0.0784 | 52.99 | 76.97 | 0.1088 | 56.94 | 79.23 |
| | MoVieS | 0.0931 | 45.45 | 69.32 | 0.0584 | 63.15 | 84.22 | 0.0686 | 68.07 | 86.20 |
| | Ours (dense) | 0.0834 | 47.27 | 73.72 | 0.0599 | 64.69 | 85.30 | 0.0640 | 73.32 | 88.79 |
| Query-based | C4G | 0.1808 | 26.96 | 45.26 | 0.2434 | 12.33 | 29.35 | 0.2388 | 14.09 | 34.00 |
| | C4G (motion trained) | 0.1768 | 27.62 | 48.75 | 0.1977 | 15.12 | 33.59 | 0.2270 | 14.74 | 34.87 |
| | TokenGS | 0.1643 | 32.46 | 52.59 | 0.2685 | 20.83 | 38.36 | 0.5921 | 15.66 | 27.64 |
| | TokenGS (motion trained) | 0.1507 | 35.24 | 59.91 | 0.1751 | 24.88 | 46.53 | 0.2827 | 18.53 | 33.62 |
| | Ours (compact) | 0.0838 | 49.08 | 72.82 | 0.0868 | 45.26 | 76.38 | 0.1006 | 52.70 | 79.37 |

### 4.3 Dynamic 3D Tracking

As shown in Tab. 2, Ours (compact) outperforms all query-based baselines, including their motion-trained variants, on every metric. On DyCheck, it reduces EPE by 49% relative to TokenGS, nearly matches Ours (dense) (0.0838 vs. 0.0834), and attains the highest $\delta_{5\text{cm}}$ among all methods, whereas MoVieS and Ours (dense) remain stronger on Dynamic Replica and PointOdyssey. Fig. 5 visualizes the predicted 3D trajectories and their image projections. For 3D point tracking, Ours (dense), which reads the dense trajectories of our foundation model, attains the highest average APD and AJ (Tab. 5 in Appendix C).

Table 3: **Ablation studies on Kubric.** Starting from Ours (compact) in the last row ($M = 1024$, $N = 64$), each variant replaces or removes one component of the trajectory parameterization, key trajectory extraction, Gaussian decoder, or motion supervision, and is trained with the same schedule. We report tracking and dynamic novel-view synthesis metrics, and highlight the best, second-best, and third-best results in each column.

| | | Tracking | | | Dynamic NVS | | |
|---|---|---|---|---|---|---|---|
| Component | Variant | EPE↓ | $\delta_{0.05}$ ↑ | $\delta_{0.1}$ ↑ | PSNR↑ | SSIM↑ | LPIPS↓ |
| Trajectory | w/o null-space constraint | 0.04117 | 72.44 | 91.82 | 21.315 | 0.6690 | 0.3974 |
| Key extraction | Random sampling | 0.04224 | 76.38 | 91.01 | 22.093 | 0.6673 | 0.3872 |
| | Grid downsampling | 0.04631 | 69.89 | 89.84 | 21.146 | 0.6585 | 0.4057 |
| | w/o color criterion | 0.03686 | 80.63 | 93.27 | 22.746 | 0.6749 | 0.3808 |
| Decoder | w/o anchor offsets | 0.16825 | 18.98 | 36.01 | 20.640 | 0.6296 | 0.4845 |
| | w/o ray–anchor cross-attention | 0.03674 | 79.94 | 94.41 | 21.886 | 0.6423 | 0.4197 |
| Training | Rasterized-flow loss | 0.03817 | 75.88 | 92.99 | 22.689 | 0.6753 | 0.3825 |
| Ours (compact) | Full | 0.03600 | 81.25 | 94.07 | 23.038 | 0.6811 | 0.3718 |

### 4.4 Ablation Studies

We examine how trajectory prediction, key trajectory extraction, Gaussian decoding, motion supervision, and output size contribute to tracking and rendering for our query-based method on the Kubric-4D split of TokenGS (Ren et al., 2026). All variants follow the same schedule: the decoder is first warmed up for 5K steps with key trajectories extracted from ground-truth trajectories, and each variant is then trained separately for 30K steps with key trajectories predicted by the foundation model. We evaluate tracking with EPE, $\delta_{0.05}$, and $\delta_{0.1}$, and rendering with PSNR, SSIM, and LPIPS, using Ours (compact) with $M = 1024$ and $N = 64$ as the reference.

As shown in Tab. 3, every component of our design contributes to both tracking and rendering. Removing the null-space constraint (Eq. 2) as in TraceAnything, replacing grouping-based key trajectory extraction with random or grid sampling, or supervising motion with Shape-of-Motion-style rasterized flow instead of trajectories all degrade both, and dropping the color criterion from grouping causes smaller but consistent drops. Without the null-space constraint, PSNR drops by 1.7 dB and EPE rises by $14\%$, confirming the benefits for both rendering and tracking. Directly predicting dense Gaussian trajectories as in TokenGS, instead of offsets relative to key trajectories (Eq. 4), causes the largest degradation, while removing ray–anchor cross-attention mainly hurts rendering.

We further vary the number of key trajectories $M$ from 1K to 8K with $N = 64$, and the number of child Gaussians per trajectory $N$ from 2 to 64 with $M$ =1K (Fig. 4 in Appendix C). More tokens mainly improve rendering (+1.0 dB PSNR with nearly constant EPE), whereas more child Gaussians mainly improve motion (20% lower EPE with only +0.3 dB PSNR).

