

RayPE: Ray-Space Positional Encoding for 3D-Aware Video Generation

Project Page: <https://raype-project.github.io>

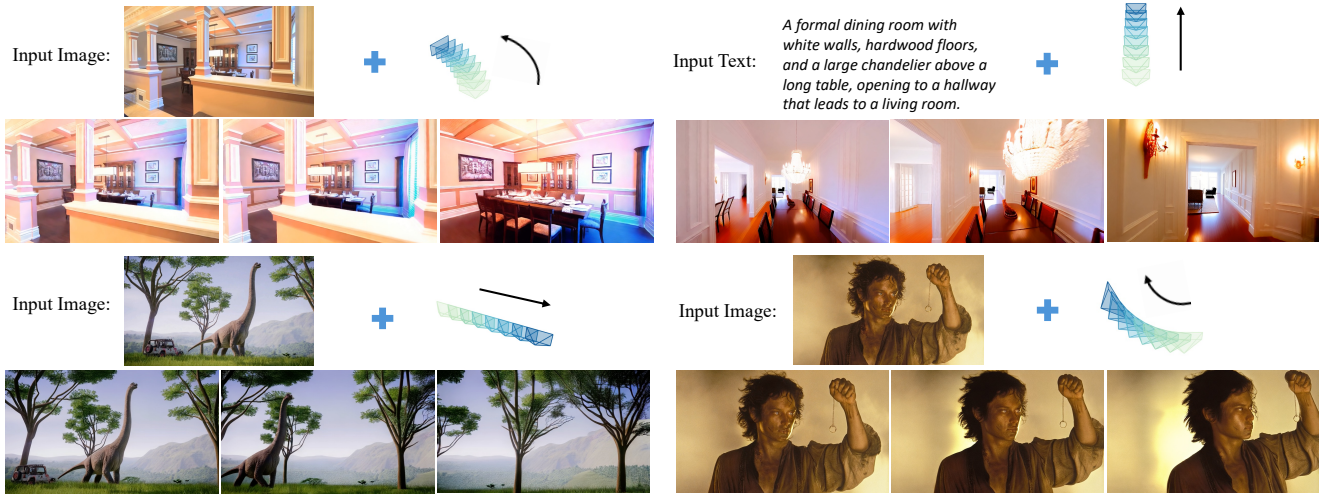


Figure 1. RayPE enables precise relative camera control for pretrained video diffusion models. Given a target camera trajectory, our method generates videos that faithfully follow the trajectory while preserving the base model’s generation quality. Top: results on image-to-video (left) and text-to-video (right). Bottom: out-of-distribution generalization to movie stills.

Abstract

Modern video diffusion transformers position their tokens through RoPE on the (u, v, t) axes — a description of the camera’s sampling grid that says nothing about the 3D structure of the scene. We observe that the geometric relation between two camera rays is captured by the Plücker reciprocal product, which is bilinear in the two rays — the same algebraic form as the dot product in Transformer attention. Building on this analogy, we propose RayPE, a positional-encoding extension that injects per-token 6D Plücker coordinates additively into the queries and keys of self-attention, with a query/key flip arrangement under which the symmetric identity configuration coincides exactly with the reciprocal product. The injection is additive, the resulting attention score decomposes into a content term, a geometry term, and two content \leftrightarrow geometry cross-terms — all of which our experiments find individually necessary. To make the encoding stable across video data with heterogeneous camera-translation scales (SfM, deep SLAM, metric), we further decouple ray direction from moment magnitude, gate the encoding by a learned function of the log-magnitude, and apply RMSNorm to align it with the QKNorm-normalized content branch. The full module adds less than 0.1% parameters to a pretrained video

DiT, is zero-initialized to start from the pretrained weights, and improves camera controllability, cross-frame 3D consistency, and overall video quality on a four-dataset training mixture.

1. Introduction

Video diffusion transformers have become the dominant backbone for large-scale text-to-video and image-to-video generation [5, 14, 16, 24, 32, 37]. These models patchify each frame into a grid of latent tokens and inject positional information through a factorized RoPE on the (u, v, t) index [12, 28]. This positional encoding describes the sampling grid of the camera: it tells the model where each token sits in the 2D lattice and along the temporal axis, but says nothing about where in 3D the corresponding ray points. Two tokens that observe the same physical surface from different viewpoints carry no positional relationship, and the model must recover their geometric correspondence entirely from pixel content.

Recent camera-conditioned video models [3, 10, 33, 36, 38, 39] address this gap by injecting camera pose through adapter modules, ControlNet encoders, rendered point-cloud images, or cross-attention streams. These approaches treat geometry as an auxiliary feature processed outside

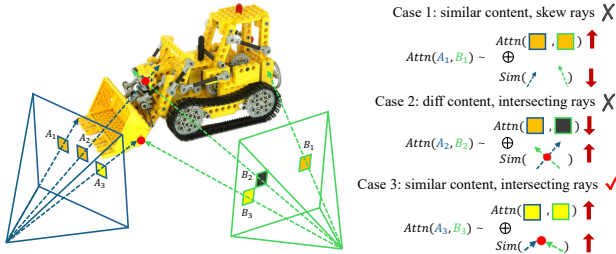


Figure 2. Why attention needs ray geometry. Two cameras (blue, green) each cast three rays through the tokens $A_{1,2,3}$ and $B_{1,2,3}$ on their image planes; red dots mark where two rays meet in 3D. Whether a pair (A_*, B_*) should attend factors as content similarity and ray-geometric similarity. **Case 1** (similar content, skew rays) and **Case 2** (different content, intersecting rays) are both misleading; only **Case 3** (both agree) is a genuine correspondence. RayPE realizes this product inside the attention dot product by injecting Plücker rays into Q, K (Sec. 4.1).

of self-attention, while the attention dot product remains purely content-based. A parallel line of work [17, 18, 42] attempts to build camera information into positional encodings, but does so multiplicatively by replacing or splitting the original RoPE, which disrupts the pretrained positional structure and employs reduced parameterizations that do not exploit the algebraic structure of attention.

We approach the problem from a different angle. A video pixel can be identified with the camera ray that produced it, parameterized as a 6D Plücker coordinate. The geometric relationship between two rays is captured by the Plücker reciprocal product (the Klein form on line space), which is bilinear in the two rays, $SE(3)$ -invariant, and vanishes exactly when the rays are coplanar (i.e., observe the same 3D point). Transformer attention computes a dot product between queries and keys, which is also a bilinear form. These two share the same algebraic shape. We ask: can the geometric relationship between two camera rays be made to live *inside* the attention dot product, so that the model receives 3D information through the same channel that already carries positional and content signals?

We answer affirmatively with RayPE (*Ray-Space Positional Encoding*). For each token we compute its 6D Plücker coordinate from the camera pose and intrinsics, project it through a small learned head, and add the result to the query and key vectors of self-attention. The injection is additive and zero-initialized, leaving the pretrained 3D RoPE intact at the start of training. The resulting attention score decomposes into a content term, a geometry term, and two content \leftrightarrow geometry cross-terms; our ablations show all three components are individually necessary.

Two practical issues must be solved before this encoding works at scale. First, the Plücker moment scales linearly with camera translation, so datasets with different pose conventions (SfM [27], SLAM [6, 29], metric) produce wildly

different encoding magnitudes. Second, naively adding the geometric signal to the content branch does not control their relative magnitude. We address both through *Normalize-Gate-Inject* (NGI): we decouple the ray into a scale-invariant direction and a log-magnitude, gate the encoding by a learned function of the log-magnitude, and apply an RMSNorm [40] symmetric to the content-branch QKNorm [11]. The result is stable across heterogeneous data while preserving absolute distance information.

We integrate RayPE into Wan2.2-TI2V-5B [32], a 5B-parameter pretrained video diffusion transformer, and train on a mixture of four datasets (RealEstate10K [43], DL3DV [19], PanShot [42], and OmniWorld [44]). The encoding is zero-initialized so training starts exactly from the pretrained model, the original RoPE is preserved, and the total parameter increase is below 0.1%. Experimental results on camera controllability, cross-frame 3D consistency, and video generation quality are reported in Section 5. Our contributions center on two ideas and their practical realization:

- **Plücker rays as positional encoding.** The Plücker reciprocal product is bilinear in two rays — the same algebraic form as the Transformer attention score — so camera rays admit a PE-style injection inside the attention dot product, not around it.
- **RayPE.** An additive 6D Plücker injection on q and k that splits attention into a content term, a geometry term, and two content \leftrightarrow geometry cross-terms. Ablations show all three are individually necessary; the cross-terms are a coupling that V-side, cross-attention, AdaLN, and rotation-based ray PEs cannot structurally represent.
- **Normalize-Gate-Inject.** A decoupling, gating, and PE-side RMSNorm pipeline that stabilizes 6D Plücker injection across video data with heterogeneous pose-scale conventions (SfM, SLAM, metric).

