TKCAM: Text and Keyframe to Camera Trajectory Generation

Haozhe Yang[1] **Zhiyang Dou**[1,†] **Zekai Gu**[2] **Cheng Lin**[3]
Wenping Wang[4] **Yuan Liu**[2] **Taku Komura**[1,†]

[1]The University of Hong Kong [2]The Hong Kong University of Science and Technology
[3]Macau University of Science and Technology [4]Texas A&M University

{yhz2004, zhiyang0}@connect.hku.hk taku@cs.hku.hk
zekai.gu@connect.ust.hk yuanly@ust.hk
chenglin@must.edu.mo wenping@tamu.edu

Abstract

Generating high-quality and controllable camera motion is essential for AI-assisted cinematography, video synthesis, and 3D scene understanding. We introduce **TKCAM**, a *Text*- and *Keyframe*-conditioned *CAM*era-motion synthesis framework based on generative masked modeling. We represent camera dynamics using a 12-dimensional kinematic feature comprising position, velocity, and a continuous rotation representation and discretize them into hierarchical motion tokens via a Residual Vector Quantizer (RVQ). A two-stage masked transformer architecture then learns to reconstruct and refine these tokens, utilizing explicit self- and cross-attention modules for multimodal conditioning. A central feature of our framework is sparse visual keyframe conditioning: users can provide free-form text prompts together with RGB observations at selected timestamps, which provide temporally localized visual guidance for generating coherent in-between trajectories. Furthermore, to advance evaluation standards, we curate **RealEstate10K-Cap**, a large-scale text-camera dataset, and establish a cross-domain benchmark with a Universal CLaTr Evaluator. Extensive experiments demonstrate that TKCAM surpasses recent state-of-the-art baselines on Fréchet distance (FID), text-motion matching scores, and retrieval metrics (R@K), while additional analyses evaluate temporal smoothness and cross-domain generalization. Code and the RealEstate10K-Cap dataset are available at `https://github.com/linearalgebrayhz/TKCAM`.

1 Introduction

Camera motion determines how a scene is perceived, understood, and narrated. For modern vision–language systems, the ability to synthesize high-quality and controllable 6-DoF camera trajectories enables AI-assisted cinematography [15, 16], video synthesis [12, 17, 18, 28, 33, 37], and spatial reasoning in 3D environments [4, 22, 27, 32]. Yet translating sparse, symbolic language into metrically consistent 3D motion remains challenging: language under-specifies geometry and timing; naïve regression drifts semantically; and existing datasets rarely couple reliable camera annotations with temporally grounded captions or standardized tools for evaluating text and camera-motion alignment.

Recent approaches [7, 16, 19, 36] have modeled text-to-trajectory either autoregressively or via diffusion. While these models can produce smooth paths given prompts, they still face three practical limitations. *First*, direct regression from text often ignores fine-grained visual semantics, leading to

[†]Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

trajectories that miss spatial relations or temporal emphasis. *Second*, existing control mechanisms commonly rely on explicit camera states, scene geometry, or other structured inputs, whereas users may instead wish to guide motion using sparse visual observations at selected timestamps. *Third*, the modality gap between discrete text and continuous 3D motion, combined with the limited availability of semantically rich supervision, hinders efficient learning and generalization.

In this paper, we address these issues with **TKCAM**, a text- *and* keyframe-conditioned camera-motion generator built on *generative masked modeling*. TKCAM first discretizes continuous 3D camera trajectories into hierarchical motion tokens using a Residual Vector Quantizer (RVQ). To learn the multimodal correspondences, we train a base masked transformer that performs conditional demasking to reconstruct the masked motion tokens. Instead of complex token-level masking heuristics, we adopt a simple attention design within the transformer blocks: a motion-motion self-attention module to capture the temporal dynamics of the trajectory, followed by a motion-condition cross-attention module to directly inject the text and visual keyframe embeddings. A *residual masked transformer* then further refines the higher RVQ quantization levels in a coarse-to-fine manner, enhancing the high-frequency fidelity of the camera motion. As a result, at inference, TKCAM supports both text-only generation and *animator-style keyframe conditioning*: users can provide free-form text prompts alongside sparse RGB keyframes (e.g., first-frame or start/end observations), which serve as temporally localized visual guidance while the model generates coherent in-between trajectories conditioned on both the text and visual observations.

To train TKCAM, we curate **RealEstate10K-Cap**, a large-scale camera motion dataset built upon RealEstate10K [38]. RealEstate10K-Cap couples high-quality 3D camera trajectories with temporally grounded directorial captions. Specifically, the corpus comprises **25K** annotated sequences (**5 million** frames, approximately **50** hours in total) across diverse real estate environments. Furthermore, a significant bottleneck in current camera generation research is the absence of a standardized and cross-domain evaluation metric. To support evaluation across different camera-motion domains, we further construct a Universal CLaTr Evaluator by retraining the CLaTr contrastive objective [7] on a balanced mixture of indoor, cinematic, and character-centric trajectory data.

We conduct extensive experiments to validate that TKCAM generates semantically faithful camera trajectories with smooth translational motion under both text-only and text-keyframe settings. On our mixed-domain benchmark, TKCAM outperforms recent state-of-the-art baselines on Fréchet distance (FID), CLaTr-based text–motion matching score, and retrieval metrics (R@K). Furthermore, ablations support the benefits of our 12-dimensional kinematic representation, RVQ discretization, and the dual masked transformer architecture. Our main contributions can be summarized as follows:

- We present **TKCAM**, a hierarchical masked generative framework tailored to multimodal camera trajectory synthesis. Building on established RVQ-VAE and masked modeling techniques, TKCAM introduces a camera-specific 12-dimensional kinematic representation together with decoupled motion self-attention and multimodal cross-attention for camera-motion generation.
- We introduce animator-style sparse-keyframe conditioning for text-driven camera generation. RGB observations at selected timestamps provide soft visual guidance through cross-attention, enabling bidirectional generation of coherent in-between trajectories without requiring explicit camera extrinsics as user input.
- We curate **RealEstate10K-Cap**, a large-scale dataset of reconstructed camera trajectories paired with temporally grounded, scene-aware captions, and establish a cross-domain evaluation protocol with a *Universal CLaTr Evaluator*. We further validate the framework through controlled baseline retraining, temporal-smoothness analysis, and human evaluation of the proposed evaluator. Among the compared methods, ours achieves the best FID, matching, and retrieval scores.

2 Related Work

Camera Trajectory Generation. Early efforts in camera trajectory generation relied on optimization-based motion planning [5, 8] or direct continuous regression, which frequently suffered from high-frequency spatial jitter and struggled with semantic alignment. To address these issues, recent frameworks employ VLM-based planners to orchestrate 3D coordinates from abstract text (e.g., ShotVerse [31]), alongside generative paradigms spanning diffusion and discrete sequence modeling. CamSketch [26] further explores interactive camera-trajectory authoring from user sketches and example-derived motion codes. One prominent line of research focuses on diffusion models:

CCD [16] introduced a text-guided diffusion model in character-centric coordinates, whose keyframe conditioning specifies explicit camera states; E.T. [7] and Director3D [19] subsequently leveraged Diffusion Transformers (DiT) to synthesize global trajectories conditioned on character dynamics and on text for subsequent 3D scene generation, respectively. However, despite their impressive generative quality, diffusion-based approaches inherently require dozens of iterative denoising steps, leading to high computational latency. Alternatively, auto-regressive discrete models (such as GenDoP [36]) have also been explored to bypass diffusion latency; we discuss their limitations regarding unidirectional generation in our subsequent discussion on discrete sequence modeling.

Camera Trajectory Datasets. The evolution of camera trajectory datasets reflects a trade-off between geometric reliability and cinematographic expressiveness. Early datasets either relied on synthetic generation [16] or leveraged structured captures of static scenes, such as MVImgNet2.0 [11], RealEstate10K [38], and DL3DV-10K [20]. While these datasets offer high-quality, calibrated trajectories, they rarely come with descriptions of shot intent. To address this, recent benchmarks like E.T. [7] and DataDoP [36] extract trajectories directly from feature films accompanied by rich textual descriptions. However, cinematic footage frequently contains artistic effects—such as motion blur, rapid cuts, and shallow depth-of-field—which can introduce severe artifacts during Structure-from-Motion (SfM) reconstruction, deviating from stable, natural navigation behaviors. Our curated RealEstate10K-Cap corpus reaches a balance: it preserves the geometric precision of stable, real-world indoor captures while circumventing the reconstruction hazards of movie clips. DataDoP adopts a related two-stage strategy that derives motion descriptions from trajectories before language-model captioning; our contribution is the resulting indoor dataset with scene-grounded captions rather than this captioning strategy itself.

Discrete Generative Modeling for Sequences. To avoid the instability of continuous high-dimensional regression and the high latency of diffusion, another line of research formulates trajectory generation as a discrete sequence modeling task. Auto-regressive (AR) models, such as GenDoP [36], utilize tokenizers and transformers to predict camera movements token-by-token. While AR models excel at capturing discrete multimodal distributions, their strict sequential generation nature often leads to error accumulation over long sequences. More critically, the unidirectional (left-to-right) autoregressive paradigm limits their ability to condition on future observations, making animator-style tasks like filling the gap between a start and an end keyframe more difficult.

Building upon discrete paradigms, TKCAM is designed to avoid both the iterative sampling of diffusion and the unidirectional generation of AR models. Following MoMask [10], we combine a 12-dimensional kinematic input, an RVQ tokenizer, and a masked generative framework for camera trajectory generation. Rather than generating frame-by-frame, our dual masked transformer predicts trajectories holistically across time. This enables parallel decoding and bidirectional keyframe conditioning via masked generation, with an explicit cross-attention mechanism for multimodal conditioning.

3 Dataset

Data Source and Preprocessing. We construct our dataset upon the RealEstate10K collection [38], which provides high-quality SfM-reconstructed camera trajectories across diverse indoor scenes. To obtain clean and reliable motion data, we apply a series of rigorous preprocessing steps. We discard videos shorter than 4 seconds to ensure sufficient motion variation, and truncate continuous motions longer than 10 seconds to maintain temporal consistency. We further exclude near-static segments by filtering out trajectories where the cumulative camera displacement falls below a predefined threshold. After filtering, the curated set contains approximately 25K high-fidelity trajectory sequences, preserving a rich diversity of cinematographic indoor motion patterns.