## 5 Conclusion

We presented a trajectory-guided framework for feed-forward 4D Gaussian reconstruction from monocular video that treats the correspondence a dynamic scene requires and the primitives its rendering requires as separate budgets. A null-space B-spline anchors each trajectory to its source point, local grouping over 3D proximity, relative motion, and appearance extracts explicit key trajectories, and these trajectories serve as a motion scaffold for Gaussian queries, so the rendering budget no longer inherits the redundancy of dense observations. Because the scaffold persists across the sequence, the same compact representation supports both novel view synthesis and 3D tracking without a task-specific tracking module. Experiments show that, with up to $74\times$ fewer Gaussians than pixel-aligned methods, our compact model achieves the highest PSNR and SSIM among all baselines and outperforms every query-based baseline on all Gaussian-tracking metrics. Our approach remains limited to two-stage training for this compact representation. In future work, we will extend to jointly integrate the two-stage training and processing for long video sequences.

### AI USE STATEMENT

We used generative AI tools to assist with literature searches, manuscript drafting and revision, the organization and presentation of the method and experiments, method implementation, supporting qualitative and thematic data analysis and interpretation of results. The authors take responsibility for the final manuscript, including its technical claims, citations, and reported results.

## REFERENCES

Matteo Balice, Yanik Kunzi, Chenyangguang Zhang, Matteo Matteucci, Marc Pollefeys, and Sungwhan Hong. No pose, no problem in 4D: Feed-Forward dynamic gaussians from unposed Multi-View videos. In *Advances in Neural Information Processing Systems*, 2026.

Alessandro Burzio, Tobias Fischer, Sven Elflein, Qunjie Zhou, Riccardo de Lutio, Jiawei Ren, Jiahui Huang, Shengyu Huang, Marc Pollefeys, Laura Leal-Taixé, Zan Gojcic, and Haithem Turki. Déjà view: Looping transformers for multi-view 3d reconstruction. In *Advances in Neural Information Processing Systems*, 2026.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *arXiv preprint arXiv:2001.10773*, 2020.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Tiancheng Chen, Sheng Tang, Wenhua Jin, Weiqi Zhang, Juntong Fang, Junsheng Zhou, and Zesong Li. UniQuery4R: unified 4D scene reconstruction from a single query. *arXiv preprint arXiv:2608.17283*, 2026.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: efficient 3D gaussian splatting from sparse Multi-View images. In *European Conference on Computer Vision*, 2024.

Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4RTrack: simultaneous 4D reconstruction and tracking in the world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. In *Advances in Neural Information Processing Systems*, 2022.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti Derek Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

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*, 2024.

Junhwa Hur, Charles Herrmann, Songyou Peng, Philipp Henzler, Zeyu Ma, Todd Zickler, and Deqing Sun. UFO-4D: unposed feedforward 4D reconstruction from two images. In *International Conference on Learning Representations*, 2026.

Minsik Jeon, Jay Karhade, Deva Ramanan, and Shubham Tulsiani. Point4D: long-range 4D motion reconstruction. *arXiv preprint arXiv:2609.09145*, 2026.

Yanqin Jiang, Tengfei Wang, Zhengwei Wang, Chenjie Cao, Junta Wu, Wenhan Luo, Weiming Hu, Jin Gao, and Chunchao Guo. OmniX: any-view and any-time 4D reconstruction via feed-forward trajectory fields. In *European Conference on Computer Vision*, 2026.

Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4D: Learning how things move in 3D from internet stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

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*, pp. 3334–3342, 2015.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. DynamicStereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024.

Jay Karhade, Nikhil Keetha, Yuchen Zhang, Tanisha Gupta, Akash Sharma, Sebastian Scherer, and Deva Ramanan. Any4D: unified Feed-Forward metric 4D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota BulÃ², Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: universal Feed-Forward metric 3D reconstruction. In *International Conference on 3D Vision (3DV)*, 2026.

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

Mungyeom Kim, Minkyeong Jeon, Honggyu An, Jaewoo Jung, Hyuna Ko, Jisang Han, Hyeonseo Yu, Donghwan Shin, Sunghwan Hong, Takuya Narihira, Kazumi Fukuda, Yuki Mitsufuji, and Seungryong Kim. Learning global motion with compact gaussians for feed-forward 4D reconstruction. *arXiv preprint arXiv:2605.31595*, 2026.

Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, João Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. TAPVid-3D: A benchmark for tracking any point in 3D. In *Advances in Neural Information Processing Systems*, 2024.

Agelos Kratimenos, Jiahui Lei, and Kostas Daniilidis. DynMF: Neural motion factorization for real-time dynamic view synthesis with 3D gaussian splatting. In *European Conference on Computer Vision*, 2024.

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*, 2025.

Jiahui Lei, Yijia Weng, Adam W. Harley, Leonidas Guibas, and Kostas Daniilidis. MoSca: Dynamic gaussian fusion from casual videos via 4D motion scaffolds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6165–6177, 2025.

Wenyu Li, Sidun Liu, Tongrui Hu, Peng Qiao, and Yong Dou. LocusGS: Spatially grounded tokens for feed-forward 3D gaussian splatting. *arXiv preprint arXiv:2608.12825*, 2026a.