2. Related Work

Video Diffusion Models. Video diffusion models have converged on a DiT backbone [23] with a latent-space training recipe [7, 26]. Representative models — Stable Video Diffusion [5], CogVideoX [37], HunyuanVideo [16], MovieGen [24], and Wan [32] — all patchify latent video frames into a 2D-by-time token grid and inject positional information through a factorized 3D RoPE on the (u, v, t) axes [12, 28]. This encodes *where* in the sampling grid a token lives, not *where* in 3D the corresponding ray points. Two tokens imaging the same surface from different frames receive unrelated positions, and 3D correspondence must be recovered from pixel content alone — motivating a ray-space positional signal inside attention.

Camera-Conditioned Video Generation. A growing body of work conditions video generation on explicit cam-

era trajectories, differing in how camera information enters the network. *Adapter-based methods* inject camera signals through modules external to self-attention: MotionCtrl [33] encodes extrinsics into per-frame tokens added to temporal attention; CameraCtrl [10] rasterizes per-pixel Plücker maps and fuses them via a ControlNet-style residual encoder; CamCo [36] concatenates Plücker embeddings at the input channel and adds epipolar-constrained cross-view attention; AC3D [1] and VD3D [2] attach trainable cross-attention adapters or LoRA modules conditioned on camera embeddings. *Rendering-conditioned methods* first build a 3D proxy and render it from the target trajectory as a structural condition: ViewCrafter [38] lifts the input to a point cloud and uses rendered views as pixel-aligned conditioning; TrajectoryCrafter [39] feeds both rendered point-cloud video and the source video as a dual-stream channel condition; DaS [8] uses 3D trajectories [21, 22, 34, 35, 41] as control signal; GS-DiT [4] replaces the 3D trajectories with tracked 3D Gaussians. *Implicit cross-view methods* such as ReCamMaster [3] concatenate source and target tokens along the frame dimension, letting the Transformer’s native attention implicitly build cross-view correspondences with a lightweight camera encoder. Despite their diversity, all these approaches treat camera geometry as *content* processed by auxiliary encoders, adapters, or rendering pipelines external to the attention score. The dot product remains purely content-based; cross-frame geometric correspondence must still be discovered implicitly. RayPE takes a different route: it places ray geometry directly inside queries and keys as a positional encoding, so the attention score itself carries geometric information without any adapter or rendering proxy.

Positional Encodings for Video Transformers.

RoPE [28] encodes position by a per-pair rotation; its multi-dimensional extension [12] is standard in video DiTs. Recent works generalize it into camera-aware encodings: PRoPE [18] replaces temporal RoPE with a multiplicative relative-projective rotation; UCPE [42] modulates RoPE frequencies via a spatial adapter encoding full camera geometry; ReRoPE [17] repurposes only the low-frequency temporal RoPE bands for relative camera-pose information. These designs share three limitations. First, they operate *multiplicatively* on q, k , requiring replacement or splitting of the original 3D RoPE — disrupting the pretrained positional structure. Second, they use reduced parameterizations (rotation angles, projective matrices) and none exploits the *bilinear* structure of the dot product to realize a geometric invariant. Third, they do not address *scale heterogeneity*: PRoPE and ReRoPE discard absolute translation by operating on relative rotations only; UCPE applies a fixed normalization that cannot adapt to per-clip scale variation. RayPE addresses all three: an additive injection

preserving the original RoPE, a canonical Q/K flip basis under which the symmetric configuration $E_q = E_k = I_6$ realizes the Klein form, and a direction/magnitude decoupling with a learned scale-aware gate for instance-adaptive robustness (Section 4).

3. Background and Notation

This section fixes the notation we use for video diffusion transformers, the camera model, and the Plücker representation of a ray, and recalls the geometric facts that motivate our design.

3.1. Video Diffusion Transformers

We follow the latent-video diffusion / flow-matching setup [5, 13, 20, 32, 37]. A video $V \in \mathbb{R}^{T \times 3 \times H \times W}$ is encoded by a 3D VAE into a latent volume $Z \in \mathbb{R}^{T_z \times C \times H_z \times W_z}$, patchified into a sequence of tokens $X \in \mathbb{R}^{N \times d}$ with $N = T_z \cdot H_p \cdot W_p$, where H_p, W_p are the post-patchification spatial sizes. Each token carries an integer index $(u, v, t) \in \{0, \dots, W_p - 1\} \times \{0, \dots, H_p - 1\} \times \{0, \dots, T_z - 1\}$.

A self-attention layer [31] with query/key/value projections $W_Q, W_K, W_V \in \mathbb{R}^{d \times d}$ produces, for each token i ,

$$q_i = W_Q x_i, \quad k_i = W_K x_i, \quad v_i = W_V x_i, \quad (1)$$

and computes the attention output $o_i = \sum_j \text{softmax}_j(\frac{1}{\sqrt{d}} \langle q_i, k_j \rangle) v_j$. Modern video DiTs add (i) RMS-based query/key normalization [11] (often called QKNorm) and (ii) a factorized 3D rotary positional encoding (RoPE) that applies a per-pair rotation to q and k as a function of (u, v, t) :

$$\tilde{q}_i = R(u_i, v_i, t_i) q_i, \quad \tilde{k}_j = R(u_j, v_j, t_j) k_j, \quad (2)$$

where $R(\cdot)$ is a block-diagonal rotation built from per-axis 1D RoPE [12, 28]. The rotation is applied so that the dot product $\langle \tilde{q}_i, \tilde{k}_j \rangle$ depends on $(u_i - u_j, v_i - v_j, t_i - t_j)$, giving the standard relative positional bias on the sampling lattice.

Crucially, (u, v, t) are indices into the camera’s sampling grid; they say nothing about *where* in 3D each token’s pixel originated. We do not change Eq. (2); RayPE adds an additive term that is computed from the 3D camera ray of each token.

3.2. Camera Model and Per-Token Rays

Each video clip is annotated with per-frame extrinsics $T_t \in \text{SE}(3)$ (camera-to-world) and per-frame intrinsics $K_t \in \mathbb{R}^{3 \times 3}$. We adopt the OpenCV convention (x -right, y -down, z -forward) throughout. For a token whose patch center has pixel coordinate (p_u, p_v) in frame t , the camera-frame ray direction is

$$\mathbf{d}_i^{\text{cam}} = \frac{K_t^{-1} [p_u, p_v, 1]^\top}{\|K_t^{-1} [p_u, p_v, 1]^\top\|}, \quad (3)$$

the world-frame ray direction is $\mathbf{d}_i = R_t \mathbf{d}_i^{\text{cam}}$ where R_t is the rotation part of T_t , and the world-frame ray origin is \mathbf{o}_i , the position of camera t . By construction $\|\mathbf{d}_i\| = 1$.

3.3. Plücker Decomposition Coordinates and Reciprocal Product

A 3D line (or ray) in \mathbb{R}^3 is identified by a pair of 3-vectors $(\mathbf{d}, \mathbf{m}) \in \mathbb{R}^6$, where \mathbf{d} is the direction and

$$\mathbf{m} = \mathbf{o} \times \mathbf{d} \quad (4)$$

is the Plücker moment, with \mathbf{o} any point on the line. Two constraints make (\mathbf{d}, \mathbf{m}) a parameterization of an oriented line up to scale: $\|\mathbf{d}\| > 0$ and $\mathbf{d} \cdot \mathbf{m} = 0$. Both are direct consequences of Eq. (4) [9, 25].

Given two rays $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{m}_i)$ and $\mathbf{r}_j = (\mathbf{d}_j, \mathbf{m}_j)$, their *Plücker reciprocal product* (also called Plücker inner product) is

$$\begin{aligned} \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} &\equiv \mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{d}_j \cdot \mathbf{m}_i \\ &= \mathbf{r}_i^\top J \mathbf{r}_j, \quad J = \begin{pmatrix} \mathbf{0}_3 & I_3 \\ I_3 & \mathbf{0}_3 \end{pmatrix}. \end{aligned} \quad (5)$$

Three properties of Eq. (5) are central to our design [9, 25]:

- **Bilinearity.** $\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}}$ is linear in \mathbf{r}_i and in \mathbf{r}_j separately; this is the same algebraic shape as the attention dot product $\langle q_i, k_j \rangle$, which is linear in q and k separately.
- **Coplanarity criterion.** $\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} = 0$ if and only if the two rays are coplanar; equivalently they intersect, are parallel, or coincide. For two rays that observe the *same* 3D point through two different cameras, the two rays meet at that point and so are coplanar, hence $\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} = 0$.
- **SE(3) invariance.** For $g \in \text{SE}(3)$ acting jointly on \mathbf{r}_i and \mathbf{r}_j , $\langle g\mathbf{r}_i, g\mathbf{r}_j \rangle_{\text{Pl}} = \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}}$; the reciprocal product depends only on the relative geometry of the two rays, not on the choice of world frame.