Caption Generation. A major challenge in building text-motion datasets is the tendency of Vision-Language Models (VLMs) to hallucinate 3D camera dynamics. To keep motion descriptions consistent with the trajectories while adding scene detail, we leverage a **two-stage captioning pipeline** that couples deterministic trajectory analysis with generative visual grounding.

In the first stage, we derive a strict geometric motion prior directly from the relative camera poses. We integrate local camera velocities while suppressing minor pose-estimation jitter (threshold < 0.005). We then apply rule-based kinematic thresholds to classify translational (e.g., track, dolly) and rotational (e.g., pan, tilt) movements based on the OpenGL convention. Notably, simultaneous lateral

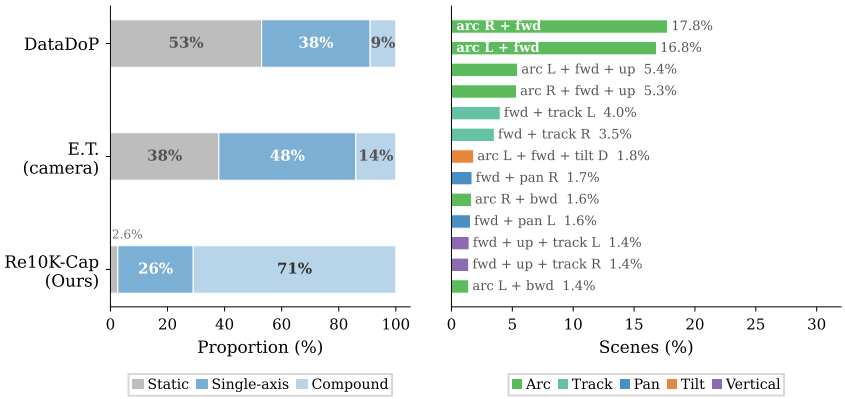


Figure 1: Motion distribution comparison across camera trajectory datasets. **Left:** Coarse-grained breakdown into static, single-axis, and compound motion (DataDoP and E.T. values estimated from their published figures). **Right:** Top-13 motion combinations in RealEstate10K-Cap; the remaining 35.1% is distributed across 50+ other combination patterns.

tracking and yaw are merged and labeled as cinematic *arcs*. This step yields a rule-derived motion description (e.g., *"camera dollies forward, then pans left"*).

In the second stage, we employ Qwen3-VL-4B-Instruct [3, 29] to ground this motion prior within the visual context. For each sequence, we uniformly sample up to 32 video frames. The VLM is strictly prompted to treat the deterministic geometric prior as an immutable constraint (*absolute motion*), while enriching it with salient scene elements, spatial layouts, and cinematic pace. Generation is performed using stochastic decoding (temperature 0.7, top-p 0.9, max new tokens 128). If visual frames are unavailable, the caption falls back to the rule-derived description. The full prompt template is provided in Appendix A.2.

Dataset Statistics. Our dataset comprises 25K high-fidelity trajectory sequences sampled from diverse indoor environments, yielding approximately **5 million** frames and **50** hours of continuous footage. Each trajectory spans an average of **201.7** frames (∼**6.72** seconds at 30 FPS), capturing smooth, uninterrupted camera navigation through complex real-world spaces. Table 1 provides a comprehensive quantitative comparison with existing camera trajectory datasets.

Table 1: Comparison with existing camera trajectory datasets. Our dataset focuses on indoor environments with scene-aware captions.

Dataset	#Samples	Domain	Caption	Vocab	Avg. Length (s)
RealEstate10K [38]	79K	Indoor	×	-	-
CCD [16]	25K	Synthetic	Synthetic	48	7.2
E.T. [7]	115K	Film	Camera-Char	1.7K	3.8
DataDoP [36]	29K	Film	Directorial	8.7K	14.4
Ours	**25K**	**Indoor**	**Scene-aware**	**3.8K**	**6.72**

Comparison with Existing Datasets. Prior camera trajectory datasets mainly rely on synthetic environments or film clips. E.T. [7] and DataDoP [36], for example, use feature films with stylized cinematography, but such footage often contains motion blur, rapid cuts, and shallow depth-of-field, causing SfM artifacts and deviating from stable, natural camera navigation. As shown in Figure 1, film-derived datasets also suffer from motion imbalance: near-static shots comprise 53% of DataDoP and 38% of E.T., whereas our filtering reduces static content to 2.6%. In contrast, 71% of RealEstate10K-Cap contains compound motion, compared to 9% in DataDoP and 14% in E.T. The top-13 motion patterns cover diverse behaviors such as forward arcs, dolly-track combinations, and multi-axis motion, with the remaining 35.1% spread across 50+ patterns. These statistics characterize the motion distribution of RealEstate10K-Cap but do not by themselves establish downstream dataset utility. We therefore additionally conduct a controlled comparison by training the same TKCAM architecture on equally sized subsets of RealEstate10K-Cap and DataDoP; the full results are provided in Appendix C.4. Unlike film-based datasets, RealEstate10K-Cap is built from stable real-estate tours, yielding reliable 3D reconstructions, while its two-stage captioning pipeline provides trajectory-grounded motion descriptions together with rich spatial context.

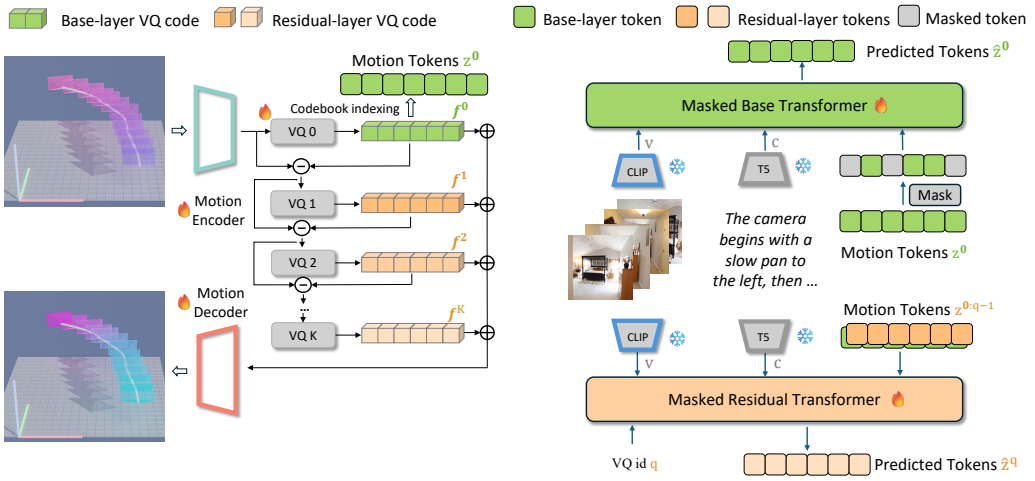


Figure 2: Overview of our proposed pipeline. Given per-frame camera poses $\mathcal{M}$ as trajectory, a text description $\mathcal{T}$ and sparse keyframe images $\mathcal{V}$, we first encode the continuous 12-dimensional camera trajectory into latent features and discretize them with a multi-level Residual Vector Quantizer (RVQ), producing a hierarchy of base-layer (green) and residual-layer (orange) motion tokens. During training, a masked base transformer predicts masked base-layer tokens conditioned on the text embedding from a frozen T5 encoder and visual tokens extracted by a frozen CLIP encoder, while a masked residual transformer refines higher-level residual tokens conditioned on the previously predicted tokens and the same multimodal context. Finally, a motion decoder reconstructs smooth and temporally coherent camera trajectories from the predicted hierarchical motion tokens.

4 Method

Given a text description $\mathcal{T}$ and a sparse set of keyframe images $\mathcal{V} = \{v_t\}_{t=1}^{T_k}$, our goal is to generate a temporally coherent and semantically consistent camera trajectory $\mathcal{M} = \{m_t\}_{t=1}^{T}$ that is conditioned on these keyframes and aligned with the described scene dynamics.

Our overall pipeline is illustrated in Fig. 2. We discretize the continuous camera trajectories into hierarchical motion tokens using a Residual Vector Quantizer (Sec. 4.1). A masked generative modeling framework is then adopted to learn multimodal correspondences between text, RGB keyframes, and motion tokens (Sec. 4.2), where base and residual masked transformers collaboratively predict and refine hierarchical motion tokens for coherent trajectory generation.

4.1 Motion Representation and Quantization

To ensure mathematical continuity and mitigate the instability of direct high-dimensional regression, we formulate the camera trajectory at frame t as a 12-dimensional kinematic vector $m_t = [p_t, v_t, r_t]$. This incorporates normalized absolute coordinates p_t, per-frame linear velocities v_t (acting as an implicit temporal regularization to smooth trajectories), and a continuous 6D rotation representation r_t to avoid singularities [39]. To better capture the modes of cinematographic dynamics, we map this continuous 12D sequence into discrete tokens [35] using Residual Vector Quantization (RVQ) [10, 34], transforming the geometric motion modeling into a tractable discrete sequence generation problem.

Hierarchical Motion Tokens. The continuous kinematic camera trajectories $m_t = [p_t, v_t, r_t]$ are first passed through a lightweight encoder composed of 1D convolutions and residual connections. This maps the motion sequence into latent embeddings $z \in \mathbb{R}^d$. These embeddings are then quantized by a multi-level RVQ module equipped with K codebooks. The quantized representation $\hat{z}$ is obtained by summing the selected codewords across valid quantization levels: $\hat{z} = \sum_{k=1}^{K'} f_{q^{(k)}}$ where $f_{q^{(k)}}$ denotes the selected codeword from the k-th codebook. To encourage robust representation learning, we employ *quantizer dropout* [34] during training, randomly truncating the number of active quantizers $K' \leq K$ with a probability $p = 0.2$. This hierarchical discrete design provides two main benefits: first, it transforms continuous regression into a stable token prediction task, improving semantic coherence; second, lower quantization levels capture coarse trajectory patterns (e.g., global direction), while higher levels progressively refine high-frequency motion details.