Yinglong Li, Donghui Shen, Xiaoyu Zhang, Zhichao Ye, Hongyu Wu, Aimin Hao, Guofeng Zhang, and Haomin Liu. QuerySplat: Decoupling geometry and appearance representations in 3DGS prediction. *arXiv preprint arXiv:2608.01186*, 2026b.

Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. MatrixCity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

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*, 2024.

Hanxue Liang, Jiawei Ren, Ashkan Mirzaei, Antonio Torralba, Ziwei Liu, Igor Gilitschenski, Sanja Fidler, Cengiz Oztireli, Huan Ling, Zan Gojcic, and Jiahui Huang. Feed-forward bullet-time reconstruction of dynamic scenes from monocular videos. In *Advances in Neural Information Processing Systems*, 2025a.

Yiming Liang, Tianhan Xu, and Yuta Kikuchi. HiMoR: Monocular deformable gaussian reconstruction with hierarchical motion representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025b.

Yiqing Liang, Abhishek Badki, Hang Su, James Tompkin, and Orazio Gallo. Zero-shot monocular scene flow estimation in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025c.

Chenguo Lin, Yuchen Lin, Panwang Pan, Yifan Yu, Tao Hu, Honglei Yan, Katerina Fragkiadaki, and Yadong Mu. MoVieS: Motion-aware 4D dynamic view synthesis in one second. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a.

Chieh Hubert Lin, Zhaoyang Lv, Songyin Wu, Zhen Xu, Thu Nguyen-Phuoc, Hung-Yu Tseng, Julian Straub, Numair Khan, Lei Xiao, Ming-Hsuan Yang, Yuheng Ren, Richard Newcombe, Zhao Dong, and Zhengqin Li. DGS-LRM: Real-time deformable 3D gaussian reconstruction from monocular videos. In *Advances in Neural Information Processing Systems*, 2025.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations*, 2026b.

Shing Ho J. Lin, Wenzhao Zheng, Dong Zhuo, Yuqi Wu, Jie Zhou, and Jiwen Lu. SM4RT: learning structured motion geometry for 4D reconstruction. *arXiv preprint arXiv:2607.22534*, 2026c.

Yehonathan Litman, Xiaoxuan Ma, Manan Shah, Nicolas Ugrinovic, Kris Kitani, Fernando De la Torre, and Shubham Tulsiani. Lift4D: harmonizing Single-View 3D estimation for 4D reconstruction In-the-Wild. In *SIGGRAPH Asia*, 2026.

Qingming Liu, Yuan Liu, Jiepeng Wang, Xianqiang Lyv, Peng Wang, Wenping Wang, and Junhui Hou. MoDGS: Dynamic gaussian splatting from casually-captured monocular videos with depth priors. In *International Conference on Learning Representations*, 2025.

Xinhang Liu, Yuxi Xiao, Donny Y. Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace Anything: Representing any video in 4D via trajectory fields. In *International Conference on Learning Representations*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Jiahao Lu, Jiayi Xu, Wenbo Hu, Ruijie Zhu, Chengfeng Zhao, Sai-Kit Yeung, Ying Shan, and Yuan Liu. Track4world: Feedforward world-centric dense 3d tracking of all pixels. *arXiv preprint arXiv:2603.02573*, 2026.

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.

Yihang Luo, Shangchen Zhou, Yushi Lan, Xingang Pan, and Chen Change Loy. 4RC: 4D reconstruction via conditional querying anytime and anywhere. In *International Conference on Machine Learning*, 2026.

Lars Mescheder, Wei Dong, Shiwei Li, Xuyang Bai, Marcel Santos, Peiyun Hu, Bruno Lecouat, Mingmin Zhen, Amaël Delaunoy, Tian Fang, Yanghai Tsin, Stephan R. Richter, and Vladlen Koltun. Sharp monocular view synthesis in less than a second. *arXiv preprint arXiv:2512.10685*, 2025.

Jisu Nam, Jahyeok Koo, Soowon Son, Jaewoo Jung, Honggyu An, Junhwa Hur, and Seungryong Kim. TrackCraft3R: repurposing video diffusion transformers for dense 3D tracking. In *Advances in Neural Information Processing Systems*, 2026.

Tuan Duc Ngo, Peiye Zhuang, Chuang Gan, Evangelos Kalogerakis, Sergey Tulyakov, Hsin-Ying Lee, and Chaoyang Wang. DELTA: dense efficient long-range 3D tracking for any video. In *International Conference on Learning Representations*, 2025.

NVIDIA. Instant NuRec: Feed-forward 3D gaussian reconstruction for driving scene simulation. *arXiv preprint arXiv:2607.14203*, 2026.

Kazuki Ozeki, Shun Kenney, Yuto Shibata, Eisuke Takeuchi, Takuya Narihira, Kazumi Fukuda, Ryosuke Sawata, Yuki Mitsufuji, and Yoshimitsu Aoki. 4D reconstruction from sparse dynamic cameras. *arXiv preprint arXiv:2606.04593*, 2026.

Jongmin Park, Minh-Quan Viet Bui, Juan Luis Gonzalez Bello, Jaeho Moon, Jihyong Oh, and Munchurl Kim. SplineGS: robust Motion-Adaptive spline for Real-Time dynamic 3D gaussians from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021a.

Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics*, 40(6), 2021b.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Jiawei Ren, Michal Jan Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3D gaussian prediction from pixels with learnable tokens. *arXiv preprint arXiv:2604.15239*, 2026.

Colton Stearns, Adam W. 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.

Edgar Sucar, Zihang Lai, Eldar Insafutdinov, and Andrea Vedaldi. Dynamic point maps: A versatile representation for dynamic 3D reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Edgar Sucar, Eldar Insafutdinov, Zihang Lai, and Andrea Vedaldi. V-DPM: 4D video reconstruction with dynamic point maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Allen Tu, Haiyang Ying, Alex Hanson, Yonghan Lee, Tom Goldstein, and Matthias Zwicker. 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.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-$\Omega$. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a.

Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of Motion: 4D reconstruction from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025b.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3D perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Wenguan Wang, Jianbing Shen, Jianwen Xie, and Fatih Porikli. Super-Trajectory for video segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1671–1679, 2017.

Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. TartanAir: A dataset to push the limits of visual SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. $\pi^3$: Permutation-Equivariant visual geometry learning. In *International Conference on Learning Representations*, 2026b.

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*, 2024.

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. SpatialTracker: tracking any 2D pixels in 3D space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. SpatialTrackerV2: 3D point tracking made easy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Zhen Xu, Zhengqin Li, Zhao Dong, Xiaowei Zhou, Richard Newcombe, and Zhaoyang Lv. 4DGT: Learning a 4D gaussian transformer using real-world monocular videos. *arXiv preprint arXiv:2506.08015*, 2025.

Jiawei Yang, Jiahui Huang, Boris Ivanovic, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, Danfei Xu, Yue Wang, and Marco Pavone. STORM: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *International Conference on Learning Representations*, 2025.

Yuxue Yang, Lue Fan, Ziqi Shi, Junran Peng, Feng Wang, and Zhaoxiang Zhang. NeoVerse: Enhancing 4D world model with in-the-wild monocular videos. *arXiv preprint arXiv:2601.00393*, 2026.

Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4D gaussian splatting. In *International Conference on Learning Representations*, 2024a.

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

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. In *International Conference on Learning Representations*, 2025.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Bowei Zhang, Lei Ke, Adam W. Harley, and Katerina Fragkiadaki. TAPIP3D: Tracking any point in persistent 3D geometry. In *Advances in Neural Information Processing Systems*, 2025a.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K. Barral, Raia Hadsell, Zoubin Ghahramani, Andrew Zisserman, Junlin Zhang, and Mehdi S. M. Sajjadi. Efficiently reconstructing dynamic scenes one d4rt at a time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. MonST3R: a simple approach for estimating geometry in the presence of motion. In *International Conference on Learning Representations*, 2025b.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large reconstruction model for 3D gaussian splatting. In *European Conference on Computer Vision*, 2024.

Songyan Zhang, Yongtao Ge, Jinyuan Tian, Guangkai Xu, Hao Chen, Chen Lv, and Chunhua Shen. POMATO: marrying pointmap matching with temporal motions for dynamic 3D reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025c.

Xuankai Zhang, Junjin Xiao, Shangwei Huang, Wei shi Zheng, and Qing Zhang. Learning explicit continuous motion representation for dynamic gaussian splatting from monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026b.

Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. PointOdyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Liyuan Zhu, Shengyu Huang, Amrita Mazumdar, Tianye Li, Zan Gojcic, Gordon Wetzstein, Iro Armeni, Shalini De Mello, and Alex Trevithick. World from motion: Generative dynamic gaussian reconstruction from monocular video. *arXiv preprint arXiv:2607.01202*, 2026a.

Ruijie Zhu, Jiahao Lu, Wenbo Hu, Xiaoguang Han, Jianfei Cai, Ying Shan, and Chuanxia Zheng. MotionCrafter: dense geometry and motion reconstruction with a 4D VAE. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026b.

# A GEOMETRY-ANCHORED B-SPLINE TRAJECTORY

## A.1 PROBLEM AND CONSTRAINT

Trace Anything predicts a trajectory from absolute controls $\mathbf{P} \in \mathbb{R}^{K\times 3}$ as $\mathbf{X}(t) = \mathbf{b}(t)\mathbf{P}$, where $\mathbf{b}(t) \in \mathbb{R}^{1\times K}$ is the row of cubic B-spline basis functions. Every control contributes to the source position $\mathbf{X}(t_i) = \mathbf{b}(t_i)\mathbf{P}$, so fitting motion moves it.

We instead take the point $\mathbf{p}_i$ predicted at time $t_i$ from the geometry backbone, treat it as a fixed anchor, and let the spline produce displacements. Setting $\mathbf{O} = \mathbf{P} - \mathbf{1}_K\mathbf{p}_i$ and using partition of unity, $\mathbf{b}(t)\mathbf{1}_K = 1$, rewrites the curve exactly:

$$\mathbf{X}(t) = \mathbf{b}(t)(\mathbf{1}_K\mathbf{p}_i + \mathbf{O}) = \mathbf{p}_i + \mathbf{b}(t)\mathbf{O}. \tag{9}$$

Anchoring $\mathbf{X}(t)$ at $\mathbf{p}_i$ at time $t_i$ poses the following linear constraint,

$$\mathbf{X}(t_i) = \mathbf{p}_i \iff \mathbf{r}_i\mathbf{O} = 0 \tag{10}$$

where $\mathbf{r}_i := \mathbf{b}(t_i)$. Note that this constraint should hold true for any choices of $\mathbf{O}$.