The bilinear shape of Eq. (5) and the bilinear shape of $\langle q, k \rangle$ are the algebraic match that RayPE exploits.

3.4. Scale Behavior of Plücker Coordinates

A subtlety of Plücker coordinates that becomes important in practice is their behavior under rescaling of the scene. Suppose all camera positions are rescaled by a factor $s > 0$: $\mathbf{o} \rightarrow s\mathbf{o}$. Then \mathbf{d} is unchanged (it is a unit direction) while $\mathbf{m} = \mathbf{o} \times \mathbf{d} \rightarrow s\mathbf{m}$. Substituting into Eq. (5) gives

$$\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} \rightarrow s \cdot \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}}, \quad (6)$$

so the reciprocal product scales linearly with translation magnitude. Different videos in the wild come with very different translation scales — normalized scene volumes, metric meters, or DROID-SLAM’s internal scale [29] — and

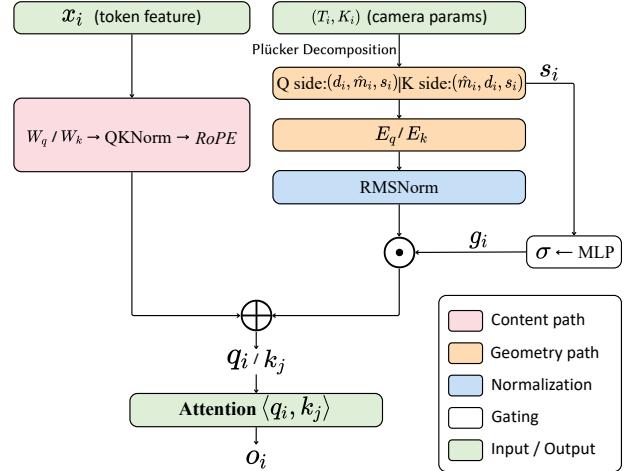


Figure 3. **Applying RayPE to a self-attention layer.** The content path (pink) preserves the pretrained $W_q/W_k \rightarrow \text{QKNorm} \rightarrow \text{RoPE}$ pipeline unchanged. The geometry path (orange/blue) computes per-token Plücker coordinates from camera parameters, decomposes them into scale-invariant direction and log-magnitude, projects via zero-initialized E_q/E_k , normalizes (RMSNorm), and gates by a learned function of the log-magnitude.

a naive raw-Plücker encoding would produce wildly different geometry-channel magnitudes across the dataset. Section 4.2 introduces our Normalize-Gate-Inject step that is designed to address exactly this issue.

4. Method

RayPE adds a ray-space positional term to the self-attention layers of a pretrained video diffusion transformer, as shown in Figure 3. The design follows three constraints from Section 1: preserving the original RoPE mechanism, injecting camera-ray geometry directly into the query–key dot product, and maintaining stable feature scaling across camera poses from diverse sources. We first introduce the core *Plücker Flip PE* (Section 4.1), followed by the *Normalize–Gate–Inject* (NGI) procedure for stable optimization (Section 4.2). Finally, we discuss the architectural design choices. (Section 4.3).

4.1. Plücker Flip Positional Encoding

The key observation is that the Plücker reciprocal product and the attention score have the same algebraic form. For two rays $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{m}_i)$ and $\mathbf{r}_j = (\mathbf{d}_j, \mathbf{m}_j)$,

$$\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} = \mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{d}_j \cdot \mathbf{m}_i$$

is bilinear in the two rays, while the attention score $\langle q_i, k_j \rangle$ is bilinear in query and key. This suggests adding ray-derived features to q and k in an order that makes the geometric reciprocal product appear as part of the attention score.

Each token i has a camera ray with Plücker coordinate $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{m}_i) \in \mathbb{R}^6$ (Section 3.2). We use two per-layer projections $E_q, E_k \in \mathbb{R}^{d \times 6}$ and add the projected ray features after the usual content projection and RoPE:

$$q_i = R(u_i, v_i, t_i) W_Q x_i + E_q(\mathbf{d}_i, \mathbf{m}_i), \quad (7)$$

$$k_j = R(u_j, v_j, t_j) W_K x_j + E_k(\mathbf{m}_j, \mathbf{d}_j). \quad (8)$$

Two choices matter here. The geometry term is additive, so the pretrained (u, v, t) RoPE remains unchanged. The Plücker halves are also flipped between the two sides: the query receives (\mathbf{d}, \mathbf{m}) , while the key receives (\mathbf{m}, \mathbf{d}) . We discuss the role of this flip below; this subsection uses the raw 6D coordinate to make the algebra explicit, and Section 4.2 gives the normalized 7D feature used in practice.

Attention score decomposition. Expanding $\langle q_i, k_j \rangle$ using Eqs. (7)–(8) yields four terms:

$$\begin{aligned} \langle q_i, k_j \rangle = & \underbrace{\langle \tilde{W}_Q x_i, \tilde{W}_K x_j \rangle}_{\text{(A) content}} + \underbrace{\langle \tilde{W}_Q x_i, E_k \tilde{\mathbf{r}}_j \rangle}_{\text{(B) content} \rightarrow \text{geom}} \\ & + \underbrace{\langle E_q \mathbf{r}_i, \tilde{W}_K x_j \rangle}_{\text{(C) geom} \rightarrow \text{content}} + \underbrace{\langle E_q \mathbf{r}_i, E_k \tilde{\mathbf{r}}_j \rangle}_{\text{(D) geometry}}, \end{aligned} \quad (9)$$

where $\tilde{\mathbf{r}}_j = (\mathbf{m}_j, \mathbf{d}_j)$ is the flipped coordinate of token j . Term (A) is the original content attention. Terms (B) and (C) couple appearance with ray geometry, so the model can learn when geometry should matter for a particular type of content. Term (D) is the geometry-only interaction:

$$\text{(D)} = \mathbf{r}_i^\top M \tilde{\mathbf{r}}_j, \quad M = E_q^\top E_k \in \mathbb{R}^{6 \times 6}. \quad (10)$$

Flip as a canonical basis. The role of the flip is best stated as a choice of basis on the geometric Q/K projection space rather than an algorithmic necessity:

Proposition 1. *With the flip arrangement, $E_q = E_k = I_6$ (so $M = I_6$) makes term (D) exactly the Plücker reciprocal product: $(D)|_{M=I_6} = \mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{m}_i \cdot \mathbf{d}_j = \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{P1}}$.*

(Proof in Appendix A.4.) For unconstrained learnable E_q, E_k , the flipped and unflipped parameterizations realize the same hypothesis class — any E_k in one form is equivalent to $E_k P$ in the other, where P swaps the (\mathbf{d}, \mathbf{m}) blocks (Appendix A.5). We do not claim the flip enlarges the function space the model can express. Its role is to choose a coordinate system in which the symmetric, geometrically meaningful configuration $E_q = E_k = I_6$ corresponds to the textbook Plücker reciprocal product, so that M admits a direct post-hoc reading: $\|M - I_6\|$ measures how far the learned bilinear bias has drifted from the ray-incidence form. The empirical gap between flipped and unflipped configurations is small (Section 5.3.2).

Parameter and compute cost. In the practical NGI version, each layer adds two $d \times 7$ Q/K projections, lightweight gates, and RMSNorms. The per-token projection cost remains $O(N \cdot d)$, which is small compared with the $O(N^2 d)$ cost of attention.

4.2. Normalize-Gate-Inject (NGI)

The flip above defines where geometry enters attention. To use it on mixed real-world data, we still need to control the scale of the injected signal. This is where NGI is used.

The first issue is **scale heterogeneity**. The moment $\mathbf{m} = \mathbf{o} \times \mathbf{d}$ scales linearly with the camera translation magnitude $\|\mathbf{o}\|$. Different datasets use different scale conventions, such as SfM-normalized scenes [27], DROID-SLAM [6, 29] internal scale, or metric meters. Even within one dataset, different baselines can produce moments with very different magnitudes. A raw projection $E\mathbf{r}$ would therefore mix a unit direction \mathbf{d} with a scale-dependent moment \mathbf{m} , making the geometry term unstable across batches.

The second issue is **magnitude imbalance** with the content branch. Modern video DiTs apply QKNorm (RMS normalization) to q and k before the dot product, giving the content branch a controlled magnitude. An unnormalized $E\mathbf{r}$ added on top does not respect this normalization. A single global scalar is also too coarse: it cannot handle both per-clip pose scale and the local balance between geometry and content.

