TEMPORALLY GROUNDED COMPOSITIONAL CAMERA MOTION UNDERSTANDING VIA GEOMETRIC KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding camera motion is fundamental to video perception, with applications in spatial intelligence and controllable video generation. Multimodal large language models (MLLMs) provide a natural interface for this task, but existing work typically assigns one or more labels to an entire clip. Such clip-level recognition overlooks two defining properties of real camera motion: it can change within a shot, and multiple movements can occur simultaneously. We therefore formulate camera-motion understanding as *temporally grounded, compositional recognition*, which requires a model to localize motion-consistent intervals and identify every movement active within each interval. We introduce CAMCHOREO, a benchmark of 4,229 real single-shot clips with expert-annotated temporal segments. Its annotations use a compact vocabulary of 20 direction-aware labels, and nearly half of the segments contain compound camera motion, with multiple movement primitives active simultaneously. Recognizing such fine-grained, compositional motion is hard for current MLLMs, whose visual encoders emphasize semantic content rather than the geometric evidence on which camera motion depends. Directly injecting features from a frozen 3D foundation model addresses this gap, but requires running the expensive geometry model on every input; we refer to this baseline as CAMINJECT. We instead propose CAMDISTILL, which distills the same geometric knowledge into lightweight camera tokens during training and removes the 3D model at inference. CAMDISTILL matches the accuracy of direct feature injection without running the 3D teacher at inference. Together, CAMCHOREO and CAMDISTILL advance camera-motion understanding from clip-level labeling to temporally grounded, compositional recognition.

1 INTRODUCTION

A video contains motion in the scene and motion of the camera observing it. Camera motion has its own expressive vocabulary and a long tradition in film grammar (Spottiswoode, 1959; Yilmaz et al., 2023): a pan redirects the viewer's gaze, a dolly reveals depth through parallax, and a zoom reframes the shot while the camera remains stationary. Recognizing these movements requires separating changes caused by the observer from those occurring in the scene. This capability is important for spatial intelligence (Zhang et al., 2026), controllable video generation (Bai et al., 2025a; Xing et al., 2025), and cinematic analysis (Wang et al., 2026b). Nevertheless, most camera-motion benchmarks still formulate the problem as single- or multi-label classification over an entire clip (Lin et al., 2026; Liu et al., 2026; Feng et al., 2026). That formulation does not reflect real footage: even within an uninterrupted shot, the camera may transition between movements or execute several movements at once. Clip-level labels consequently lose both temporal structure and physical composition.

We instead formulate the problem as *temporally grounded, compositional recognition*. Given a video, the model partitions each shot into motion-consistent intervals and predicts the complete set of direction-aware movements active in each interval. Figure 1 illustrates the task: the top panel annotates one shot as a sequence of intervals, each carrying several simultaneous movements, and the bottom panel shows our taxonomy grouped into five families. The task thus asks *what* the camera does, *when* each movement occurs, and *which* movements co-occur, none of which clip-level classification can isolate. To support it, we introduce CAMCHOREO, named for how a shot