## A.2 STRUCTURE OF THE OFFSETS

Equation 10 is a single linear condition on $\mathbf{O}$, so this effectively reduce the degree of freedom of control points from $K$ to $K-1$. We expose exactly those degrees of freedom by factoring

$$\mathbf{O} = \mathbf{N}_i\mathbf{\Theta}, \tag{11}$$

where the reduced coefficients $\mathbf{\Theta} \in \mathbb{R}^{(K-1)\times 3}$ is predicted by a neural network, while $\mathbf{N}_i \in \mathbb{R}^{K\times(K-1)}$ has a closed form fixed by the source time. Substituting into Equation 9,

$$\mathbf{X}(t) = \mathbf{p}_i + \mathbf{b}(t)\mathbf{N}_i\mathbf{\Theta}, \tag{12}$$

The constraint in Eq. 10 becomes $\mathbf{r}_i\mathbf{N}_i = 0$.

## A.3 THE CONSTRUCTION OF $\mathbf{N}_i$

We construct $\mathbf{N}_i$ from the full singular value decomposition of $\mathbf{r}_i = \mathbf{b}(t_i)$:

$$\mathbf{r}_i = \mathbf{U}_i\mathbf{\Sigma}_i\mathbf{V}_i^\top, \qquad \mathbf{N}_i = [\mathbf{v}_{i,2}, \ldots, \mathbf{v}_{i,K}], \tag{13}$$

where $\mathbf{v}_{i,k}$ is the $k$th column of $\mathbf{V}_i$. Since the B-spline basis sums to one, $\mathbf{r}_i \neq 0$ and has rank one. The remaining $K-1$ right singular vectors therefore form an orthonormal basis of its null space, giving $\mathbf{r}_i\mathbf{N}_i = 0$ and $\mathbf{N}_i^\top\mathbf{N}_i = \mathbf{I}_{K-1}$. Thus $\mathbf{O} = \mathbf{N}_i\mathbf{\Theta}$ satisfies the anchor constraint for any predicted $\mathbf{\Theta}$ and can represent every offset satisfying that constraint.

The basis depends only on the source time and knot vector, so we compute it once per source frame, cache it, and share it across all pixels. We use the same basis convention during training and inference because the predicted coefficients are expressed in that basis.

# B Additional Implementation Details

## B.1 Implementation Details

Our foundation model builds on Depth Anything 3 (DA3) (Lin et al., 2026b). Based on the DA3 backbone encoder, a geometry head predicts depth and camera parameters, and a spline head predicts per-pixel motion as a cubic B-spline with $K = 7$ control points, parameterized by $K - 1 = 6$ free offset coefficients (Eq. 2). The query-based Gaussian decoder pairs each of the $M$ key trajectories with a learnable Gaussian token and decodes $N$ child Gaussians per token, yielding $M \times N$ Gaussians in total. We use $M = 1024$ and $N = 64$ by default, with degree-zero spherical harmonics for color.

Training follows the two-stage procedure in Sect. 3.5. In Stage I, we train the foundation model on the training splits of eight datasets: Kubric-4D (Greff et al., 2022), Dynamic Replica (Karaev et al., 2023), PointOdyssey (Zheng et al., 2023), Stereo4D (Jin et al., 2025), MatrixCity (Li et al., 2023), TartanAir (Wang et al., 2020), Virtual KITTI 2 (Cabon et al., 2020), and ScanNet++ (Yeshwanth et al., 2023). Each training clip contains 24 to 32 frames with a temporal stride of 1 or 2, at a resolution of $252 \times 252$ ($252 \times 448$ for PointOdyssey and Dynamic Replica). We initialize from DA3-Giant, fine-tune the last 27 of the 40 encoder blocks, keep the depth head and camera decoder frozen, and train the spline head from scratch. We use AdamW (Loshchilov & Hutter, 2019) with a learning rate of $10^{-4}$ ($10^{-5}$ for the pretrained encoder), a cosine schedule, and a batch size of 8 on 8 NVIDIA H100 GPUs for 100K iterations.

In Stage II, we freeze the foundation model and train the query-based Gaussian decoder, including its Gaussian tokens, in two phases. We first warm up the query-based Gaussian decoder on the Kubric dataset with $10K$ steps, using anchors extracted from the ground-truth dense trajectories with the same key trajectory extraction as described in (Sect. 3.3). We then jointly train the query-based Gaussian decoder with the key trajectories extracted from the foundation model, and continue training with the same datasets used in Stage I with 100K steps.

## B.2 Motion Supervision

**Stage I trajectory loss.** For each valid temporal pair $(s, t)$ of a tracked point, let $e_{s,t} = \|[\widehat{\mathbf{x}}(t) - \widehat{\mathbf{x}}(s)] - [\mathbf{x}^*(t) - \mathbf{x}^*(s)]\|_2$ and $c_{s,t} = \min(c_s, c_t)$. All valid pairs are used, not only adjacent times, and the confidence-weighted loss follows Track4World (Lu et al., 2026):

$$\lambda_t \mathcal{L}_{\text{traj}} = 100\langle c_{s,t} e_{s,t}\rangle + 10\langle -0.2 \log c_{s,t}\rangle, \tag{14}$$

where $\langle\cdot\rangle$ denotes the mean over valid pairs.