Training Objective. To optimize the RVQ autoencoder, we employ a multi-component objective tailored to 12D kinematic properties. The codebooks are updated via Exponential Moving Average (EMA) [10, 25], while the encoder is optimized by the commitment loss [25]: $\mathcal{L}_c = \frac{1}{K'} \sum_{k=1}^{K'} \|z^{(k)} - \mathrm{sg}[f_{q^{(k)}}]\|_2^2$, where $z^{(k)}$ is the residual feature at level k, $f_{q^{(k)}}$ is its selected codeword as defined above, and $\mathrm{sg}[\cdot]$ is the stop-gradient operator.

For trajectory reconstruction, let $\ell(x, \hat{x})$ denote the Smooth L1 (Huber) loss. The RVQ objective $\mathcal{L}_{\text{RVQ}}$ combines a global reconstruction term $\mathcal{L}_{\text{rec}} = \ell(m_t, \hat{m}_t)$ with an explicit kinematic loss $\mathcal{L}_{\text{exp}}$ and commitment weight β:

$$\mathcal{L}_{\text{RVQ}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{exp}} \mathcal{L}_{\text{exp}} + \beta \mathcal{L}_c \tag{1}$$

Specifically, $\mathcal{L}_{\text{exp}}$ independently supervises translation-velocity and 6D rotation components to ensure structural integrity: $\mathcal{L}_{\text{exp}} = \ell([p_t, v_t], [\hat{p}_t, \hat{v}_t]) + \ell(r_t, \hat{r}_t) + \lambda_{\text{orth}} \mathcal{L}_{\text{orth}}$. The term $\mathcal{L}_{\text{orth}}$ enforces the orthogonality of the predicted 6D rotation vectors $\hat{r}_1, \hat{r}_2 \in \mathbb{R}^3$:

$$\mathcal{L}_{\text{orth}} = \mathbb{E}\left[(\|\hat{r}_1\|_2 - 1)^2 + (\|\hat{r}_2\|_2 - 1)^2 + (\hat{r}_1^\top \hat{r}_2)^2\right] \tag{2}$$

Following MoMask [10], we set $\lambda_{\text{exp}} = 0.5$ and $\beta = 0.02$. λ_{orth} is fixed at 0.1.

4.2 Masked Generative Modeling

Our framework adopts a two-stage masked generative modeling paradigm [10]. Following RVQ discretization, the continuous trajectory is mapped to a discrete latent space $Z \in \{0, \ldots, V-1\}^{B \times S \times K}$, where V is the codebook size, S is the sequence length, and $K = 4$ is the number of quantization levels. We train two separate networks: a base masked transformer to model the primary structural motion $Z^{(0)}$, and a residual masked transformer to predict fine-grained details $Z^{(1:K-1)}$.

Base Masked Transformer. The base transformer p_θ reconstructs the first-level motion tokens $Z^{(0)}$ from a corrupted observation. During training, a subset of valid tokens is replaced with a `[MASK]` token following a randomized cosine schedule [6, 10]. To inject multimodal conditioning, we extract semantic embeddings c via a frozen T5 encoder [24], and visual tokens v from sparse RGB keyframes via a frozen CLIP encoder [23]. Crucially, to resolve spatial-temporal ambiguity (e.g., distinguishing start from end keyframes), we explicitly inject a learned temporal positional embedding into each visual token corresponding to its VQ-aligned timestamp. These visual features provide temporally localized conditioning and do not directly specify or clamp metric camera poses.

Rather than heuristically adding visual tokens to motion tokens, our architecture utilizes decoupled attention [1, 2]. Each transformer block sequentially executes motion-motion self-attention over the masked sequence $\tilde{Z}^{(0)}$ to capture internal kinematic dynamics, followed by motion-condition cross-attention querying the concatenated multimodal condition $[c; v]$. This ordering ensures motion tokens establish temporal coherence before attending to multimodal conditions. To enable Classifier-Free Guidance [14], we independently drop c and v with a probability of $p = 0.2$ during training. Finally, the model is optimized using a standard cross-entropy loss applied exclusively to the masked positions, seamlessly avoiding the need for complex token-level heuristics.

Residual Masked Transformer. To generate cinematographically plausible high-frequency details (e.g., subtle camera jitters), the residual masked transformer p_ϕ refines motion tokens level-by-level in a fully parallel manner. For an active residual layer $q \in \{1, \ldots, K-1\}$ (sampled per batch via an inverted cosine schedule to emphasize lower levels), it predicts the residual tokens conditioned on the cumulative historical embeddings of all previous levels $H^{(q)} = \sum_{j=0}^{q-1} E(Z^{(j)})$, alongside the identical multimodal conditions c and v. The network shares weights across all quantization levels, utilizing q as an additional level-embedding. It is optimized by minimizing a conditional masked cross-entropy loss over the temporal dimension for the active layer. This coarse-to-fine framework allows TKCAM to establish a stable macroscopic trajectory structure before progressively enriching it with microscopic physical realism. Complete objective formulation can be found in Appendix B.

During inference, given a target trajectory length, we extract text and visual embeddings c, v via T5 and CLIP encoders and initialize the corresponding motion-token sequence as fully masked, including positions temporally aligned with the visual keyframes. The base transformer reconstructs the motion tokens using parallel decoding and Classifier-Free Guidance (CFG) to establish the coarse structure $\hat{Z}^{(0)}$. Subsequently, the residual transformer refines the trajectory across all quantization levels in

a feed-forward manner to predict $\hat{Z}^{(1:K-1)}$. The final sequence is passed to the RVQ decoder to produce the continuous trajectory:

$$\hat{m}_t = \text{Decoder}_{\text{RVQ}}\big(\hat{Z}^{(0)}, \ldots, \hat{Z}^{(K-1)}\big). \tag{3}$$

This mechanism supports both text-only generation and animator-style keyframe conditioning.

5 Experiment

Datasets and Setup. TKCAM is trained exclusively on RealEstate10K-Cap and evaluated zero-shot on the E.T. and DataDoP test subsets, demonstrating cross-domain generalization without any in-domain supervision on cinematic or character-centric data. All trajectories are normalized to first-frame-relative coordinates to ensure spatial comparability. We compare with CCD [16], E.T. [7], Director3D [19], and GenDoP [36]. Following [36], CCD and E.T. receive zeroed character-motion tokens at inference. Since GenDoP requires RGB-D input, we estimate depth with Depth Anything V2 [30] and fine-tune GenDoP on RealEstate10K-Cap. For a controlled comparison, we additionally fine-tune/retrain CCD and E.T. on RealEstate10K-Cap under the same evaluation protocol, and report both released pretrained and matched-training results where available.

TKCAM requires a target trajectory length at inference. For benchmark evaluation, we use the per-sample ground-truth trajectory length for all methods, with every trajectory evaluated at the same target duration. Thus, the target generation horizon is assumed to be known during evaluation. Baseline training and trajectory-length handling are detailed in Appendix C.2 and C.3.

We use the CLaTr framework [7], training an evaluator on 15K validation samples balanced across RealEstate10K-Cap, E.T., and DataDoP, all disjoint from model training data. Results are reported on a separate 1.5K-sample test set. Metrics include Matching Score and retrieval ($R@1, R@3, R@10$) for text-motion alignment, FID [13], computed on Universal CLaTr features, for distributional realism, and ΔDiv for diversity alignment.

To validate the Universal CLaTr evaluator against human judgment, we further conduct a pilot blinded human study over 90 caption-trajectory pairs from four methods and the ground-truth reference. Universal CLaTr shows a positive correlation with mean human judgments (Spearman $\rho = 0.312$, 95% CI $[0.089, 0.492]$). Full study details are provided in Appendix C.6.

Implementation Details. All experiments run on a single NVIDIA A100 80GB GPU. Our RVQ-VAE uses 256-code codebooks and 4 quantization stages. Both base and residual transformers have 4 layers, 6 heads, and hidden size 384, with T5 and CLIP ViT-B/32 as multimodal encoders. Non-autoregressive decoding yields about 30 trajectories/s. Full hyperparameters are in Appendix C.1.

5.1 Main Results

Text-to-Camera Generation. In Track A (Table 2), TKCAM (Text Only) achieves the best results among the compared methods across all six metrics. It attains an R@10 of **8.40%** and a Matching Score of **0.094**, compared with 5.40% and 0.069 for the strongest text-only baseline, fine-tuned GenDoP. It also achieves the lowest FID of **0.476** and diversity gap of **0.064**, indicating closer agreement with the reference trajectory distribution in the evaluator's feature space. The examples in Figure 3 illustrate TKCAM's ability to capture compound camera motions with correct directional semantics and smooth phase transitions. In these examples, the baselines exhibit reduced motion magnitude, directional misalignment, near-static motion, or circular loops that do not follow the prompts.

Keyframe-Conditioned Generation and Video Synthesis. In Track B, both visual-conditioning configurations of TKCAM outperform the compared RGBD-conditioned GenDoP baselines across all six metrics. Using RGB input without explicit depth conditioning, TKCAM (First-Frame) achieves an FID of **0.461** and an R@10 of 12.27%, compared with 0.566 and 8.25% for fine-tuned GenDoP (RGBD). The Sparse Keyframes configuration achieves the highest Matching Score of **0.129** and R@10 of **15.19%** within this track, while retaining a nearly identical FID of 0.462. We further demonstrate this animator-style conditioning in a downstream video synthesis pipeline (Figure 4). Given first- and last-frame observations alongside a directorial prompt, TKCAM generates a temporally coherent trajectory that serves as a geometric motion prior for Diffusion-as-Shader [9].