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

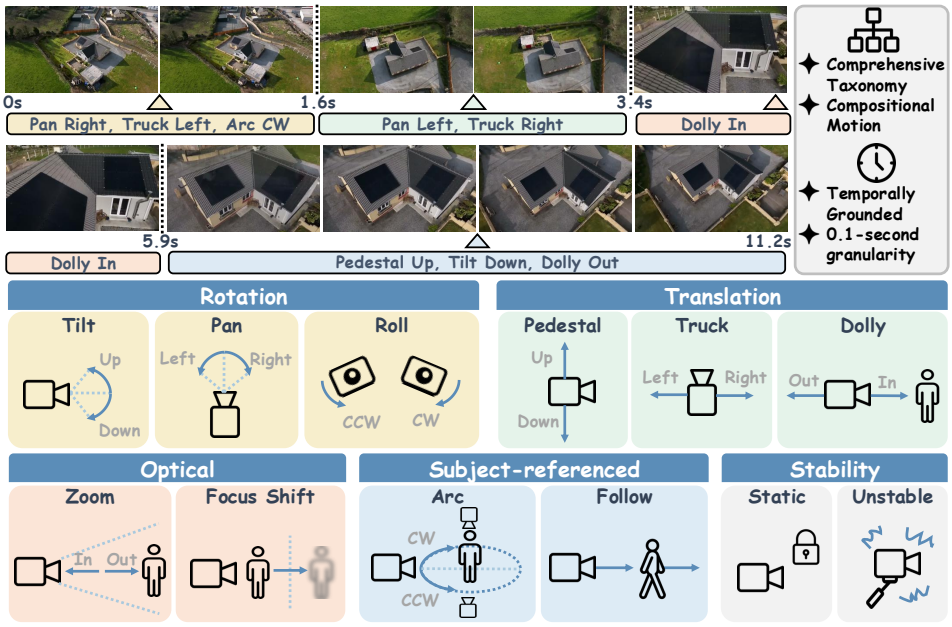


Figure 1. Temporally grounded compositional camera motion. A shot is represented by motion-consistent intervals, each carrying all simultaneous movements. CAMCHOREO covers 12 types (20 direction-aware labels) grouped into five families: rotation, translation, optical, subject-referenced, and stability.

choreographs camera-motion primitives over time. It contains 4,229 real single-shot YouTube clips, 8,591 expert-annotated segments, and 14,258 motion instances, covering 12 movement types and 20 direction-aware labels grounded in classical camera terminology (Nielsen et al., 2007), across nine content domains with boundaries at 0.1-second resolution. Within this benchmark, temporal variation and compound motion are common: 2,411 clips contain multiple segments, and 3,797 segments contain compound camera motion, with multiple movement primitives occurring simultaneously.

Current MLLMs perform poorly in this setting, revealing a representational gap: their vision encoders are optimized for semantic alignment (Radford et al., 2021; Tschannen et al., 2025), while camera motion depends on cross-frame geometry, including parallax, perspective change, and horizon rotation. Injecting features from a frozen 3D foundation model helps, but this baseline, CAMINJECT, must run the expensive geometry model on every test video. We propose CAMDISTILL to retain the geometric benefit without this inference-time dependency. A lightweight Geometry-aware Camera Token Extractor (GCTE) predicts one camera token per frame from intermediate frozen vision features (Dosovitskiy et al., 2020). A distillation objective aligns these tokens with the camera representation of a frozen 3D teacher. Through this objective, the model learns a geometry-informed camera representation during training. At inference, the 3D teacher is removed, leaving a compact geometry-aware stream with almost no runtime overhead.

Experiments confirm both the difficulty of the task and the value of geometric supervision. On CAMCHOREO, the strongest closed-source MLLM reaches 43.3 frame-level micro F1. SFT raises a 4B model to 62.2, and CAMDISTILL further improves it to 67.5, matching direct injection without running the 3D model at inference. The distilled representation also transfers to external benchmarks with different task formats, suggesting that it captures reusable camera-motion cues.

Our contributions are threefold:

- **Task and benchmark.** We formulate temporally grounded, compositional camera-motion recognition and introduce CAMCHOREO, to our knowledge the first real-video benchmark combining variable-length segments with direction-aware multi-label annotations.
- **Empirical diagnosis.** We show that within-shot transitions and simultaneous movements are common, and that generic MLLMs and geometry-only pose rules remain inadequate.
- **Efficient geometry distillation.** We propose CAMDISTILL, whose GCTE predicts per-frame camera tokens from frozen visual features and aligns them with a 3D teacher during training. It matches direct feature injection while removing the teacher and its cost at inference.