**Stage II motion loss.** For a supervised track with source pixel $p_s$, we associate it with the child Gaussians that render $p_s$ at the source time using their normalized, detached compositing contributions, and penalize the displacement error of each contributing Gaussian:

$$\bar{w}_k = \text{stopgrad}\left(\frac{\alpha_k(p_s)T_k(p_s)}{\sum_j \alpha_j(p_s)T_j(p_s)}\right), \qquad e = \sum_k \bar{w}_k \, \ell(\boldsymbol{\mu}_k(t) - \boldsymbol{\mu}_k(s) - \Delta\mathbf{x}^*), \tag{15}$$

where $\ell(\mathbf{v}) = \|\mathbf{v}\|_1/3$ and $\Delta\mathbf{x}^*$ is the ground-truth displacement. The loss averages $e$ separately over dynamic and static tracks with equal weights. The rasterized-flow alternative in our ablation instead blends the displacements before the comparison, $\ell(\sum_k \bar{w}_k \Delta\boldsymbol{\mu}_k - \Delta\mathbf{x}^*)$, which allows the errors of different Gaussians to cancel. We ramp $\lambda_d$ to 1 and $\lambda_m$ to 0.1 over the first 1,000 and 2,000 updates, respectively.

## B.3 Evaluation Settings

**Novel View Synthesis (NVS).** We evaluate on DyCheck (Gao et al., 2022), NVIDIA (Yoon et al., 2020), and Kubric-4D (Greff et al., 2022) using PSNR, SSIM, and LPIPS (Tab. 1). For DyCheck, we use moving-camera inputs and synchronized novel views, with metrics computed over co-visible regions for 32-frame sequences at stride=2. We slide a non-overlapping 32-frame window over the entire sequence of each scene, reconstruct each window independently, and average the metrics over

all target frames of each scene. We provide the details of the co-visibility and pose alignment in Appendix B.4. For NVIDIA, 12 input views are sampled in a round-robin style, where the $i$-th frame comes from the $i$-th camera at timestep $i$, and the first camera at the other 11 timesteps serves as the target. We estimate the cameras of the input frames with DA3 (Lin et al., 2026b) for pose-conditioned baselines. Kubric-4D based on the validation datasets from TokenGS (Ren et al., 2026), we use 24 consecutive frames for each method and evaluate the rendering on the same novel-view camera to evaluate the dynamic NVS performance.

**Gaussian Trajectories.** We evaluate the center trajectories of the reconstructed Gaussians to measure the actual motion captured by feed-forward 4DGS methods. Each query point is associated with Gaussians at its source frame, and we compare the displacement of their centers with the ground-truth displacement of the point. We report the endpoint error (EPE) of the displacement and $\delta_\tau$, the percentage of errors below a threshold $\tau$, on DyCheck (Gao et al., 2022), Dynamic Replica (Karaev et al., 2023), and PointOdyssey (Zheng et al., 2023). On DyCheck, we build on the evaluation of Shape of Motion (Wang et al., 2025b), which reconstructs each full video and tracks annotated keypoints, lifted to 3D with LiDAR depth and calibrated cameras, between all pairs of annotated frames. Since feed-forward methods process a limited number of frames at a time, we instead use a sliding window for each pair of consecutive annotated frames, starting at the source frame and covering the target frame. Each window evaluates only the displacement between its source and target frames, measured in meters with $\tau \in \{5\,\mathrm{cm}, 10\,\mathrm{cm}\}$. On Dynamic Replica and PointOdyssey, we use the first 32 frames of 50 and 55 test clips, respectively. We query ground-truth points visible in the first frame, and report EPE and $\delta_\tau$ with $\tau \in \{0.05, 0.1\}$ in normalized units.

**3D Point Tracking.** We further evaluate the absolute 3D positions of predicted point tracks on PointOdyssey, Dynamic Replica, and Panoptic Studio (PStudio) (Joo et al., 2015), following the world-coordinate evaluation of Track4World (Lu et al., 2026). We use the first 16 frames of 50 clips per dataset and query ground-truth points visible in the first frame. Predicted tracks are aligned to the ground truth with a global scale and translation, followed by the median scaling of TAPVid-3D (Koppula et al., 2024). We report APD, the fraction of visible points within distance thresholds, and AJ, which additionally accounts for predicted visibility, both averaged over thresholds on a 0–1 scale (Tab. 5 in Appendix C).

### B.4 Evaluation Details

**Co-visibility.** Novel-view metrics are computed only on target pixels that are observed in the input views. The official DyCheck masks (Gao et al., 2022) mark pixels co-visible with the entire training video, which does not fit our short input windows, in which many of these pixels are never observed. We therefore compute window-specific masks by depth reprojection, similar to the masks used for novel-view supervision in SHARP (Mescheder et al., 2025) and for novel-view evaluation in Let-CamsGo (Ozeki et al., 2026). On DyCheck, the input is the handheld camera and the targets are the two synchronized static cameras. For each input timestamp, we unproject the ground-truth LiDAR depth of the input camera, project it into each target camera with the ground-truth calibration, and mark the covered pixels after a fixed morphological densification. The mask of a window is the temporal union of these same-time masks, so a target pixel is evaluated if it is covered at least once within the window, analogous to the valid region of World from Motion (Zhu et al., 2026a). On Kubric, the same-time co-visibility mask between the input and target cameras is computed from ground-truth depth, and dynamic-region metrics further intersect it with the ground-truth dynamic mask. NVIDIA is evaluated on full images. All masks depend only on ground truth and are shared by all methods; PSNR, SSIM, and LPIPS are computed within the masks, and all methods render on a black background on DyCheck and Kubric.