Table 2: Quantitative evaluation of text-to-camera trajectory generation on the mixed benchmark. Metrics include Fréchet distance computed on Universal CLaTr features (FID), text-trajectory Matching Score (mean cosine similarity in $[-1, 1]$), and Retrieval metrics (R@1, R@3, R@10, in %) computed over the full evaluation set of each track. ΔDiv represents the absolute difference between the generated trajectory diversity and the reference set diversity (closer to 0 is better). "Fine-tuned" denotes models fine-tuned/retrained on RealEstate10K-Cap. **Bold** indicates the best performance within each track.

Method	FID ↓	Matching ↑	R@1 ↑	R@3 ↑	R@10 ↑	ΔDiv ↓
Track A: Text-Conditioned (1500 samples)						
CCD [16]	0.688	0.003	0.00	0.20	0.93	0.140
CCD (fine-tuned) [16]	0.577	0.037	0.27	0.80	2.20	0.109
E.T. [7]	1.167	-0.013	0.00	0.27	1.00	0.502
E.T. (fine-tuned) [7]	0.577	0.040	0.20	0.80	2.20	0.095
Director3D [19]	0.849	-0.002	0.00	0.07	0.53	0.186
GenDoP (Text only) [36]	0.760	0.029	0.20	0.40	2.13	0.138
GenDoP (Text only, fine-tuned) [36]	0.507	0.069	0.87	2.33	5.40	0.076
TKCAM (Text Only)	**0.476**	**0.094**	**1.00**	**2.87**	**8.40**	**0.064**
Track B: Vision- & Text-Conditioned (Visual Keyframe Conditioning, 994 samples)						
GenDoP (RGBD) [36]	0.835	0.043	0.60	1.21	3.92	0.206
GenDoP (RGBD, fine-tuned) [36]	0.566	0.098	1.01	3.62	8.25	0.104
TKCAM (First-Frame)	**0.461**	0.120	1.61	4.63	12.27	0.079
TKCAM (Sparse Keyframes)	0.462	**0.129**	**2.82**	**5.43**	**15.19**	**0.071**

The resulting video exhibits stable camera motion consistent with the described motion, suggesting that the generated trajectories can serve as useful geometric priors for downstream video synthesis.

Temporal Smoothness. We further evaluate translational acceleration/jerk and angular velocity/acceleration/jerk. TKCAM with sparse keyframes closely matches the ground-truth translational dynamics, while rotational acceleration and jerk remain less well matched than GenDoP-RGBD, indicating rotational smoothness as a remaining limitation. Full derivative statistics and distributional comparisons are provided in Appendix C.5.

5.2 Ablation Studies

We validate our design choices on the mixed-domain benchmark using unified training configurations. The 12D-representation and fusion ablations use the Sparse Keyframes setting (Track B), while the RVQ-level ablation in Table 3 uses the Text Only setting (Track A).

12D Representation. Replacing our 12D formulation with a 9D variant (no velocity) degrades all metrics: FID 0.503→0.593, Matching 0.129→0.106, R@10 12.67%→10.66%, and ΔDiv 0.070→0.074, consistent with explicit velocity acting as an implicit temporal regularizer.

Multimodal Fusion. Our decoupled cross-attention outperforms a *Prefix Concatenation* baseline on both realism (FID 0.503 vs. 0.580) and alignment (R@10 12.67% vs. 12.07%, Matching 0.129 vs. 0.119), supporting the benefit of cross-attention for fine-grained temporal grounding.

Hierarchical Refinement. Table 3 reveals a trade-off across quantization depths: the base level alone (L1) attains the highest R@1 but the largest diversity gap, while adding residual levels lowers FID and the diversity gap at some cost to exact-match retrieval. The full model (L1–L4) achieves the best R@10 while maintaining competitive FID and diversity.

Table 3: Ablation on hierarchical RVQ levels.

RVQ Stages	FID↓	Match.↑	R@1↑	R@3↑	R@10↑	ΔDiv↓
L1 (Base Only)	0.582	0.094	**1.40**	4.07	9.33	0.081
L1 + L2	**0.522**	**0.102**	0.93	4.07	9.20	0.068
L1 + L2 + L3	0.525	0.099	1.33	**4.27**	8.73	**0.063**
Full (L1–L4)	0.529	0.100	1.13	3.80	**9.60**	0.065

Sparse Keyframe Conditioning. On the same 994-sample Track B evaluation set, adding sparse keyframes beyond the first frame improves the Matching Score from 0.120 to 0.129 and R@1/R@3/R@10 from 1.61/4.63/12.27% to 2.82/5.43/15.19%, respectively. The diversity gap also decreases from 0.079 to 0.071, while FID remains nearly unchanged (0.461 vs. 0.462). These results support the benefit of additional temporally localized visual guidance for text–trajectory alignment.

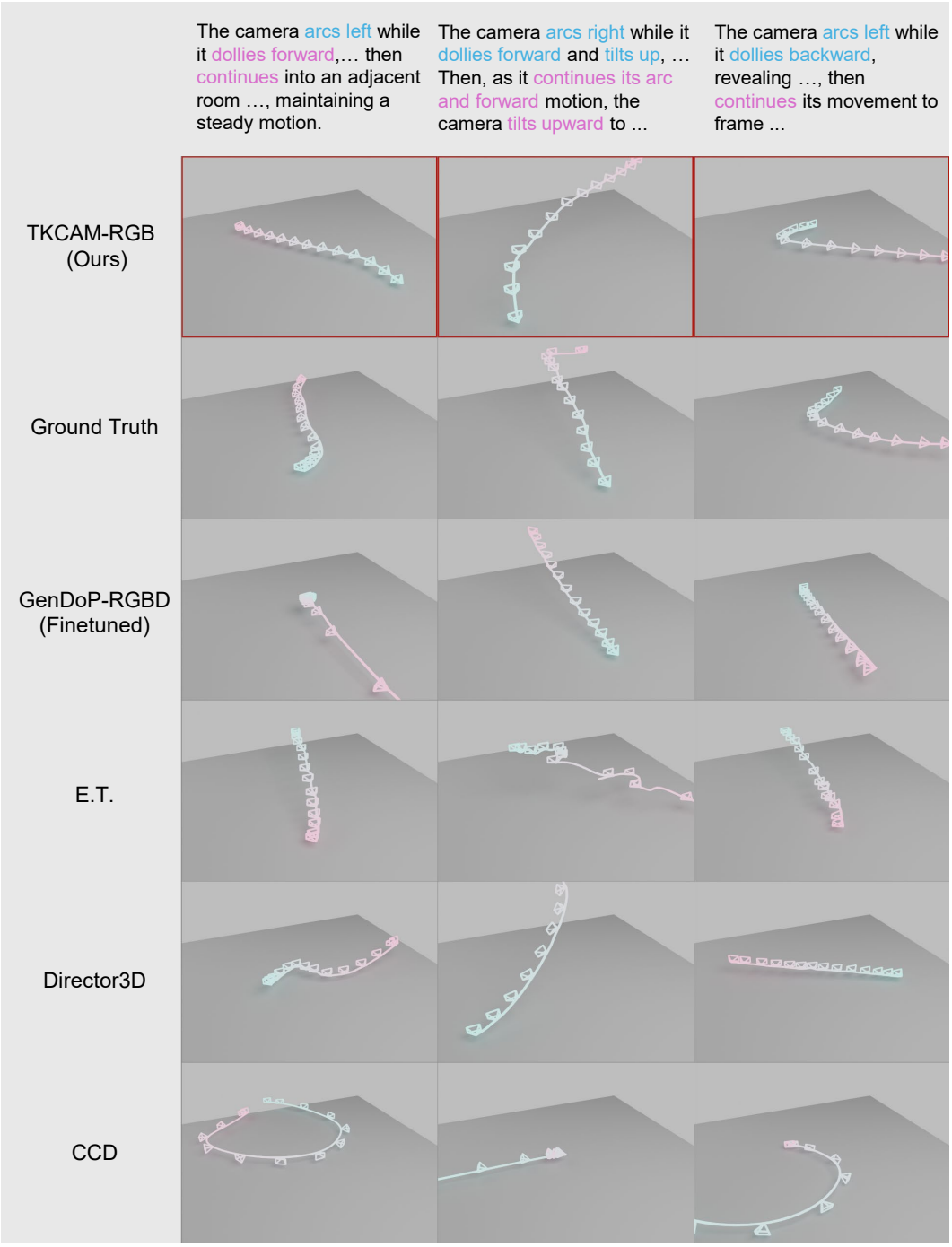


Figure 3: Qualitative comparison of generations. While baselines struggle with motion magnitude, directional alignment, or trajectory quality (e.g., CCD's circular loops), TKCAM produces semantically faithful paths that capture complex compound motions.

As discussed in Sec. 4.2, RGB keyframes provide soft visual guidance rather than explicit metric camera-pose constraints.

6 Conclusion

In this paper, we presented TKCAM, a text- and keyframe-conditioned framework for camera trajectory generation. TKCAM represents camera motion as 12-dimensional kinematic sequences and discretizes them with RVQ-VAE, reducing the instability of continuous regression. Its two-stage masked transformer enables multimodal fusion, high-frequency refinement, and animator-

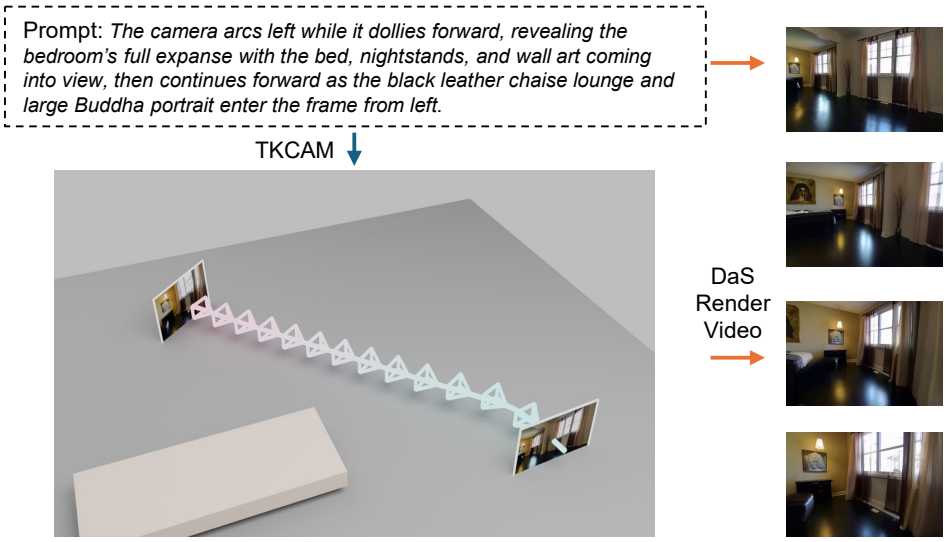


Figure 4: TKCAM as a geometric motion prior for downstream video generation tasks.

style keyframe conditioning. Experiments with our Universal CLaTr Evaluator demonstrate strong distributional realism and cross-modal alignment.