NGI handles these two issues in three steps:

Step 1: Direction/magnitude decomposition. We decouple the Plücker coordinate into a scale-invariant directional part and a scalar log-magnitude:

$$\hat{\mathbf{m}} = \mathbf{m} / \max(\|\mathbf{m}\|, \epsilon), \quad s = \log \max(\|\mathbf{m}\|, \epsilon), \quad (11)$$

where ϵ is a small numerical constant. The 7D geometric input to E_q is $f_i^{(q)} = (\mathbf{d}_i, \hat{\mathbf{m}}_i, s_i)$, and to E_k (after the Q/K flip) is $f_j^{(k)} = (\hat{\mathbf{m}}_j, \mathbf{d}_j, s_j)$. The first six dimensions are invariant to a global rescaling of camera positions; the last dimension keeps the absolute distance information available as a separate scalar.

Step 2: Scale-aware gating. A small two-layer MLP G_s maps the per-token log-magnitude to a per-channel sigmoid gate:

$$g_i = \sigma(G_s(s_i)) \in (0, 1)^d, \quad (12)$$

with biases initialized so that $g \approx 0.5$ at $s = 0$. The gate lets the model adjust the strength of the geometry branch as a function of pose scale, rather than relying on one fixed weight for all clips.

Step 3: Normalize and inject. We project the 7D feature, apply a learnable RMSNorm analogous to the content-side QKNorm, and modulate the result by the gate:

$$pe_i^q = g_i \odot N_q(E_q f_i^{(q)}), \quad pe_j^k = g_j \odot N_k(E_k f_j^{(k)}), \quad (13)$$

where N_q, N_k are per-layer learnable RMSNorms. A scalar α , initialized to zero, controls the overall geometry/content balance:

$$q_i = \text{QKNorm}(\tilde{W}_Q x_i) + \alpha \cdot pe_i^q, \quad (14)$$

$$k_j = \text{QKNorm}(\tilde{W}_K x_j) + \alpha \cdot pe_j^k. \quad (15)$$

Since both branches are normalized, α has a consistent meaning across layers and training examples.

Log-scale augmentation. We also apply a simple scale augmentation during training. When computing the gate, we optionally perturb s by a clip-level offset:

$$\tilde{s}_i = s_i + \delta, \quad \delta \sim \text{Uniform}(-1.2, 1.6), \quad \text{with prob. } 0.3, \quad (16)$$

shared across all tokens of a clip. This simulates a global rescaling $\mathbf{o} \rightarrow e^\delta \mathbf{o}$ of the camera trajectory without changing the supervision. Only the gate receives \tilde{s}_i ; the projection $E_q f^{(q)}$ still uses the true s_i .

4.3. Architectural Discussion

Pseudocode. The full one-step procedure is given in Algorithm 1. The per-token Plücker computation, decomposition $(\mathbf{d}, \hat{\mathbf{m}}, s)$, scale gate, and Q/K flip are applied independently in each attention layer.

Relation to alternative camera-aware positional encodings. Unlike camera-aware positional encodings that replace or partition RoPE [17, 18, 42], RayPE preserves the pretrained 3D RoPE and injects geometric information as an additional Q/K bias term. Quantitative comparisons with these alternatives are provided in Section 5.

5. Experiments

We evaluate RayPE along three axes: (i) video generation quality, (ii) camera-trajectory controllability, and (iii) cross-frame geometric consistency. We compare RayPE with four camera-conditioned video generation baselines and four alternative camera-aware positional encoding designs on the same backbone, isolating the contribution of the proposed Plücker-flip/NGI formulation.

5.1. Setup

Backbone. For the main comparisons (Table 1), we evaluate our method on two official variants from Wan-2.2 [32]: the 5B TI2V model and the 14B I2V model. To ensure computational efficiency, all ablation studies are conducted

Algorithm 1 One forward-backward step of RayPE on top of a pretrained video DiT.

Require: Latent video z_0 , text embedding c , per-frame extrinsics $\{T_t\}$ and intrinsics $\{K_t\}$, dataset id ds , near-depth z_n (or \perp).

- 1: Apply trajectory rescaling: $\mathbf{o}_t \leftarrow (\eta_{ds} / \max(z_n, \epsilon)) \cdot \mathbf{o}_t$
▷ Appendix B.2–B.3
 - 2: Compute per-token rays $\{(\mathbf{d}_i, \mathbf{m}_i)\}$ from $\{T_t, K_t\}$ and the latent patch grid
▷ Section 3.2
 - 3: Sample noise $\epsilon \sim \mathcal{N}(0, I)$, timestep τ , build noisy latent z_τ
 - 4: **for** each DiT block $\ell = 1 \dots L$ **do**
 - 5: $q, k, v \leftarrow W_Q^\ell h, W_K^\ell h, W_V^\ell h$
▷ Pretrained projections, possibly fine-tuned
 - 6: $q, k \leftarrow \text{QKNorm}(q), \text{QKNorm}(k)$
 - 7: $q, k \leftarrow R(u, v, t) \cdot q, R(u, v, t) \cdot k$
▷ Standard 3D RoPE
 - 8: $f^{(q)} \leftarrow (\mathbf{d}, \hat{\mathbf{m}}, s), f^{(k)} \leftarrow (\hat{\mathbf{m}}, \mathbf{d}, s)$ where $s = \log \|\mathbf{m}\|$
 - 9: $g \leftarrow \sigma(G_s^\ell(\tilde{s}))$
▷ \tilde{s} optionally aug. per Eq. (16)
 - 10: $q \leftarrow q + \alpha \cdot g \odot N_q^\ell(E_q^\ell f^{(q)})$
 - 11: $k \leftarrow k + \alpha \cdot g \odot N_k^\ell(E_k^\ell f^{(k)})$
 - 12: $h \leftarrow h + \text{Attn}(q, k, v) W_O^\ell$
 - 13: $h \leftarrow h + \text{CrossAttn}(h, c) + \text{FFN}(h)$
 - 14: **end for**
 - 15: Compute diffusion / flow loss between predicted noise/velocity and target and backpropagate.
-

exclusively using the Wan-2.2-TI2V-5B model. Across all runs, videos are generated at 480×832 resolution with $T = 81$ frames ($T_z = 21$ latent frames), and the models are trained on the four-dataset mixture (Appendix B). All variants share their respective pretrained weights; only the newly integrated camera-conditioning module differs.

Baselines. We compare against four external methods: CameraCtrl [10] (Plücker maps via ControlNet encoder), ReCamMaster [3] (video-to-video re-rendering with camera control), UCPE [42] (full camera geometry via spatial-attention adapter), and ReRoPE [17] (low-frequency temporal RoPE replaced by relative camera pose). Where official checkpoints are unavailable we re-implement on our backbone under the same data and budget (Appendix A). We additionally train four *FreqSplit-RoPE* variants that replace the low-frequency half of temporal RoPE *multiplicatively* with (i) a 4×4 projective transform, (ii) camera-plane near/far UV coordinates, (iii) learned Plücker-to-phase mapping, or (iv) analytical (θ, ϕ, u, v) phases. All four share our backbone and budget; RayPE differs by adding Plücker features *additively* without modifying RoPE.

Evaluation. We evaluate on a held-out RE10K [43] split (image-to-video: first GT frame + GT trajectory) and re-

Method	Quality	Camera controllability				Distribution		
	CLIP \uparrow	RotErr \downarrow	TransErr \downarrow	CamMC \downarrow	ATE \downarrow	FVD \downarrow	FVD $_c$ \downarrow	FID \downarrow
<i>Wan-2.2 5B Scale</i>								
CameraCtrl [10]	25.04	0.152	1.292	1.355	1.501	824.37	805.62	64.08
ReCamMaster [3]	24.97	0.131	1.226	1.279	1.460	874.30	890.52	62.53
ReRoPE [17]	25.20	0.137	1.109	1.215	1.350	684.57	650.31	60.77
UCPE [42]	25.39	0.113	0.856	0.909	0.990	703.41	755.83	61.50
RayPE (ours)	26.05	0.085	0.751	0.802	0.884	543.17	588.62	57.83
<i>Wan-2.2 14B Scale</i>								
ReCamMaster [3]	25.31	0.109	0.976	1.012	0.995	675.23	697.10	59.21
ReRoPE [17]	25.85	0.114	0.820	0.905	0.859	493.30	525.61	49.18
UCPE [42]	25.72	0.082	0.693	0.760	0.788	529.42	558.70	54.75
RayPE (ours)	26.30	0.058	0.517	0.530	0.605	280.17	354.52	41.01

Table 1. Main comparison on the RE10K held-out split across different backbone scales. Higher is better for CLIP; lower is better for the rest. Pose metrics (RotErr, TransErr, CamMC, ATE) are computed on raw, unscaled ViPE trajectories estimated identically on both GT and generated videos (Section 5.1); rescaled-alignment results matching the conventional protocol are in Appendix A.

port eight metrics. *Quality*: **CLIP** \uparrow (inter-frame similarity). *Camera controllability*: **RotErr** \downarrow (geodesic rotation error), **TransErr** \downarrow (L2 translation error), **CamMC** \downarrow (per-frame pose Frobenius norm), **ATE** \downarrow (RMS absolute trajectory error), all on per-frame poses anchored to frame 0. *Distribution*: **FVD** \downarrow / **FVD** $_c$ \downarrow (full-frame / center-cropped Fréchet Video Distance [30]), **FID** \downarrow (per-frame Fréchet Inception Distance). Trajectories are estimated from both GT and generated videos using ViPE [15] and compared *without rescaling*. Unlike the common protocol of normalizing trajectories by their maximum translation magnitude, we preserve absolute-scale fidelity. This avoids artificially inflating the apparent controllability of methods that produce only small motions. Results under the rescaled evaluation protocol are provided in Appendix. All methods use 50 denoising steps with CFG scale 5.0 (text) / 1.0 (camera), matching Appendix B.4.