**Pose Alignment.** Methods that require input poses receive ground-truth cameras on DyCheck and Kubric and DA3-estimated cameras on NVIDIA, and render the target views in the same coordinate frame. Methods without input poses reconstruct the scene in their own coordinate frame and scale, so we first map the ground-truth target cameras into that frame with a similarity transform estimated from the input views only, anchored at the first input camera and scaled by the input camera baseline or, when the input camera is static, by the input-view depth. On NVIDIA, the target camera coincides with the first input camera, so its pose is taken directly from the predicted input camera.

Table 4: **Quantitative Comparison on DyCheck (stride=1).** These results complement the DyCheck ($stride = 2$) evaluation in Tab. 1. Posed denotes required input camera poses. We highlight the best, second-best, and third-best results in each metric column.

| | | | DyCheck (stride=1) | | |
|---|---|---|---|---|---|
| Method | Posed | #Primitives | PSNR↑ | SSIM↑ | LPIPS↓ |
| 4DGT | Yes | 544,320 | 11.385 | 0.2924 | 0.5524 |
| MoVieS | Yes | 4,816,896 | 13.599 | 0.3449 | 0.3504 |
| NeoVerse | No | 4,816,896 | 12.149 | 0.2584 | 0.4652 |
| C4G | No | 2,048 | 13.525 | 0.3402 | 0.6139 |
| TokenGS | Yes | 81,920 | 10.551 | 0.3070 | 0.5854 |
| Ours (dense) | No | 4,816,896 | 14.620 | 0.4191 | 0.2857 |
| Ours (compact) | No | 65,536 | 14.738 | 0.3912 | 0.3630 |

On DyCheck, we further refine the target camera poses at test time with the reconstruction frozen, the standard protocol for methods without input poses, used by MoSca (Lei et al., 2025) and by the feed-forward methods NeoVerse (Yang et al., 2026) and C4G (Kim et al., 2026); we keep this protocol for consistency with them. Unlike MoSca, which optimizes a separate pose for every test frame, we optimize a single pose for each static target camera over 1,000 steps, since the DyCheck test cameras are static and per-frame optimization trades camera accuracy for better scores in dynamic regions (Zhu et al., 2026a).

# C Additional Results

## C.1 Quantitative Comparison on Dynamic Novel-View Synthesis

Tab. 4 reports additional DyCheck results at $stride = 1$. Ours (compact) achieves the highest PSNR (14.738 dB), while Ours (dense) achieves the highest SSIM (0.4191) and lowest LPIPS (0.2857). Using about $74\times$ fewer Gaussians, Ours (compact) slightly exceeds Ours (dense) in PSNR but has lower SSIM and higher LPIPS. Both variants outperform the external baselines in PSNR and SSIM, consistent with the $stride = 2$ results in the main table. Additional results are provided in the supplementary material.

## C.2 Quantitative Comparison on 3D Point Tracking

Tab. 5 evaluates 3D point tracking with Ours (dense), whose tracks are the dense trajectories predicted by our foundation model before key trajectory extraction, following the protocol in Sect. B.3. Ours (dense) attains the highest average APD (0.5868) and AJ (0.4821), ahead of MoVieS (Lin et al., 2026a) (0.5831 and 0.4787) and Track4World (Lu et al., 2026) (0.5638 and 0.4682). It performs best on Dynamic Replica and PStudio, while MoVieS is stronger on PointOdyssey. It also outperforms TraceAnything (Liu et al., 2026) (0.4801 APD and 0.3665 AJ), which uses the same B-spline trajectory representation with absolute control points; because this comparison also involves different backbones and training data, Sect. 4.4 compares the two parameterizations directly.

## C.3 Representation Budget

Fig. 4 reports the effect of the number of key trajectories $M$ and child Gaussians per trajectory $N$ on Kubric, complementing the ablations in Sect. 4.4.

## C.4 Qualitative Comparison on 3D Point Tracking

Fig. 5 compares sparse and dense 3D points and their trajectories. The first two rows show sparse points and tracks from compact representations, while the last two rows show dense points and tracks. These examples illustrate the dense temporal correspondence captured by our method.

Table 5: **Quantitative Comparison on 3D Point Tracking.** APD and AJ (↑) on PointOdyssey, Dynamic Replica, PStudio (Panoptic Studio), and the average metrics.

| | PointOdyssey | | Dynamic Replica | | PStudio | | Average | |
|---|---|---|---|---|---|---|---|---|
| Method | APD↑ | AJ↑ | APD↑ | AJ↑ | APD↑ | AJ↑ | APD↑ | AJ↑ |
| St4RTrack | 0.5067 | 0.3647 | 0.5223 | 0.4404 | 0.6023 | 0.5069 | 0.5438 | 0.4373 |
| SpatialTrackerV2 | 0.5339 | 0.3946 | 0.5166 | 0.4499 | 0.6089 | 0.4918 | 0.5531 | 0.4454 |
| TraceAnything | 0.4135 | 0.2730 | 0.4330 | 0.3309 | 0.5938 | 0.4957 | 0.4801 | 0.3665 |
| Track4World | 0.5387 | 0.3965 | 0.5288 | 0.4655 | 0.6238 | 0.5426 | 0.5638 | 0.4682 |
| MoVieS | 0.5648 | 0.4170 | 0.5305 | 0.4570 | 0.6539 | 0.5622 | 0.5831 | 0.4787 |
| Ours (dense) | 0.5592 | 0.4133 | 0.5470 | 0.4699 | 0.6541 | 0.5631 | 0.5868 | 0.4821 |