Limitations and Future Work. TKCAM has several limitations. First, the residual transformer is trained with ground-truth base tokens but uses predicted tokens at inference, which may introduce exposure bias. Second, training data is mainly indoor real-estate footage, leaving highly dynamic or stylized cinematography underexplored. Third, rotational smoothness remains weaker than translational smoothness, and inference requires a predefined target generation horizon. Finally, TKCAM currently focuses on single-camera extrinsic motion; incorporating explicit scene geometry, camera intrinsics, and multi-camera coordination is a promising future direction.

Acknowledgments

We thank Huaijin Pi, Boao Zhan, and Liangyu Zhang for valuable discussions. We are also grateful to the members of the HKU CGVU Lab for their helpful discussions and support, as well as for providing computational resources. Special thanks to Hanna Li for her personal support and encouragement.

Funding and competing interests. The authors declare no external funding or competing interests related to this work.

References

[1] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges. A Spatio-temporal Transformer for 3D Human Motion Prediction. *arXiv e-prints*, art. arXiv:2004.08692, Apr. 2020. doi: 10.48550/arXiv.2004.08692.

[2] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan. Flamingo: a Visual Language Model for Few-Shot Learning. *arXiv e-prints*, art. arXiv:2204.14198, Apr. 2022. doi: 10.48550/arXiv.2204.14198.

[3] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, W. Ge, Z. Guo, Q. Huang, J. Huang, F. Huang, B. Hui, S. Jiang, Z. Li, M. Li, M. Li, K. Li, Z. Lin, J. Lin, X. Liu, J. Liu, C. Liu, Y. Liu, D. Liu, S. Liu, D. Lu, R. Luo, C. Lv, R. Men, L. Meng, X. Ren, X. Ren, S. Song, Y. Sun, J. Tang, J. Tu, J. Wan, P. Wang, P. Wang, Q. Wang, Y. Wang, T. Xie, Y. Xu, H. Xu, J. Xu, Z. Yang, M. Yang, J. Yang, A. Yang, B. Yu, F. Zhang, H. Zhang, X. Zhang, B. Zheng, H. Zhong, J. Zhou, F. Zhou, J. Zhou, Y. Zhu, and K. Zhu. Qwen3-VL Technical Report. *arXiv e-prints*, art. arXiv:2511.21631, Nov. 2025. doi: 10.48550/arXiv.2511.21631.

[4] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[5] J. F. Blinn. Where am I? What am I looking at? [cinematography]. *IEEE Computer Graphics and Applications*, 8(4):76–81, 1988. doi: 10.1109/38.7751.

[6] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[7] R. Courant, N. Dufour, X. Wang, M. Christie, and V. Kalogeiton. E.T. the exceptional trajectories: Text-to-camera-trajectory generation with character awareness. In *European Conference on Computer Vision*, pages 464–480, 2024.

[8] Q. Galvane, M. Christie, C. Lino, and R. Ronfard. Camera-on-rails: automated computation of constrained camera paths. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, page 151–157, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450339919. doi: 10.1145/2822013.2822025. URL `https://doi.org/10.1145/2822013.2822025`.

[9] Z. Gu, R. Yan, J. Lu, P. Li, Z. Dou, C. Si, Z. Dong, Q. Liu, C. Lin, Z. Liu, W. Wang, and Y. Liu. Diffusion as shader: 3D-aware video diffusion for versatile video generation control. In *ACM SIGGRAPH 2025 Conference Papers*, 2025.

[10] C. Guo, Y. Mu, M. G. Javed, S. Wang, and L. Cheng. MoMask: Generative masked modeling of 3D human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1900–1910, 2024.

[11] X. Han, Y. Wu, L. Shi, H. Liu, H. Liao, L. Qiu, W. Yuan, X. Gu, Z. Dong, and S. Cui. MVImgNet2.0: A larger-scale dataset of multi-view images. *ACM Transactions on Graphics*, 43(6), 2024. doi: 10.1145/3687973.

[12] H. He, Y. Xu, Y. Guo, G. Wetzstein, B. Dai, H. Li, and C. Yang. CameraCtrl: Enabling camera control for video diffusion models. In *International Conference on Learning Representations*, 2025.

[13] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *arXiv e-prints*, art. arXiv:1706.08500, June 2017. doi: 10.48550/arXiv.1706.08500.

[14] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[15] H. Jiang, B. Wang, X. Wang, M. Christie, and B. Chen. Example-driven virtual cinematography by learning camera behaviors. *ACM Transactions on Graphics*, 39(4):45, 2020.

[16] H. Jiang, X. Wang, M. Christie, L. Liu, and B. Chen. Cinematographic camera diffusion model. *Computer Graphics Forum*, 43(2):e15055, 2024.

[17] Z. Kuang, S. Cai, H. He, Y. Xu, H. Li, L. J. Guibas, and G. Wetzstein. Collaborative video diffusion: Consistent multi-video generation with camera control. In *Advances in Neural Information Processing Systems*, volume 37, pages 16240–16271, 2024.

[18] L. Li, Z. Fan, W. Cong, X. Liu, Y. Yin, M. Foutter, P. Pan, C. You, Y. Wang, Z. Wang, et al. Martian world models: Controllable video synthesis with physically accurate 3D reconstructions. *arXiv preprint arXiv:2507.07978*, 2025.

[19] X. Li, Z. Lai, L. Xu, Y. Qu, L. Cao, S. Zhang, B. Dai, and R. Ji. Director3D: Real-world camera trajectory and 3d scene generation from text. *Advances in Neural Information Processing Systems*, 37:75125–75151, 2024.

[20] L. Ling, Y. Sheng, Z. Tu, W. Zhao, C. Xin, K. Wan, L. Yu, Q. Guo, Z. Yu, Y. Lu, X. Li, X. Sun, R. Ashok, A. Mukherjee, H. Kang, X. Kong, G. Hua, T. Zhang, B. Benes, and A. Bera. DL3DV-10K: A Large-Scale Scene Dataset for Deep Learning-based 3D Vision. *arXiv e-prints*, art. arXiv:2312.16256, Dec. 2023. doi: 10.48550/arXiv.2312.16256.

[21] I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. *arXiv e-prints*, art. arXiv:1711.05101, Nov. 2017. doi: 10.48550/arXiv.1711.05101.

[22] C. Patel, H. Nakamura, Y. Kyuragi, K. Kozuka, J. C. Niebles, and E. Adeli. UniEgoMotion: A unified model for egocentric motion reconstruction, forecasting, and generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10318–10329, 2025.

[23] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.

[24] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv e-prints*, art. arXiv:1910.10683, Oct. 2019. doi: 10.48550/arXiv.1910.10683.

[25] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural Discrete Representation Learning. *arXiv e-prints*, art. arXiv:1711.00937, Nov. 2017. doi: 10.48550/arXiv.1711.00937.

[26] W.-L. Wei, Y.-H. Hu, and J.-C. Lin. CamSketch: Authoring camera trajectories for 3D virtual concerts via example-derived codes and user sketching. In *SIGGRAPH Asia 2025 Posters*, 2025. doi: 10.1145/3757374.3771476.

[27] H. Xiong, X. Xu, J. Wu, Y. Hou, J. Bohg, and S. Song. Vision in action: Learning active perception from human demonstrations. *arXiv preprint arXiv:2506.15666*, 2025.

[28] D. Xu, W. Nie, C. Liu, S. Liu, J. Kautz, Z. Wang, and A. Vahdat. CamCo: Camera-controllable 3D-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024.

[29] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[30] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao. Depth Anything V2. *arXiv e-prints*, art. arXiv:2406.09414, June 2024. doi: 10.48550/arXiv.2406.09414.

[31] S. Yang, Z. Wang, X. Yang, S. Zhang, X. Kong, T. Wu, X. Zhao, R. Zhang, A. Zhao, and A. Rao. ShotVerse: Advancing cinematic camera control for text-driven multi-shot video creation. *arXiv preprint arXiv:2603.11421*, 2026.

[32] C. Yu, W. Zhai, Y. Yang, Y. Cao, and Z.-J. Zha. HERO: Human reaction generation from videos. *arXiv preprint arXiv:2503.08270*, 2025.

[33] W. Yu, J. Xing, L. Yuan, W. Hu, X. Li, Z. Huang, X. Gao, T.-T. Wong, Y. Shan, and Y. Tian. ViewCrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024.

[34] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi. SoundStream: An End-to-End Neural Audio Codec. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 30:495–507, Jan. 2022. doi: 10.1109/TASLP.2021.3129994.

[35] J. Zhang, Y. Zhang, X. Cun, S. Huang, Y. Zhang, H. Zhao, H. Lu, and X. Shen. T2M-GPT: Generating Human Motion from Textual Descriptions with Discrete Representations. *arXiv e-prints*, art. arXiv:2301.06052, Jan. 2023. doi: 10.48550/arXiv.2301.06052.

[36] M. Zhang, T. Wu, J. Tan, Z. Liu, G. Wetzstein, and D. Lin. GenDoP: Auto-regressive camera trajectory generation as a director of photography. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[37] G. Zheng, T. Li, R. Jiang, Y. Lu, T. Wu, and X. Li. CamI2V: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024.

[38] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics*, 37(4), 2018.

[39] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the Continuity of Rotation Representations in Neural Networks. *arXiv e-prints*, art. arXiv:1812.07035, Dec. 2018. doi: 10.48550/arXiv.1812.07035.