5.2. Main Comparison

Table 1 reports the main quantitative comparison and Figures 4–8 show qualitative results. Consistent with the design motivation in Sections 4.1 and 4.2, RayPE improves all four pose-error metrics substantially while also improving CLIP, FVD, FVD $_c$, and FID. We organize the qualitative evaluation along three progressively harder axes. *In-domain comparison* (Figure 4): on the RE10K validation split, we show the same clip generated by ReCamMaster, UCPE, and RayPE under identical camera conditioning; RayPE follows the target trajectory more faithfully across frames while the baselines drift. *In-domain gallery* (Figure 5): a broader set of RE10K scenes where each row is a different first frame driven by a distinct trajectory, illustrating consistent camera following across indoor walkthroughs, outdoor pans, and forward-moving corridors. *Out-of-distribution generalization* (Figures 6, 7, and 8): we test on inputs that never appear in the training data. Figure 6 compares baselines on artistic paintings, where RayPE realizes the intended cam-

era motion with correct parallax while baselines fail to follow the trajectory. Figure 7 extends this to a wider set of hand-painted stylized first frames, showing that RayPE preserves artistic style while accurately executing the prescribed camera path. Figure 8 fixes two cinematic movie-still first frames and drives each through several distinct trajectories, demonstrating that the geometric encoding generalizes across both content domain and trajectory diversity.

5.3. Ablations

We organize ablations into two complementary tables. The first contrasts RayPE against four alternative camera-aware PE designs that share our backbone (Section 5.3.1). The second removes one component at a time from RayPE, isolating the contribution of each design element (Section 5.3.2).

5.3.1. Camera-Aware PE Design Space

Table 2 compares RayPE against the four internal alternatives from Section 5.1, all trained under the same backbone, mixture and budget. The intent is to test two structural choices behind RayPE: (i) injecting camera geometry *additively* on q, k rather than multiplicatively replacing RoPE bands, and (ii) using the 6D Plücker coordinate rather than reduced angular or projective parameterizations of the ray. RayPE outperforms every FreqSplit-RoPE variant on every metric, indicating that the additive Plücker formulation is not interchangeable with the better-known multiplicative-RoPE family at our scale and training budget.

5.3.2. Component Ablation

Table 3 reports component ablations of the full RayPE configuration. Each row removes a single component while keeping all other settings and training recipes unchanged.

What each row tests. The **w/o Q/K flip** row sends (**d, m**) on both sides instead of the flipped pair; under



Figure 4. Qualitative comparison on RealEstate10K. Each row shows frames from the same clip generated under the same target camera trajectory. Top to bottom: ground truth, ReCamMaster [3], UCPE [42], and RayPE (ours). RayPE follows the target trajectory more faithfully across frames; ReCamMaster and UCPE drift from the camera path.



Figure 5. Gallery of RayPE on diverse scenes. Each row is a different first frame driven by a distinct target camera trajectory; frames go left to right along time, and the rightmost column shows the input trajectory as a stack of camera frusta.

arbitrary learnable E_q, E_k the flip is a reparametrization (Appendix A.5), so only a marginal change is expected, which the row confirms. **w/o Normalize-Gate-Inject** reduces the encoding to a raw 6D Plücker projection $E_{\mathbf{r}}$, removing direction/magnitude decomposition, scale gating, and PE RMSNorm together — the largest aggregate ablation of the normalize-gate-inject pipeline. **w/o PE RMSNorm** keeps decomposition and gating but drops the PE-side RMSNorm, so the geometry magnitude is no longer matched to the content branch and a single α cannot balance both. **w/o log-scale aug.** removes random log-magnitude augmentation during training, isolating the role of synthetic scale diversity in stabilizing cross-domain generalization; the modest drop suggests the augmentation helps

but is not the dominant ingredient. **w/o zero init** initializes E_q, E_k randomly instead of to identity (zero residual at $t = 0$), forcing the model to recover from a non-trivial perturbation of the pretrained attention; the larger FVD drop shows that breaking the pretrained equilibrium at the start of training is costly even when the final solution is ultimately reachable. **with V transform** additionally injects geometry into v , which consistently underperforms Q/K-only injection and confirms that ray geometry is most useful as an attention-bias positional signal rather than as a value-channel feature. The last two rows directly intervene on the attention decomposition of Eq. (9): **w/o (B)+(C) content \leftrightarrow geometry** keeps the content–content (A) and geometry–geometry (D) terms but suppresses the two



Figure 6. Out-of-distribution qualitative comparison. Inputs are artistic paintings (rather than natural photographs), conditioned on simple camera trajectories such as a forward dolly. No ground-truth video exists in this setting, so each row shows only the generated frames of a single method. RayPE realizes the intended camera motion (e.g., zoom-in with correct foreground/ background parallax) on stylized inputs, while ReCamMaster [3] and UCPE [42] fail to follow the trajectory.

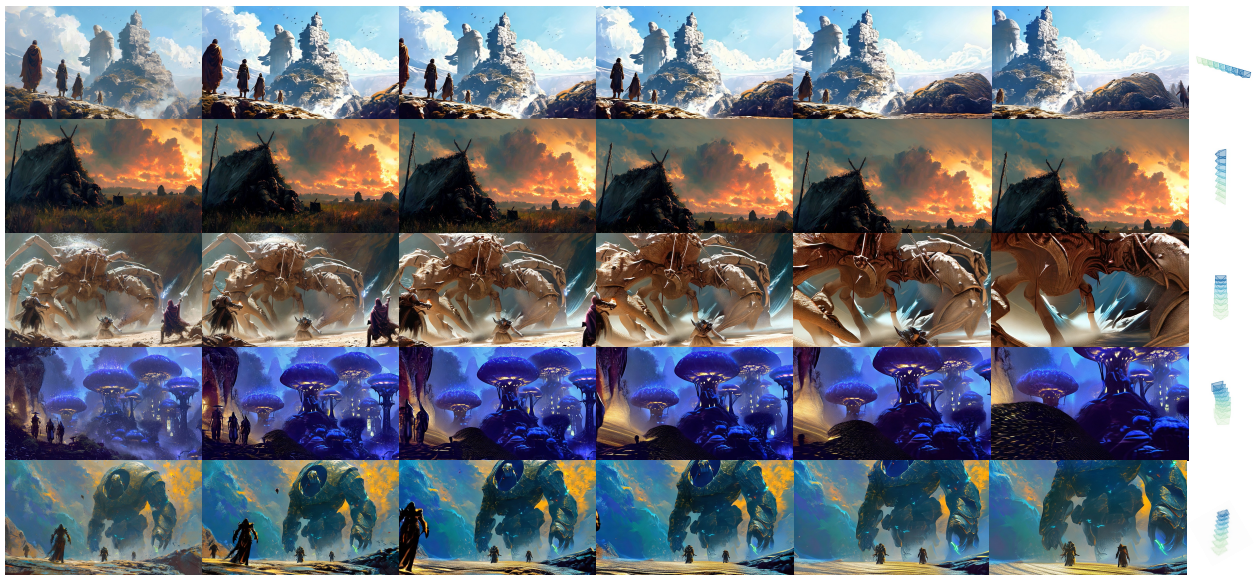


Figure 7. Generalization to stylized inputs. Each row is a different hand-painted concept-art first frame driven by a distinct target camera trajectory with RayPE; frames go left to right along time, and the rightmost column shows the input trajectory as a stack of camera frusta.



Figure 8. Multi-trajectory gallery on cinematic out-of-distribution inputs. We fix two movie-still first frames and drive each through several distinct camera trajectories (one trajectory per row within a scene), all generated by RayPE. Across both scenes, the generated videos execute the prescribed motion while preserving the cinematic style and layout of the first frame.

cross-modal terms that couple appearance with ray geometry, while **w/o (D) geometry** \leftrightarrow **geometry** removes the pure

geometric prior. Both produce sharp drops on every metric.