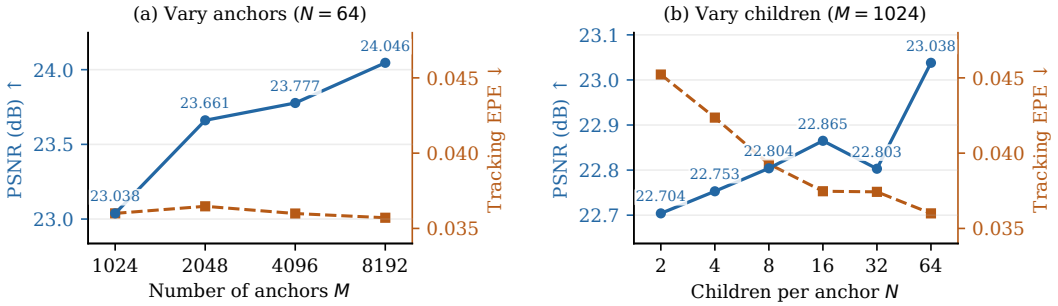


Figure 4: **Effect of the representation budget on Kubric.** PSNR (blue, left axis) and tracking EPE (orange, right axis) when varying (a) the number of anchors $M$ with $N = 64$ and (b) the number of children per anchor $N$ with $M = 1024$.

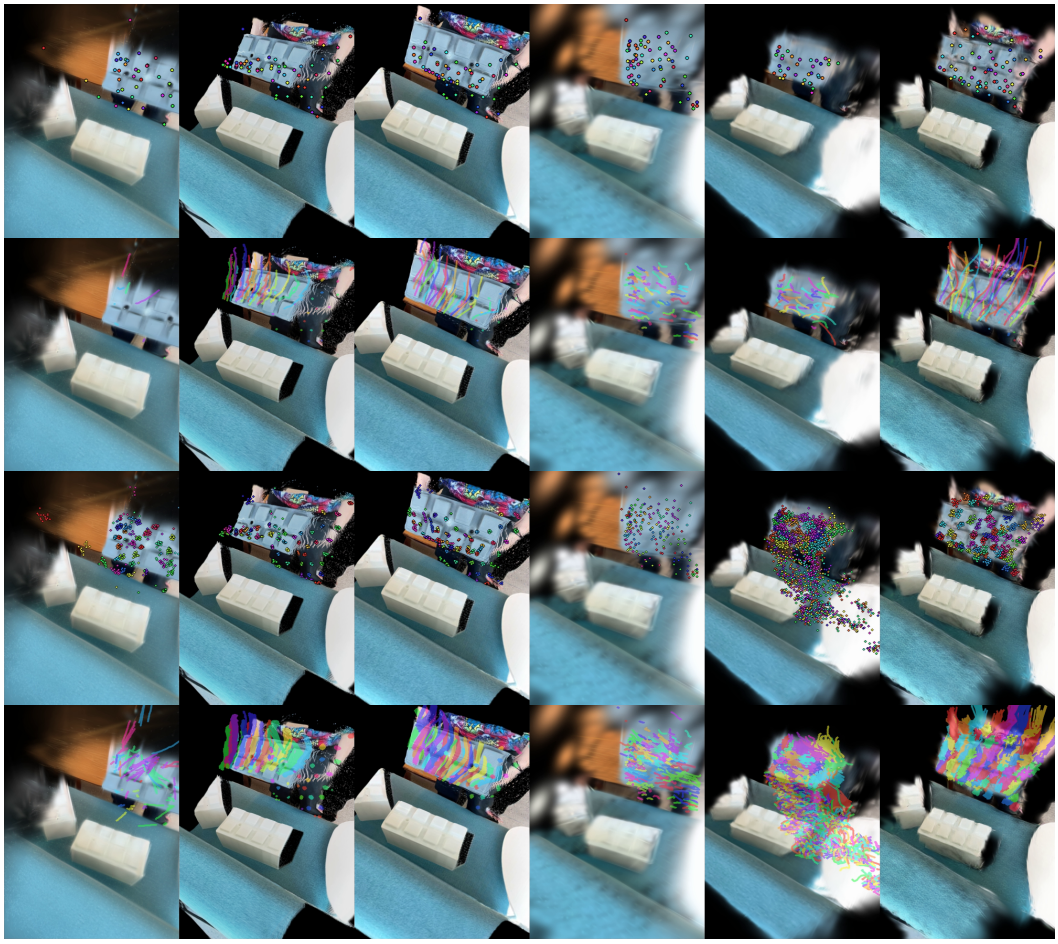


Figure 5: **Qualitative comparisons on dense 3D tracking.** Row 1 and 2 show sparse (compact) points and tracks. Row 3 and 4 visualize dense points and tracks. Note that our method achieve high-quality dense correspondence. We recommend the reader to watch the *supplemental videos* for evaluating the visual results.