2 Related Work

2.1 Camera Motion and Cinematography Benchmarks

Film theory has long studied camera movement as a device with narrative and emotional functions (Spottiswoode, 1959; Nielsen et al., 2007; Yilmaz et al., 2023). Recent cinematography benchmarks evaluate camera movement alongside shot scale, lighting, and composition, mainly through clip-level classification or multiple-choice QA (Li et al., 2024b; Tang et al., 2025; Wang et al., 2026b; Liu et al., 2026; Wu et al., 2025a). CameraBench (Lin et al., 2026) introduces a broad expert vocabulary for real videos, while CameraMotionVQA (CMVQA) (Feng et al., 2026) supports controlled multi-label recognition on one-second synthetic clips. These resources advance camera-motion recognition, but still assign a single motion set to each clip. They neither localize variable-length intervals within real shots nor evaluate how movements co-occur over time. CamChoreo is designed specifically for this temporally grounded, compositional setting, as summarized in Table 1.

2.2 MLLMs for Camera Motion Understanding

General-purpose video MLLMs (Li et al., 2024a; Zhang et al., 2024; Bai et al., 2025b; Wang et al., 2025c) provide strong semantic understanding but are not trained to perceive camera geometry. Our task also relates to video temporal grounding, in which MLLMs localize events along a timeline (Wu et al., 2025b). Recent camera-motion methods introduce structured reasoning traces, explicit pose grounding, or textual pose prompts derived from geometry models (Wu et al., 2026; Yang et al., 2026; Feng et al., 2026). Other work augments MLLMs with 3D priors (Zheng et al., 2026). These studies demonstrate the value of geometry, but either retain the geometry model at inference or compress its output into discrete text. CamDistill instead transfers the teacher's *continuous* camera representation during training and removes the geometry model at inference.

2.3 Camera Pose Estimation in Video

Classical SfM and SLAM recover camera trajectories through feature matching and geometric optimization (Schonberger & Frahm, 2016; Davison et al., 2007; Engel et al., 2014; Teed & Deng, 2021; Li et al., 2026; Zhang et al., 2022). More recent feed-forward models such as DUSt3R, VGGT, and VGGT-Ω jointly predict camera pose and 3D scene structure in a single pass (Wang et al., 2024; 2025b; 2026a), with further estimators improving robustness and multi-view consistency (Huang et al., 2025; Wang et al., 2026c). VGGT and VGGT-Ω in particular attach a dedicated camera token to each frame, from which that frame's camera pose can be decoded. We distill the camera token itself into the MLLM, transferring a pose-associated, geometry-informed representation while leaving the mapping to camera-motion labels to the language model.

3 The CamChoreo Benchmark

3.1 Task Definition

Given a video V of duration T, the model predicts a set of temporal segments

$$\hat{\mathcal{S}} = \{(\hat{s}_i, \hat{e}_i, \hat{Y}_i)\}_{i=1}^{N}, \qquad 0 \le \hat{s}_i < \hat{e}_i \le T, \tag{1}$$

where $\hat{s}_i$ and $\hat{e}_i$ are the predicted start and end times, and $\hat{Y}_i \subseteq \mathcal{C}$ is the set of active camera-motion labels. The closed label space $\mathcal{C}$ contains 20 direction-aware labels derived from 12 movement types (Figure 1; Appendix D). A correct prediction must therefore recover both the segment boundaries and the complete set of co-occurring labels within each segment.

3.2 Data Curation

Collection and filtering. We collect YouTube footage from nine content domains and split each video into single shots at hard cuts with TransNetV2 (Soucek & Lokoc, 2024). The resulting shots are filtered by duration and visual quality and de-duplicated, yielding a diverse clip pool. To surface rare motions, a preliminary model assigns pseudo labels that are used only to guide candidate sampling and never enter the released annotations. The full funnel is detailed in Appendix C.

Table 1. Comparison with existing cinematography and camera-motion benchmarks. CAMCHOREO is the only real-video benchmark combining camera-specific, multi-label annotation with variable-length temporal grounding. "#Cls." counts direction-aware labels for CAMCHOREO.