PE design	Quality	Camera controllability				Distribution		
	CLIP \uparrow	RotErr \downarrow	TransErr \downarrow	CamMC \downarrow	ATE \downarrow	FVD \downarrow	FVD $_c$ \downarrow	FID \downarrow
FreqSplit-RoPE + 4 \times 4 Proj.	25.09	0.175	1.227	1.308	1.584	675.03	723.90	62.98
FreqSplit-RoPE + Camera-UV	25.39	0.141	1.205	1.394	1.602	685.20	720.11	60.74
FreqSplit-RoPE + Plücker-RoPE	25.61	0.159	1.018	1.323	1.447	671.45	659.03	59.31
FreqSplit-RoPE + 4DOF Plücker	25.52	0.137	0.976	1.228	1.419	638.31	645.29	58.54
RayPE (ours)	26.05	0.085	0.751	0.802	0.884	543.17	588.62	57.83

Table 2. Camera-aware PE design space on RE10K. All rows share the Wan2.2-Ti2V-5B backbone, the four-dataset training mixture and the same training budget; the only difference is how the camera enters attention. RayPE adds an *additive* 6D Plücker term to q, k , while the four FreqSplit-RoPE variants *multiplicatively replace* the low-frequency half of the temporal RoPE.

Variant	CLIP \uparrow	RotErr \downarrow	CamMC \downarrow	ATE \downarrow	FVD \downarrow
Full RayPE	26.05	0.085	0.802	0.884	543.17
w/o Q/K flip	25.93	0.091	0.817	0.929	579.83
w/o Normalize-Gate-Inject	25.86	0.086	0.865	0.933	560.37
w/o PE RMSNorm	25.62	0.090	0.874	0.913	601.39
w/o log-scale aug.	25.94	0.083	0.823	0.890	557.92
w/o zero init	25.68	0.086	0.837	0.899	635.42
with V transform	25.57	0.090	0.855	0.949	724.10
w/o (B)+(C) content \leftrightarrow geometry	25.21	0.135	1.174	1.358	695.41
w/o (D) geometry \leftrightarrow geometry	24.08	0.159	1.382	1.440	733.08

Table 3. Component ablation of RayPE on RE10K. “Full” is the configuration used for the main comparison. Each subsequent row removes a single component while keeping the rest unchanged.

Cross-term and geometry-term necessity. The (B)+(C) and (D) interventions carry the strongest empirical signal in the table. Term (D) — Klein-form matching between two rays — is the geometric prior we explicitly designed for, so its necessity is expected; the larger drop on (D) confirms that the geometric matching cannot be substituted by content-only or cross- only signals. The (B)+(C) drop is more telling: these terms let the content of one token attend to (or be attended by) the geometry of another (e.g. “a sky-content token preferring upward-pointing rays”, or “a foreground-content token preferring nearby rays”), a coupling that V-side, cross-attention, AdaLN, and rotation-based ray PEs cannot structurally express. The size of the drop indicates that this content \leftrightarrow geometry coupling — not just the pure geometric prior in (D) — is a substantial part of what Plücker Flip PE provides.

5.3.3. Data composition

Table 4 ablates the effect of training-data composition. We compare training on RE10K alone, RE10K+DL3DV, RE10K+DL3DV+PanShot, and the full RE10K+DL3DV+PanShot+OmniWorld mixture. This highlights the value of *normalize-gate-inject* as more scale-heterogeneous data is added: while methods lacking scale-invariance may degrade as the trajectory-scale distribution widens, RayPE consistently benefits from the additional data.

Training mixture	CLIP \uparrow	RotErr \downarrow	CamMC \downarrow	ATE \downarrow	FVD \downarrow
RE10K only	25.58	0.091	0.897	0.983	586.45
RE10K + DL3DV	25.65	0.089	0.872	0.960	602.13
+ PanShot	25.93	0.083	0.836	0.905	550.19
+ OmniWorld (full)	26.05	0.085	0.802	0.884	543.17

Table 4. Effect of training-data composition on RE10K-test metrics.

6. Conclusion

We presented RayPE, a ray-space positional encoding for video diffusion transformers that exploits the algebraic match between the Plücker reciprocal product and the attention dot product. By adding per-token Plücker coordinates to queries and keys with a flip arrangement, RayPE makes the attention score contain a geometry–geometry term that equals the Klein form at initialization, giving the model a built-in 3D inductive bias before any learning. Normalize-Gate-Inject further decouples ray direction from translation magnitude and gates the encoding adaptively, making it stable across datasets with heterogeneous pose scales. The full module adds less than 0.1% parameters to a pretrained 5B video DiT, preserves the original RoPE, and requires no adapter or architectural change. Quantitative and qualitative results show that RayPE improves camera controllability and cross-frame 3D consistency while preserving the generation quality of the pretrained backbone.

References

- [1] Sherwin Bahmani, Ivan Skorokhodov, Guocheng Qian, Aliaksandr Siarohin, Willi Menapace, Andrea Tagliasacchi, David B Lindell, and Sergey Tulyakov. Ac3d: Analyzing and improving 3d camera control in video diffusion transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22875–22889, 2025. 3
- [2] Sherwin Bahmani, Ivan Skorokhodov, Aliaksandr Siarohin, Willi Menapace, Guocheng Qian, Michael Vasilkovsky, Hsin-Ying Lee, Chaoyang Wang, Jiayu Zou, Andrea Tagliasacchi, et al. Vd3d: Taming large video diffusion transformers for 3d camera control. In *International Conference on Learning Representations*, pages 66712–66737, 2025. 3

- [3] Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, and Di Zhang. ReCamMaster: Camera-controlled generative rendering from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2503.11647. 1, 3, 6, 7, 8, 9, 14
- [4] Weikang Bian, Zhaoyang Huang, Xiaoyu Shi, Yijin Li, Fuyun Wang, and Hongsheng Li. GS-DiT: Advancing video generation with pseudo 4D gaussian fields through efficient dense 3D point tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. arXiv:2501.02690. 3
- [5] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 1, 2, 3
- [6] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 2, 5
- [7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 2
- [8] Zekai Gu, Rui Yan, Jiahao Lu, Peng Li, Zhiyang Dou, Chenyang Si, Zhen Dong, Qifeng Liu, Cheng Lin, Ziwei Liu, et al. Diffusion as shader: 3d-aware video diffusion for versatile video generation control. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–12, 2025. 3
- [9] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004. 4, 13
- [10] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for video diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. 1, 3, 6, 7
- [11] Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020. 2, 3
- [12] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoon Yun. Rotary position embedding for vision transformer. In *European Conference on Computer Vision (ECCV)*, 2024. 1, 2, 3
- [13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3
- [14] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1
- [15] Jiahui Huang, Qunjie Zhou, Hesam Rabeti, Aleksandr Korovko, Huan Ling, Xuanchi Ren, Tianchang Shen, Jun Gao, Dmitry Slepichev, Chen-Hsuan Lin, et al. Vipe: Video pose engine for 3d geometric perception. *arXiv preprint arXiv:2508.10934*, 2025. 7
- [16] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, et al. Hunyuan-Video: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. 1, 2
- [17] Chunyang Li, Yuanbo Yang, Jiahao Shao, Hongyu Zhou, Katja Schwarz, and Yiyi Liao. ReRoPE: Repurposing RoPE for relative camera control. *arXiv preprint arXiv:2602.08068*, 2026. 2, 3, 6, 7, 14
- [18] Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as relative positional encoding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Introduces PROPE (Projective Positional Encoding) for multi-view transformers; arXiv:2507.10496. 2, 3, 6
- [19] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. DI3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2, 14
- [20] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. 3
- [21] Jiahao Lu, Weitao Xiong, Jiacheng Deng, Peng Li, Tianyu Huang, Zhiyang Dou, Cheng Lin, Sai-Kit Yeung, and Yuan Liu. Trackingworld: World-centric monocular 3d tracking of almost all pixels. *arXiv preprint arXiv:2512.08358*, 2025. 3
- [22] 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. 3
- [23] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2
- [24] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, et al. Movie Gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024. 1, 2
- [25] Helmut Pottmann and Johannes Wallner. *Computational Line Geometry*. Springer, 2001. 4, 13
- [26] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [27] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 5, 14
- [28] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced trans-