A More details on RealEstate10K-Cap

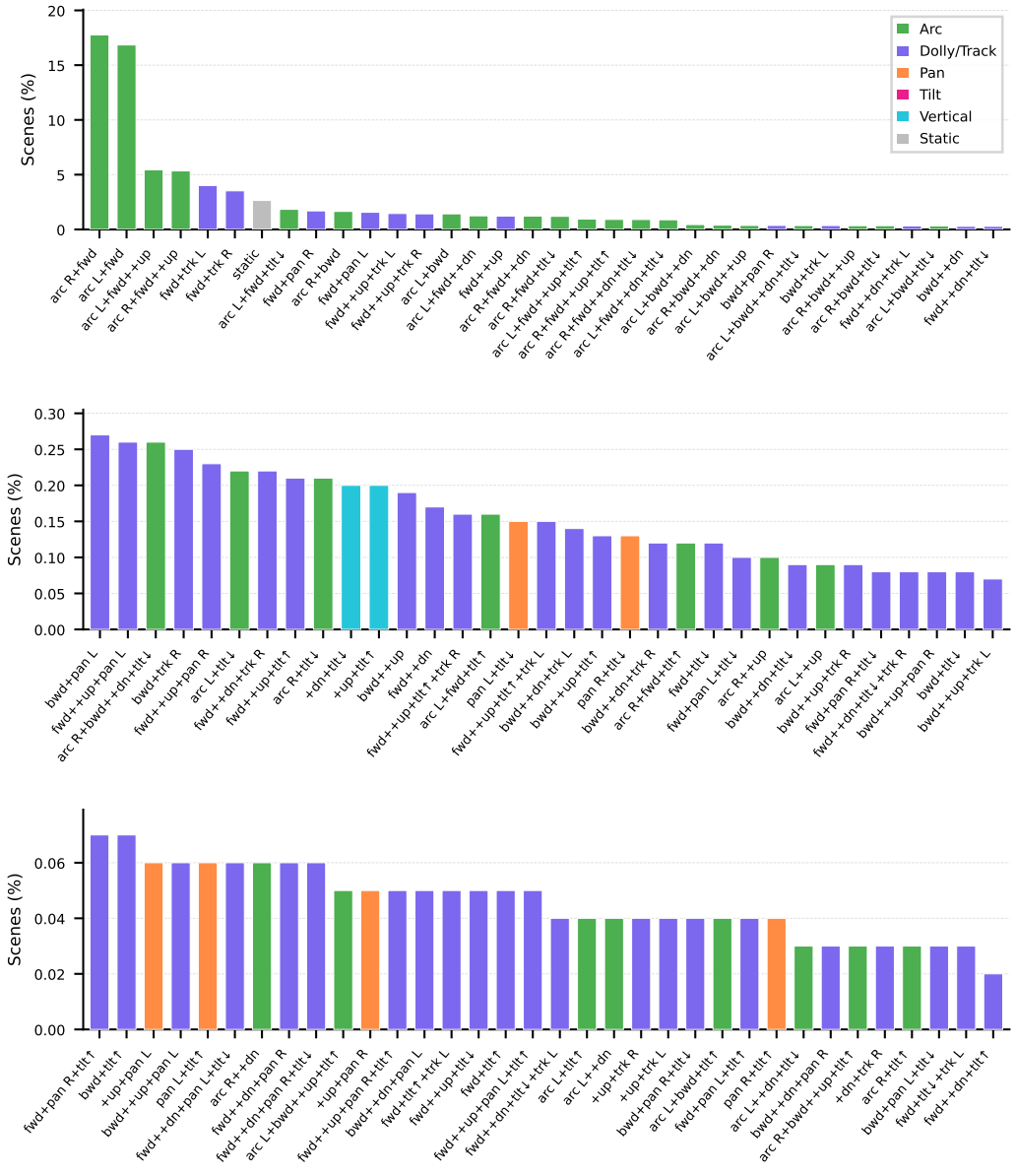


Figure 5: Complete motion category distribution of RealEstate10K-Cap, extending Figure 1 to all identified patterns. Categories are sorted by frequency and color-coded by motion family.

A.1 Full Motion Distribution

Figure 5 presents the complete motion category distribution of RealEstate10K-Cap, extending the Top-13 summary in the main text (Figure 1) to all identified motion patterns. Each category is defined by a combination of translational axes (dolly, track, boom) and rotational axes (pan, tilt), with arcs denoting simultaneous lateral tracking and yaw and compound motions representing simultaneous multi-axis movement. The long-tail distribution—where the top two categories (arc+forward) account for ∼35% of sequences while the remaining 65% spans 50+ diverse patterns—reflects the natural variety of real-estate footage and provides broad motion coverage.

A.2 Data Captioning Pipeline

As introduced in the main text, constructing the RealEstate10K-Cap dataset relies on a two-stage captioning pipeline that keeps motion descriptions consistent with the trajectories while adding scene context.

Stage 1: Deterministic Kinematic Extraction. We extract strict geometric motion priors directly from the relative camera poses provided by RealEstate10K, which were estimated with SLAM followed by bundle adjustment [38]. We integrate local camera velocities and apply rule-based kinematic thresholds to classify translational movements (e.g., track left/right, dolly in/out, boom up/down) and rotational movements (e.g., pan, tilt, roll). Simultaneous lateral tracking and yaw are merged into compound cinematic descriptions, such as *"arc left"*. This yields a rule-derived motion description of the trajectory.

Stage 2: VLM-based Visual Grounding. We uniformly sample up to 32 video frames per sequence and feed them into Qwen3-VL-4B-Instruct alongside the deterministic geometric prior. Generation is performed using stochastic decoding (temperature 0.7, top-p 0.9, max new tokens 128). The prompt template is designed to strictly enforce the geometric prior while enriching it with visual context:

> ***System Prompt:*** You act as a professional cinematographer describing a camera shot. Given the camera motion sequence reference and [n_frames] frames from `[total_frames]` uniformly sampled frames from the video, your task is to merge the motion sequence with the key scene elements into 1 or 2 fluid sentences.
>
> **Rules:**
>
> 1. **ABSOLUTE MOTION:** You MUST use the provided motion sequence exactly as written. DO NOT alter the directions or sequence.
> 2. **FORMAT:** Use 'then' or 'finally' to separate stages. Keep it cinematic but factual.
> 3. **CONTENT:** For each stage include: movement type (pan/tilt/dolly/truck/arc/pedestal), direction, and pace (slow/medium/fast).
>
> **Examples:**
>
> - The camera dollies forward with slight shaking, gradually closing the distance between the viewer and the fireplace.
> - The camera pans left slowly and smoothly, revealing a stone fireplace. Then it moves along the left wall, maintaining shot angle.
> - The camera dollies backward smoothly, revealing a basketball court in the foreground, then continues backward while panning slightly left.
>
> **Reference:** `[Insert Stage 1 Prior]`
> **Motion description:**

B Detailed Training Objectives

In this section, we provide the detailed mathematical formulations for the training objectives of our proposed TKCAM framework, complementing the high-level descriptions in the main text.

B.1 RVQ Autoencoder Objective

To optimize the Residual Vector Quantizer (RVQ), we employ a multi-component objective tailored to the 12D kinematic properties. Let $\ell(x, \hat{x})$ denote the Smooth L1 (Huber) loss. The overall RVQ objective $\mathcal{L}_{\text{RVQ}}$ combines a global reconstruction term $\mathcal{L}_{\text{rec}} = \ell(m_t, \hat{m}_t)$ with an explicit kinematic loss $\mathcal{L}_{\text{exp}}$ and a commitment loss $\mathcal{L}_c$:

$$\mathcal{L}_{\text{RVQ}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{exp}}\mathcal{L}_{\text{exp}} + \beta\mathcal{L}_c$$

Specifically, $\mathcal{L}_{\text{exp}}$ independently supervises the translation-velocity and 6D rotation components to ensure structural integrity:

$$\mathcal{L}_{\text{exp}} = \ell([p_t, v_t], [\hat{p}_t, \hat{v}_t]) + \ell(r_t, \hat{r}_t) + \lambda_{\text{orth}}\mathcal{L}_{\text{orth}}$$

The term $\mathcal{L}_{\text{orth}}$ enforces the orthogonality of the predicted 6D rotation vectors $\hat{r}_1, \hat{r}_2 \in \mathbb{R}^3$:

$$\mathcal{L}_{\text{orth}} = \mathbb{E}\left[(\|\hat{r}_1\|_2 - 1)^2 + (\|\hat{r}_2\|_2 - 1)^2 + (\hat{r}_1^\top \hat{r}_2)^2\right]$$

B.2 Masked Transformer Objective

For the generative modeling stage, the masked transformers are optimized to predict the masked motion tokens. The base model is trained using a cross-entropy loss applied exclusively to the masked non-padding positions Ω:

$$\mathcal{L}_{\text{base}} = -\frac{1}{|\Omega|} \sum_{(b,i)\in\Omega} \log p_\theta\big(Z_{b,i}^{(0)} \mid \tilde{Z}_b^{(0)}, c_b, v_b\big)$$

Similarly, the residual masked transformer minimizes the conditional masked prediction loss over the temporal dimension for the sampled active layer q_b:

$$\mathcal{L}_{\text{res}} = -\frac{1}{\sum_b S_b} \sum_b \sum_{i<S_b} \log p_\phi\big(Z_{b,i}^{(q_b)} \mid H_b^{(q_b)}, q_b, c_b, v_b\big)$$

C Experiment Details

C.1 Full List of Hyperparameters

All experiments are conducted on a single NVIDIA A100 (80 GB) GPU. Table 4 lists the hyperparameters used for training both the RVQ autoencoder and the dual masked transformers.

Table 4: Detailed hyperparameters for TKCAM training.