Benchmark	Source	#Clips	Real	#Cls.	Multi-lbl.	Cam-spec.	Temporal
Cinematic2K (Li et al., 2024b)	Web	2,000	✓	11	×	×	×
VidComposition (Tang et al., 2025)	Movies	982	✓	7	✓	×	×
CineTechBench (Wang et al., 2026b)	Movies	120	✓	15	✓	×	×
ShotBench (Liu et al., 2026)	Movies	464	✓	16	×	×	×
CameraBench (Lin et al., 2026)	Web	∼3,000	✓	23	✓	✓	×
CameraMotionVQA (Feng et al., 2026)	Synthetic	12,274	×	15	✓	✓	×
CAMCHOREO (ours)	Web	4,229	✓	20	✓	✓	✓

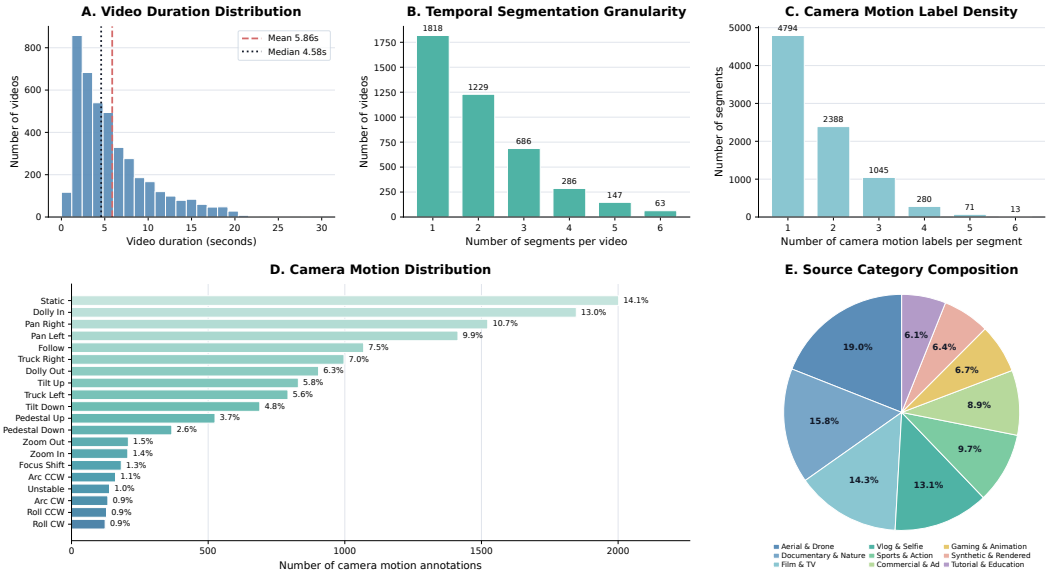


Figure 2. CAMCHOREO statistics. **(A)** video duration, **(B)** number of segments per clip, **(C)** number of simultaneous movements per segment, **(D)** the long-tailed distribution of the 20 direction-aware labels, and **(E)** the nine content domains.

Expert annotation. A team of five annotators with film- and media-related backgrounds label every clip from scratch, marking motion-consistent intervals and all active direction-aware movements. They use parallax to separate rotation from translation, perspective change to distinguish Dolly from Zoom, and scene context to separate camera from subject motion. Follow and Arc are always paired with their underlying primitive so that a semantic label never replaces the physical motion, and boundaries are placed at 0.1-second resolution wherever the active motion set changes. Annotations are cross-checked for quality. Clips with unresolved disagreements are discarded, leaving 4,229 videos in the final benchmark. Details are given in Appendix D.

3.3 DATASET STATISTICS

CAMCHOREO contains 4,229 single-shot clips totaling 6.88 hours, with 8,591 expert-annotated segments and 14,258 movement instances. Clips average 5.9 seconds (Figure 2A), keeping the benchmark focused on within-shot camera behavior rather than editing or long-form narrative. Figure 2 summarizes the properties most relevant to the task.