- former with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. 1, 2, 3
- [29] Zachary Teed and Jia Deng. DROID-SLAM: Deep visual SLAM for monocular, stereo, and RGB-D cameras. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 4, 5, 14
- [30] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. FVD: A new metric for video generation. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2019. 7
- [31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 3
- [32] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 1, 2, 3, 6, 15
- [33] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. MotionCtrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 1, 3
- [34] 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*, pages 20406–20417, 2024. 3
- [35] 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. *arXiv preprint arXiv:2507.12462*, 2025. 3
- [36] Dejjia Xu, Weili Nie, Chao Liu, Sifei Liu, Jan Kautz, Zhangyang Wang, and Arash Vahdat. CamCo: Camera-controllable 3D-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024. 1, 3
- [37] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, 2025. 1, 2, 3
- [38] Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhiber Huang, Xiang Gao, Xiaogang Luo, Ying Shan, and Yonghong Tian. ViewCrafter: Taming video diffusion models for high-fidelity novel view synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2409.02048. 1, 3
- [39] Wangbo Yu, Wenbo Hu, Jinbo Xing, and Ying Shan. TrajectoryCrafter: Redirecting camera trajectory for monocular videos via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 1, 3
- [40] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019. 2
- [41] BOWEI Zhang, Lei Ke, Adam Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *Advances in Neural Information Processing Systems*, 38: 135284–135303, 2026. 3
- [42] Cheng Zhang, Boying Li, Meng Wei, Yan-Pei Cao, Camilo Cruz Gambardella, Dinh Phung, and Jianfei Cai. Unified camera positional encoding for controlled video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. arXiv:2512.07237. 2, 3, 6, 7, 8, 9, 14
- [43] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *ACM SIGGRAPH*, 2018. 2, 6, 14
- [44] Yang Zhou, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Haoyu Guo, Zizun Li, Kaijing Ma, Xinyue Li, Yating Wang, Haoyi Zhu, Mingyu Liu, Dingning Liu, Jiange Yang, Zhoujie Fu, Junyi Chen, Chunhua Shen, Jiangmiao Pang, Kaipeng Zhang, and Tong He. Omniworld: A multi-domain and multi-modal dataset for 4d world modeling. *arXiv preprint arXiv:2509.12201*, 2025. 2, 14

Appendix

A. Additional Derivations

A.1. Bilinearity of the Plücker Reciprocal Product

We restate the elementary fact used in Section 3.3. Let $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{m}_i)$ and $\mathbf{r}_j = (\mathbf{d}_j, \mathbf{m}_j)$ be two 6-vectors. Define

$$\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} = \mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{d}_j \cdot \mathbf{m}_i. \quad (17)$$

For any $\alpha, \beta \in \mathbb{R}$ and any \mathbf{r}'_i ,

$$\begin{aligned} \langle \alpha \mathbf{r}_i + \beta \mathbf{r}'_i, \mathbf{r}_j \rangle_{\text{Pl}} &= (\alpha \mathbf{d}_i + \beta \mathbf{d}'_i) \cdot \mathbf{m}_j + (\alpha \mathbf{m}_i + \beta \mathbf{m}'_i) \cdot \mathbf{d}_j \\ &= \alpha (\mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{m}_i \cdot \mathbf{d}_j) \\ &\quad + \beta (\mathbf{d}'_i \cdot \mathbf{m}_j + \mathbf{m}'_i \cdot \mathbf{d}_j) \\ &= \alpha \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} + \beta \langle \mathbf{r}'_i, \mathbf{r}_j \rangle_{\text{Pl}}. \end{aligned}$$

The same holds in the second argument by symmetry. Hence $\langle \cdot, \cdot \rangle_{\text{Pl}}$ is bilinear.

A.2. Coplanarity Criterion

Two rays $\mathbf{r}_i, \mathbf{r}_j$ are coplanar (i.e. they intersect, are parallel, or coincide) if and only if $\langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}} = 0$. The proof is standard [9, 25] and rests on the observation that $\mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{d}_j \cdot \mathbf{m}_i$ is, up to a sign, the determinant of a 4×4 matrix whose columns are the homogeneous coordinates of two points on each line; the determinant vanishes iff the four points are coplanar.

A.3. SE(3) Invariance

For a rigid motion $g = (R, \mathbf{t}) \in \text{SE}(3)$, the action on a ray is

$$g \cdot (\mathbf{d}, \mathbf{m}) = (R\mathbf{d}, R\mathbf{m} + \mathbf{t} \times R\mathbf{d}). \quad (18)$$

Substituting into Eq. (5),

$$\begin{aligned} \langle g \cdot \mathbf{r}_i, g \cdot \mathbf{r}_j \rangle_{\text{Pl}} &= R\mathbf{d}_i \cdot (R\mathbf{m}_j + \mathbf{t} \times R\mathbf{d}_j) \\ &\quad + R\mathbf{d}_j \cdot (R\mathbf{m}_i + \mathbf{t} \times R\mathbf{d}_i) \\ &= \mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{d}_j \cdot \mathbf{m}_i \\ &\quad + R\mathbf{d}_i \cdot (\mathbf{t} \times R\mathbf{d}_j) \\ &\quad + R\mathbf{d}_j \cdot (\mathbf{t} \times R\mathbf{d}_i). \end{aligned}$$

The last two terms cancel because the scalar triple product $\alpha \langle \mathbf{b} \times \mathbf{c} \rangle$ is antisymmetric in any two of its arguments, giving $R\mathbf{d}_i \cdot (\mathbf{t} \times R\mathbf{d}_j) = -R\mathbf{d}_j \cdot (\mathbf{t} \times R\mathbf{d}_i)$. Hence $\langle g \cdot \mathbf{r}_i, g \cdot \mathbf{r}_j \rangle_{\text{Pl}} = \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}}$.

A.4. Reduction of Term (D) to the Plücker Reciprocal Product

Recall (Eq. (10)) that, with the Q/K flip,

$$(D) = \mathbf{r}_i^\top M \tilde{\mathbf{r}}_j, \quad \mathbf{r}_i = (\mathbf{d}_i, \mathbf{m}_i), \quad \tilde{\mathbf{r}}_j = (\mathbf{m}_j, \mathbf{d}_j), \quad (19)$$

where $M = E_q^\top E_k \in \mathbb{R}^{6 \times 6}$. Writing $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ with $A, B, C, D \in \mathbb{R}^{3 \times 3}$,

$$(D) = \mathbf{d}_i^\top A \mathbf{m}_j + \mathbf{d}_i^\top B \mathbf{d}_j + \mathbf{m}_i^\top C \mathbf{m}_j + \mathbf{m}_i^\top D \mathbf{d}_j. \quad (20)$$

Setting $A = D = I_3$, $B = C = \mathbf{0}_3$ collapses this to $\mathbf{d}_i \cdot \mathbf{m}_j + \mathbf{m}_i \cdot \mathbf{d}_j = \langle \mathbf{r}_i, \mathbf{r}_j \rangle_{\text{Pl}}$. The choice $E_q = E_k = I_6$ in Proposition 1 produces exactly $M = I_6$, so the full M -block decomposition above shows which sub-blocks of M correspond to the reciprocal-product component and which sub-blocks generalize beyond it: the off-diagonal blocks A, D control the reciprocal-product-like part, while the diagonal blocks B, C control direction–direction and moment–moment couplings that are not present in the classical reciprocal product.

A.5. Flipped vs. Unflipped Parameterization Are Linearly Equivalent

Let $\mathbf{r} = (\mathbf{d}, \mathbf{m}) \in \mathbb{R}^6$ and let $P = \begin{pmatrix} 0 & I_3 \\ I_3 & 0 \end{pmatrix}$ be the orthogonal involution that swaps the (\mathbf{d}, \mathbf{m}) blocks ($P^2 = I_6$, $P^\top = P$). With learnable $E_q, E_k \in \mathbb{R}^{d \times 6}$, the flipped parameterization in Eqs. (7)–(8) feeds \mathbf{r}_i to E_q and $P\mathbf{r}_j$ to E_k , while the unflipped variant feeds \mathbf{r}_j to E_k directly. The corresponding term (D) in Eq. (10) reads

$$(D)_{\text{flip}} = \mathbf{r}_i^\top E_q^\top E_k P \mathbf{r}_j, \quad (D)_{\text{no-flip}} = \mathbf{r}_i^\top E_q^\top E_k \mathbf{r}_j.$$

For unconstrained E_k the map $E_k \mapsto E_k P$ is a bijection on $\mathbb{R}^{d \times 6}$, so the two parameterizations realize the same set of bilinear forms in $(\mathbf{r}_i, \mathbf{r}_j)$. Under iid Gaussian initialization the columns of E_k and of $E_k P$ also share the same distribution, so ℓ_2 weight decay is invariant under the substitution. The flip is therefore a choice of basis: it singles out a distinguished symmetric configuration ($E_q = E_k = I_6$) under which the bilinear form coincides with the Plücker reciprocal product (Proposition 1), but it does not change the function class realizable by the model.

A.6. Implementation Notes

Numerical safety. We clamp $\|\mathbf{m}\|$ at $\epsilon = 10^{-6}$ before normalization (Eq. (11)) to avoid division by zero in near-static-camera tokens.

Old-checkpoint compatibility. A previous version of the encoding (without PE RMSNorm) shipped α_q, α_k as zero-dimensional scalars, which are not compatible with FSDP’s flat-parameter constraint that all sharded tensors have ≥ 1 dimension. Current checkpoints store these as shape-(1,) tensors and the loader silently promotes old scalars at load time.