Module	Hyperparameter	Value
RVQ Autoencoder	Codebook Size (V)	256
	Quantization Levels (K)	4
	Quantizer Dropout Prob (p)	0.2
	Kinematic Loss Weight (λ_{exp})	0.5
	Orthogonality Weight (λ_{orth})	0.1
	Commitment Weight (β)	0.02
	Base Learning Rate	1e-4
	Learning Rate Warm-up Iterations	500
	Batch Size	256
	Window Size	128
Transformers (Base & Residual)	Number of Layers	4
	Number of Attention Heads	6
	Hidden Dimension	384
	CFG Dropout Prob (p_{drop})	0.2
	Optimizer	AdamW [21]
	Base Learning Rate	5e-5
	Batch Size	64

C.2 Baseline Training and Evaluation Protocol

To reduce the effect of unequal training-domain exposure, we additionally fine-tune or retrain feasible baselines on RealEstate10K-Cap. In particular, CCD [16] and E.T. [7] are retrained on RealEstate10K-Cap, while GenDoP [36] is evaluated in both its original and fine-tuned configurations. Director3D [19] is evaluated using its released pretrained configuration, for which matched retraining is unavailable in our setup. All methods are evaluated using the same mixed-domain benchmark, trajectory-length protocol, and Universal CLaTr evaluator. Table 2 reports both pretrained and matched-training results where available.

Table 5: Controlled comparison between RealEstate10K-Cap and DataDoP as training sources. The same TKCAM architecture is trained on equally sized subsets of the two datasets. Lower FID and ΔDiv are better.

Test Domain	FID ↓		ΔDiv ↓	
	Re10K-Cap Train	DataDoP Train	Re10K-Cap Train	DataDoP Train
Mixed	**0.967**	1.190	**0.196**	0.389
RealEstate10K	**1.204**	1.458	**0.274**	0.394
E.T. (OOD)	**0.956**	1.134	**0.013**	0.179
DataDoP	**1.431**	1.572	**0.134**	0.466

C.3 Trajectory-Length Handling

TKCAM requires a target sequence length to be specified at inference. For benchmark evaluation, we use the per-sample ground-truth trajectory length, denoted as `traj_len`, and enforce the same target duration across all evaluated methods.

Because the VQ tokenizer temporally downsamples trajectories by a factor of four, TKCAM first decodes $\lceil \texttt{traj_len}/4 \rceil \times 4$ frames and then resamples the trajectory to `traj_len`. GenDoP generates its native 120 camera poses and is resampled to the target length; CCD generates 300 native frames and is similarly resampled; Director3D's native trajectory is resampled using spherical linear interpolation (slerp) for rotations; and E.T. is run directly at the per-sample target length. We additionally audited the generated outputs used in these comparisons and found no target-length mismatches for TKCAM, GenDoP, CCD, Director3D, and E.T.

This protocol ensures that the reported trajectory metrics are not confounded by differences in evaluated duration. The requirement of a predefined generation horizon remains a practical assumption of the current formulation. We do not use segment stitching in the reported experiments, and generation substantially beyond the training-length distribution has not been systematically evaluated.

C.4 Controlled Dataset Comparison

To isolate the effect of training data, we train the same TKCAM architecture separately on equally sized subsets ($\sim$8.5K trajectories each) of RealEstate10K-Cap and DataDoP. We keep the tokenizer, model size, optimization schedule, checkpoint-selection criterion, and evaluator fixed across the two runs. We then evaluate both models on the same four test domains.

As shown in Table 5, the RealEstate10K-Cap-trained model achieves lower FID and a smaller diversity gap across all four evaluation domains, including the DataDoP test subset itself and E.T., which is out-of-domain for both training sources. These controlled results complement the motion-distribution statistics in the main paper and provide direct evidence for the utility of RealEstate10K-Cap as a training source.

C.5 Temporal Smoothness Analysis

We quantitatively evaluate temporal smoothness using finite-difference translational acceleration and jerk, together with angular velocity, acceleration, and jerk. Since visual conditioning changes the evaluated subset, we compare the frame-conditioned methods against the corresponding frame-conditioned ground-truth trajectories.

Raw derivative magnitudes alone are insufficient for assessing realistic motion, as minimizing acceleration or jerk may favor over-smoothed trajectories. We therefore additionally compute the 1D Wasserstein distance between each derivative distribution and the corresponding ground-truth distribution.

Table 6 shows that TKCAM closely matches the ground-truth translational dynamics. In particular, the sparse-keyframe setting achieves the smallest Wasserstein distances for translational acceleration and jerk. Because sparse RGB keyframes enter through soft cross-attention conditioning rather than hard-spliced motion tokens, they do not explicitly introduce trajectory discontinuities at the

Table 6: Temporal smoothness analysis in the vision-conditioned setting. Top: mean per-frame derivative magnitudes. Bottom: 1D Wasserstein distance between each derivative distribution and ground truth (lower is closer to real camera dynamics).

Method	**Trans. Acc.**	**Trans. Jerk**	**Ang. Vel.**	**Ang. Acc.**	**Ang. Jerk**
Mean per-frame derivative magnitude					
GT (reference)	0.00078	0.00054	0.00460	0.00134	0.00186
TKCAM (Sparse Keyframes)	0.00045	0.00024	0.00745	0.00718	0.01187
TKCAM (First-Frame)	0.00032	0.00016	0.00629	0.00641	0.01067
GenDoP (RGBD)	0.00092	0.00133	0.00394	0.00042	0.00058
1D Wasserstein distance to GT $\downarrow$					
TKCAM (Sparse Keyframes)	**0.00032**	**0.00029**	0.00350	0.00640	0.01085
TKCAM (First-Frame)	0.00043	0.00036	0.00215	0.00525	0.00906
GenDoP (RGBD)	0.00042	0.00069	**0.00118**	**0.00097**	**0.00138**

conditioned timestamps. In contrast, rotational acceleration and jerk remain less well matched than GenDoP-RGBD, indicating that rotational smoothness is a remaining limitation of the current model.

C.6 Human Validation of Universal CLaTr

We conduct a pilot blinded human study to examine whether Universal CLaTr is consistent with human judgments of text-trajectory alignment. We sample 18 benchmark captions, with six each from RealEstate10K-Cap, E.T., and DataDoP. For each caption, trajectories from five sources—TKCAM (Text Only), GenDoP (finetuned), E.T. (finetuned), CCD (finetuned), and the ground-truth reference—are presented in anonymized and randomized order. Three raters independently score how well each camera trajectory matches the corresponding caption on a five-point Likert scale, yielding 270 raw ratings over 90 caption–trajectory pairs.

Rater Instructions. For each caption, the five trajectories were shown as anonymized clips ("Clip A"–"Clip E") in randomized order. Raters were asked:

> "How well does THIS camera motion match the caption?"

The following instruction was displayed on every page:

> "**What to rate:** for each anonymized clip, judge only how well the camera's motion path matches the caption. Ignore image quality — only caption ↔ camera-motion agreement."

Each clip was rated on a five-point scale, with the displayed anchors "5 = perfectly", "3 = partially", and "1 = unrelated".

Inter-rater agreement is moderate, with Krippendorff's $\alpha = 0.49$, while the reliability of the three-rater averaged score is higher (ICC$(2, k) = 0.76$). We therefore use the mean rating across raters for the following analysis.

Universal CLaTr exhibits a positive correlation with mean human judgments across the 90 caption-trajectory pairs (Spearman $\rho = 0.312$, caption-bootstrap 95% CI $[0.089, 0.492]$; Kendall $\tau_b = 0.219$). Restricting the analysis to generated trajectories yields a stronger correlation (Spearman $\rho = 0.395$, 95% CI $[0.186, 0.544]$).

At the method level, human raters assign the highest mean score to TKCAM (3.30), followed by GenDoP (2.78), E.T. (2.61), and CCD (2.09). On the same evaluated subset, Universal CLaTr preserves five of the six pairwise orderings among the four generated methods. The only disagreement is between TKCAM and GenDoP, whose CLaTr scores are statistically indistinguishable on this subset. These results provide preliminary evidence that Universal CLaTr reflects human assessments of text-trajectory alignment, while the limited number of captions and raters motivates larger-scale perceptual validation in future work.