Temporal structure. Camera motion changes within most clips. As shown in Figure 2B, 2,411 of the 4,229 clips contain multiple segments, with some containing as many as six. Clip-level annotation would therefore merge distinct motion phases in more than half of the benchmark.

Compound camera motion. In the curated benchmark, 3,797 of the 8,591 segments (44.2%) contain compound camera motion with at least two simultaneous movement primitives, and some contain three or more (Figure 2C). A single-label prediction therefore drops part of the active camera state in nearly half of the benchmark segments.

Long-tailed labels and domains. The 20 direction-aware labels follow a pronounced long-tailed distribution (Figure 2D). Static, Dolly, and Pan are frequent, whereas Zoom, Roll, Arc, and Focus Shift are rare. The clips span nine content domains, with aerial, documentary, and film footage contributing the largest shares (Figure 2E).

Comparison with other benchmarks. Table 1 highlights two limitations of existing benchmarks. General cinematography datasets cover camera movement only as one attribute among many, while camera-specific benchmarks provide richer motion vocabularies but still assign a single label set to an entire clip. Consequently, the representative benchmarks in Table 1 do not capture how the active camera motion changes within a real shot. CAMCHOREO addresses this gap by combining real web video with camera-specific, direction-aware multi-label annotations over variable-length temporal segments. It therefore evaluates both compound camera motion within each segment and its evolution over time.

4 CAMDISTILL: DISTILLING GEOMETRY INTO CAMERA TOKENS

4.1 MOTIVATION

Recognizing camera motion requires comparing perspective, parallax, scale, and orientation across frames. MLLM vision encoders, however, are optimized for semantic alignment and encode these geometric signals only weakly. Feed-forward 3D foundation models such as VGGT (Wang et al., 2025b) and VGGT-Ω (Wang et al., 2026a) are designed to recover them. Given a set of frames, these models estimate depth, point maps, and camera pose. In particular, they associate each frame with a dedicated *camera token* from which its pose is decoded. The evolution of these tokens across time therefore provides a compact, camera-oriented geometric representation. We use this representation as the distillation target for the MLLM.

A direct way to exploit this signal is CAMINJECT (Figure 3a), which runs the 3D model alongside the frozen vision encoder, projects each teacher camera token into the LLM hidden space, and concatenates the projected tokens with the visual sequence. This baseline is effective (Section 5) but expensive: the 3D model must process every video at inference, and models such as VGGT-Ω apply global attention over the tokens of all frames, so latency and memory grow rapidly with video length. Since these camera tokens are clearly useful, we ask whether they can be obtained without running the 3D model at test time. CAMDISTILL moves the teacher entirely to training (Figure 3b). A lightweight student predicts per-frame camera tokens from the frozen vision features the MLLM already computes, and a distillation loss aligns them with the teacher's tokens. The 3D model is then discarded, so CAMDISTILL keeps the geometric supervision with almost no inference overhead.

4.2 ARCHITECTURE

The student takes the form of a *Geometry-aware Camera Token Extractor* (GCTE), a lightweight branch attached to the frozen vision encoder (Figure 3b). It reads intermediate visual features without modifying the pretrained visual stream, produces one camera token per frame through alternating attention blocks, and places these tokens before the corresponding visual tokens. The LLM can then condition its predictions on both semantic visual features and an explicit camera representation.

Camera tokens. Let $x_i^{(\ell)} \in \mathbb{R}^{P_i \times d_v}$ denote the frozen vision features of frame i at encoder layer ℓ, where P_i is the number of visual tokens and d_v is the vision hidden size. GCTE reads *intermediate* rather than final-layer features. Intermediate layers retain local geometric cues such as parallax and perspective change, while higher layers become increasingly specialized for semantic alignment (Feng et al., 2026). This choice gives the student access to a cleaner camera-related signal.