Two-layer MLP form of E_q, E_k . When the optional MLP form $E_q : \mathbb{R}^7 \rightarrow \mathbb{R}^h \rightarrow \mathbb{R}^d$ is used, zero-initializing

Method	Quality	Camera controllability				Distribution		
	CLIP \uparrow	RotErr \downarrow	TransErr \downarrow	CamMC \downarrow	ATE \downarrow	FVD \downarrow	FVD $_c$ \downarrow	FID \downarrow
<i>Wan-2.2 5B Scale</i>								
ReCamMaster [3]	24.97	0.131	0.297	0.402	0.322	874.30	890.52	62.53
ReRoPE [17]	25.20	0.137	0.241	0.366	0.307	684.57	650.31	60.77
UCPE [42]	25.39	0.113	0.180	0.274	0.203	703.41	755.83	61.50
RayPE (ours)	26.05	0.085	0.174	0.251	0.188	543.17	588.62	57.83
<i>Wan-2.2 14B Scale</i>								
ReCamMaster [3]	25.31	0.109	0.221	0.335	0.286	675.23	697.10	59.21
ReRoPE [17]	25.85	0.114	0.185	0.290	0.243	493.30	525.61	49.18
UCPE [42]	25.72	0.082	0.163	0.238	0.180	529.42	558.70	54.75
RayPE (ours)	26.30	0.058	0.126	0.165	0.144	280.17	354.52	41.01

Table 5. Main comparison on the RE10K held-out split under the *rescaled-alignment* protocol used in prior camera-controllability work. Predicted and ground-truth ViPE trajectories are normalized to a common translation norm before pose metrics (RotErr, TransErr, CamMC, ATE) are computed. Everything else matches Table 1. Higher is better for CLIP; lower is better for the rest.

both layers would produce a permanently dead gradient (the GELU after a zero linear is identically zero, and so is its Jacobian). We therefore zero only the output layer and apply Kaiming initialization to the input layer, which keeps the geometric branch exactly zero at step 0 while leaving its gradient non-zero.

A.7. Reference Results under Rescaled-Alignment

We briefly recap why the main paper reports pose metrics on the *raw, unscaled* ViPE trajectories. A camera-conditioned video model is meant to follow a user-specified trajectory *including its absolute scale*; once the predicted and the ground-truth trajectories are rescaled to a common norm at evaluation time, the resulting numbers no longer measure whether the model honored that scale, only whether the relative shape of the trajectory was reproduced. Two practical consequences follow. First, a degenerate solution that produces only a tiny global motion can still score well under the conventional protocol, because once both trajectories are stretched to the same norm the remaining shape error is small even though the absolute motion is essentially absent; the raw protocol exposes this failure mode directly. Second, rescaled pose numbers are implicitly expressed in units of each clip’s own ground-truth norm and are therefore difficult to compare across datasets with different scale conventions (SfM-normalized, SLAM-internal, metric), whereas the raw protocol keeps the unit fixed to ViPE’s metric output. We therefore adopt the raw protocol throughout the main paper, and report rescaled-alignment numbers here only as a cross-reference to prior camera-controllability work. Concretely, Table 5 repeats the comparison from Section 5.2 after normalizing the predicted and the ground-truth ViPE trajectories to a common translation norm before computing pose metrics. Everything else — held-out split, trajectory estimator, sampler configuration, and the video-quality metrics CLIP, FVD, FVD $_c$, and

FID — is identical to Table 1; only the alignment step differs. Because rescaling absorbs absolute-scale errors, the four pose metrics are systematically smaller than their unscaled counterparts in the main table. RayPE remains best on every metric in this setting as well, confirming that its advantage under the unscaled protocol is not an artifact of the alignment choice.

B. Cross-Domain Training Strategy

This section describes the data sources we use, the steps we take to make their camera-translation scales comparable, and the optimization recipe we apply on top of the pre-trained backbone.

B.1. Data Sources

We train on a concatenation of four datasets that span quite different camera-motion distributions and pose-estimation pipelines:

- **RealEstate10K** (RE10K) [43]: about 10K real-estate walkthrough clips with COLMAP-style SfM poses [27]. Smooth, primarily forward camera motion through indoor scenes; the canonical benchmark for camera-controllable video synthesis.
- **DL3DV** [19]: a more diverse multi-scene dataset with both indoor and outdoor scenes and a wider range of camera motions; poses provided by the dataset.
- **PanShot** [42] (in-house): an internal dataset of synthesized panoramic shots, where camera trajectories are known by construction (we use them as ground-truth) and the field of view is known per clip.
- **OmniWorld** [44]: open-world video with relatively dynamic content, with camera extrinsics estimated by DROID-SLAM [29]. We use the released text captions as conditioning prompts. DROID-SLAM applies an internal scale normalization that is unrelated to either metric scale or the SfM normalization of RE10K/DL3DV.

We do not perform any further re-estimation of camera pose; RayPE treats the per-frame extrinsics and intrinsics as inputs and is agnostic to which estimator produced them. The four data sources span SfM poses, deep SLAM poses, and ground-truth-by-construction poses, which provides a stress test for whether the encoding generalizes across pose distributions.

B.2. Per-Dataset Trajectory-Scale Alignment

The four data sources differ substantially in absolute translation magnitude. To make their Plücker moments comparable in distribution, we apply a single per-dataset multiplicative factor $\eta \in \mathbb{R}_{>0}$ to the camera positions:

$$\tilde{\mathbf{o}}_t = \eta \cdot \mathbf{o}_t, \quad \tilde{\mathbf{m}}_i = \tilde{\mathbf{o}}_t \times \mathbf{d}_i = \eta \cdot \mathbf{m}_i. \quad (21)$$

The values of η for each dataset are chosen so that the median moment magnitude $\text{median}(\|\mathbf{m}\|)$ over a sample of clips falls in a comparable range across datasets; in our default configuration $\eta_{\text{RE10K}} = \eta_{\text{DL3DV}} = \eta_{\text{PanShot}} = 1.0$ and $\eta_{\text{OmniWorld}} = 20.0$, the latter chosen to bring DROID-SLAM’s internally normalized translations into approximately the same band as RE10K. Concrete moment-magnitude statistics are reported in Section 5.

B.3. Optional Per-Clip Near-Depth Normalization

A constant per-dataset η does not absorb *within*-dataset variation: a long-corridor RE10K clip and a tight-room RE10K clip can still differ by an order of magnitude in scene depth. When a per-clip near-depth estimate z_n is available, we additionally divide the clip’s translation by z_n :

$$\tilde{\mathbf{o}}_t = \frac{\eta}{z_n} \cdot \mathbf{o}_t. \quad (22)$$

The near-depth estimate z_n is obtained offline (from a depth estimator on a small set of frames) and serves as a per-clip proxy for the scene scale. When z_n is unavailable for a clip we fall back to η alone. For OmniWorld we use only η in this work.

These two normalizations (per-dataset η and per-clip $1/z_n$) together aim to make the moment-magnitude distribution roughly homogeneous across the training mixture. The Normalize-Gate-Inject mechanism (Section 4.2) provides robustness for the residual within-distribution variation that this preprocessing does not remove.

B.4. Optimization Recipe

Backbone and conditioning. We start from a pretrained 5B-parameter Wan2.2-TI2V-5B backbone [32] and use it in its image-to-video mode (the first frame of the training clip is conditioned via the standard first-frame VAE-fuse path of the backbone). All experiments are trained at 480×832 spatial resolution and $T = 81$ raw frames (corresponding to $T_z = 21$ latent frames).

Zero initialization. The new Q/K geometry branch is zero-initialized, so $\text{pe}^q = \text{pe}^k = 0$ at step 0 and the network matches the pretrained DiT. PE RMSNorm weights start at 1, scale-gate biases at 0 (so $g \approx 0.5$), and α starts at 0. When E_q, E_k are implemented as two-layer MLPs, we zero only the output layer and use Kaiming initialization for the input layer.

Trainable parameters and optimizer. We jointly train the new RayPE modules (E_q, E_k , PE RMSNorms, scale gates, α) together with the backbone self-attention, FFN, and modulation-norm layers; text cross-attention, the VAE, and the text encoder remain frozen. The newly added RayPE parameters account for less than 0.1% of the model. We use AdamW with a single learning rate of 2×10^{-5} across all trainable groups, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, weight decay 10^{-2} , and a cosine schedule that anneals to 10% of the starting value at T_{max} steps. We tried per-group asymmetric learning rates that protect the pretrained weights more aggressively but observed no consistent improvement, so we kept the simpler single-LR recipe.

Mixed precision and parallelism. We train in BF16 with FP32 master weights, gradient checkpointing on the DiT blocks, and DDP across GPUs. Gradient clipping by norm at 1.0 is applied at the DDP root level.

Validation-time generation. Every 5,000 steps we run inference on a small held-out validation set (a 100-clip RE10K-test split) and save (i) the per-step training and validation loss, (ii) the synthesized videos, and (iii) ground-truth videos for visual comparison. Validation videos are generated with the standard 50-step CFG sampler at $\text{cfg_scale} = 5.0$.