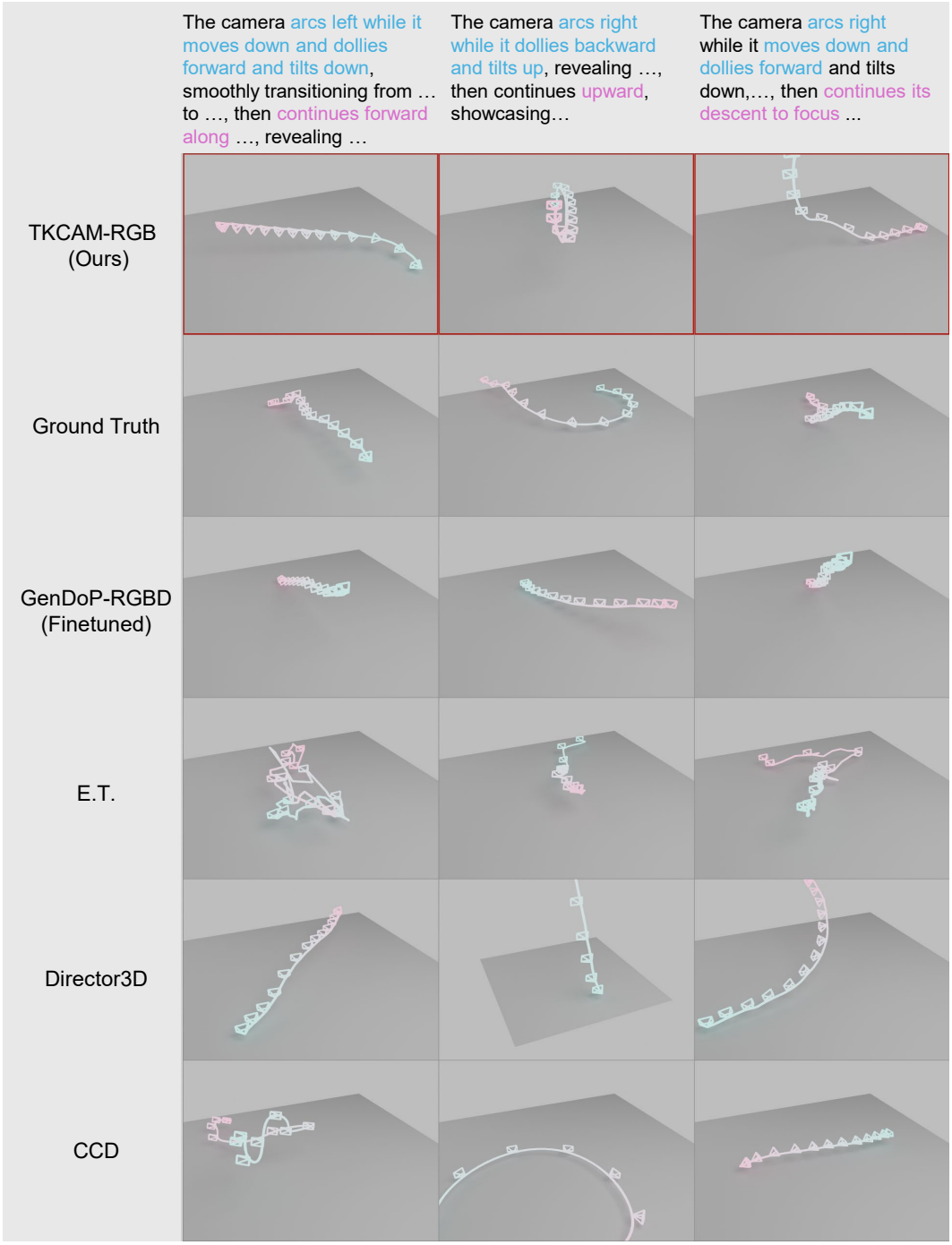


Figure 6: Additional qualitative comparisons with baselines.

D More TKCAM Generation Results

D.1 More Qualitative Comparison

Figure 6 presents additional qualitative comparisons across a broader range of motion types and scene contexts, supplementing the three examples shown in Figure 3 of the main text. We include examples featuring predominantly translational motion (dolly-forward, lateral tracking) and complex compound motions involving simultaneous arcing, dollying, and tilting. In most of these examples,

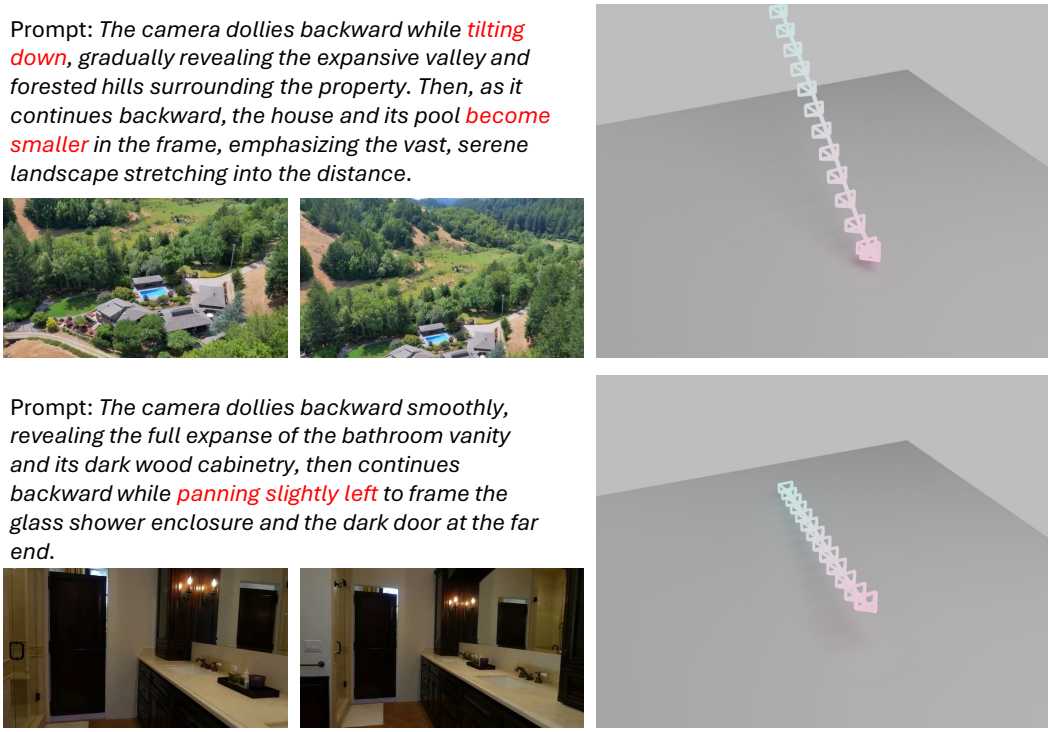


Figure 7: Two representative failure cases of TKCAM: tilt underestimation and action omission in multi-stage sequences.

TKCAM produces trajectories with correct directional semantics, while baselines exhibit the failure modes identified in the main text.

D.2 Failure Cases of TKCAM and Analysis

The vast majority of TKCAM's generated trajectories exhibit correct directional semantics and appropriate motion magnitude, as evidenced by our quantitative results (Table 2) and the qualitative comparisons in Figures 3 and 6. The following failure modes occur only in a small minority of cases, typically involving rare motion combinations or out-of-distribution prompts. Figure 7 illustrates two representative failure modes of TKCAM, with the misaligned motion keywords highlighted in red.

Tilt Underestimation. In the first example, the prompt specifies simultaneous backward dollying and tilting down, yet the generated trajectory captures only the translational component while neglecting the vertical rotation. We attribute this to a distributional bias in RealEstate10K-Cap: indoor real estate tours are dominated by horizontal navigation, leaving tilt motions underrepresented in training. As a result, the model tends to under-weight rotational components when they co-occur with a dominant translational cue.

Action Omission in Multi-Stage Sequences. In the second example, a two-stage motion (dolly backward, then pan slightly left) produces a trajectory that executes the primary dolly but omits the secondary pan. This occurs occasionally when a later-stage motion has low magnitude relative to the dominant motion, causing it to be under-sampled during masked token prediction. The issue is more pronounced when the secondary action is brief and semantically subtle.

Both failure modes are rare edge cases that arise specifically at the boundary of the training distribution—primarily for out-of-distribution vertical motions and low-magnitude secondary actions—rather than reflecting systematic architectural limitations. The strong quantitative performance reported in Table 2 indicates that such misalignments do not represent the typical generation behavior of TKCAM.

E Ethical and Societal Impact Discussion

Positive Impacts. TKCAM can make camera-path authoring more accessible by enabling non-expert users to generate controllable camera trajectories from natural language descriptions. This has direct positive applications in independent filmmaking, pre-visualization for film production, accessibility tools for virtual environments, and educational simulators for cinematography training. By open-sourcing our models and dataset, we further enable the research community to build upon this work for downstream video generation, 3D scene understanding, and embodied AI applications.

Potential Negative Impacts. The ability to generate plausible, text-controlled camera trajectories could potentially be misused in combination with video synthesis pipelines to produce deceptive deepfake content or synthetic surveillance footage that appears cinematographically authentic. Additionally, automated cinematography generation may reduce demand for certain professional roles in low-budget production contexts.

Mitigation. We will accompany the public release of TKCAM with a usage policy that prohibits malicious or deceptive applications. As a trajectory-only model, TKCAM does not generate visual content independently and requires a separate video synthesis component to produce any visual output, which introduces an additional mitigation layer. We encourage downstream users of this work to apply appropriate safeguards when integrating trajectory generation into end-to-end video generation pipelines.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction reflect the paper's contributions and scope, and they are supported in Sections 4 and 5 and the appendices.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 6 and the appendices, including potential exposure bias, the indoor-dominated training domain, weaker rotational smoothness, the need for a predefined generation horizon, and representative failure cases.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the model architecture in Section 4, the training and evaluation setup in Section 5, and the hyperparameters and evaluation protocols in the appendices. Our data are derived from the publicly available RealEstate10K dataset.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Code, models, and the RealEstate10K-Cap dataset will be publicly released. The paper provides architectural details and hyperparameters for reproduction.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Training details, the optimizer, data splits, hyperparameters, and metrics are specified in Section 5 and the appendices.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Benchmark metrics are reported as point estimates under a fixed random seed. For the human study (Appendix C.6), we report caption-bootstrap 95% confidence intervals for the correlation estimates.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the compute hardware and memory (a single NVIDIA A100 80GB GPU), training epochs, batch sizes, approximate wall-clock training times for each model component, and inference throughput.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The discussion is made in Appendix. Camera trajectory generation could potentially be misused for creating deceptive deepfake videos or synthetic surveillance footage. We acknowledge this risk and will implement usage guidelines upon public release.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This research does not involve any privacy or high-risk data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external datasets, models, and codebases are cited and used in accordance with their released licenses and terms. Major assets include RealEstate10K (CC BY 4.0), DataDoP (CC BY 4.0), Qwen3-VL-4B-Instruct (Apache-2.0), the MoMask and CLaTr codebases (MIT), E.T. code (MIT), GenDoP (Apache-2.0), and Director3D (CC BY-NC-SA 4.0). We also use T5 (Apache-2.0), OpenAI CLIP, and Depth Anything V2 under their respective released terms. Third-party code without an explicit redistribution license is used only for academic evaluation and is not redistributed with our release.

    Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce the RealEstate10K-Cap dataset. Its construction pipeline and prompt template are documented in Section 3 and Appendix A.2. It is released under CC-BY 4.0 license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We conduct a small pilot human evaluation with three volunteer raters recruited through personal contacts. Participants rated text–trajectory alignment on a five-point Likert scale using anonymized and randomized trajectory visualizations. No monetary compensation was provided. The evaluation instructions and protocol are provided in Appendix C.6.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The pilot study involved only a minimal-risk perceptual rating task on anonymized camera-trajectory visualizations and did not collect sensitive or personally identifiable information

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: Section 3 describes the use of an open-source vision-language model (Qwen3-VL-4B-Instruct) for dataset captioning, and Section 4 describes the frozen T5 text encoder used in our model.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.