For each of the T frames, GCTE maintains a camera token $c_i \in \mathbb{R}^{d_c}$ whose initial value $c_i^{(0)}$ is a learnable embedding. Following the reference-view convention of 3D reconstruction, the first frame uses a dedicated embedding and all other frames share a second one, marking the first frame as the reference view. The attention blocks below then refine these tokens.

Alternating attention blocks. GCTE stacks M alternating blocks, each a *frame-wise cross-attention* followed by a *global camera self-attention* (Figure 3b), mapping the initial tokens

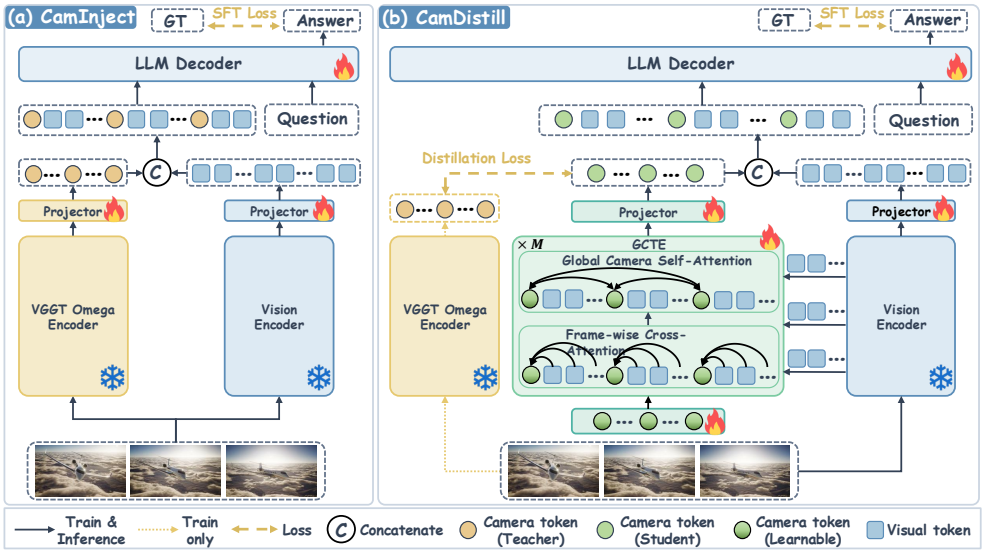


Figure 3. CAMINJECT versus CAMDISTILL. (a) CAMINJECT runs a frozen 3D foundation model (VGGT-Ω) beside the vision encoder and inserts each frame's *teacher* camera token just before that frame's visual tokens in the LLM input, so the 3D model is required at inference. **(b)** CAMDISTILL instead trains a lightweight GCTE, a stack of M alternating frame-wise cross-attention and global camera self-attention blocks, to predict the *student* camera tokens, which are inserted at the same positions. A distillation loss aligns them with the teacher, which is removed at inference. Snowflake and flame icons denote frozen and trainable components.

$\{c_i^{(0)}\}_{i=1}^T$ to the final camera states $\{c_i^{(M)}\}_{i=1}^T$. We write $c_i^{(m)}$ for the state of frame i after block m. The two attention operations serve complementary roles. Frame-wise cross-attention extracts camera-relevant evidence from each frame, while global self-attention compares that evidence across time. Together, they represent both the geometry of individual views and its temporal evolution.

Frame-wise cross-attention. Frame-wise cross-attention lets each camera token read only the visual tokens from its corresponding frame:

$$\tilde{c}_i^{(m)} = \mathrm{CrossAttn}\big(c_i^{(m-1)},\ x_i^{(\ell_m)}\big). \tag{2}$$

Here the camera token $c_i^{(m-1)}$ from the previous block is the query, the frame features $x_i^{(\ell_m)}$ at the layer ℓ_m tapped by block m are the keys and values, and $\tilde{c}_i^{(m)}$ is the resulting camera token. This interaction is one-way: the block updates only the camera token. The visual tokens remain unchanged, preserving the feature distribution expected by the pretrained MLLM.

Global camera self-attention. Global camera self-attention then allows the per-frame camera tokens to exchange information. Each token can therefore interpret its frame relative to the surrounding viewpoints rather than in isolation:

$$[c_1^{(m)}, \ldots, c_T^{(m)}] = \mathrm{SelfAttn}\big([\tilde{c}_1^{(m)}, \ldots, \tilde{c}_T^{(m)}]\big). \tag{3}$$

This operation attends over only the T camera tokens, not the full set of $\sum_i P_i$ visual patches. Both attention modules use standard pre-norm transformer blocks (Vaswani et al., 2017; Ba et al., 2016).

Output and injection. For frame i, we concatenate the final frame-level state and the temporally contextualized state, $z_i = [\tilde{c}_i^{(M)}; c_i^{(M)}]$. A two-layer MLP projects this representation to the LLM hidden size. The projected token is placed immediately before the visual tokens of frame i, making the camera representation available as context before the decoder processes the frame content. Appendix K provides the complete block specification.

4.3 DISTILLATION OBJECTIVE

CAMDISTILL is trained with two objectives (Figure 3b): the standard next-token loss $\mathcal{L}_{\mathrm{SFT}}$ for the structured task output, and a camera-token *distillation loss* $\mathcal{L}_{\mathrm{cam}}$ (Hinton et al., 2015) that aligns

each student token z_i with the teacher's target token g_i by cosine distance:

$$\mathcal{L} = \mathcal{L}_{\text{SFT}} + \lambda_{\text{cam}}\,\mathcal{L}_{\text{cam}} = \mathcal{L}_{\text{SFT}} + \frac{\lambda_{\text{cam}}}{T}\sum_{i=1}^{T}\big(1 - \cos(z_i, g_i)\big), \tag{4}$$

where λ_{cam} weights the distillation term. Through $\mathcal{L}_{\text{cam}}$, the student tokens are encouraged to reproduce the teacher's pose-associated camera representation.

5 EXPERIMENTS

5.1 SETUP

Models. We evaluate all models on CAMCHOREO using the same instruction and output format. The comparison includes four groups. *Closed-source APIs* comprise GPT-5.4 (Singh et al., 2025) and Gemini-3.1-Pro (Pichai et al., 2025). *Open-source MLLMs* include Qwen2.5-VL (Bai et al., 2025c), Qwen3-VL at 4B, 8B, and 235B (Bai et al., 2025b), Qwen3.5 (Qwen Team, 2026a), Qwen3.6 (Qwen Team, 2026b), InternVL3.5 (Wang et al., 2025c), and Cam-Motion-7B, a Qwen2.5-VL model fine-tuned on CameraBench (Lin et al., 2026). The *geometry-only baseline* estimates per-frame camera pose with VGGT-Ω and maps translational and angular velocities to labels using hand-designed rules (Appendix I). Finally, *our models* are CAMDISTILL and the direct-injection reference CAMINJECT, implemented with Qwen3-VL 4B and 8B backbones and VGGT-Ω as the default 3D teacher. For our models, we fully fine-tune the language model, freeze the vision encoder, and train GCTE jointly. The SFT baseline fully fine-tunes the same base model with neither the GCTE module nor the distillation loss. All specialized models are trained on 43,438 clips from Tencent Video, a public online video platform, annotated with the same taxonomy and protocol as CAMCHOREO, with 1,000 clips held out for validation. The training set is disjoint from the benchmark, ensuring that no evaluation clip is observed during training. Appendix G provides the training configuration.

Metrics. We evaluate predictions from two complementary perspectives (Appendix E). *Frame-level* evaluation samples the timeline every 0.1 s and computes precision, recall, and F1 over the 20 direction-aware labels. We report both micro averages, which weight instances equally, and macro averages, which weight classes equally. Because labels are evaluated at each timestamp, these metrics capture both boundary and recognition errors. *Segment-level* evaluation instead matches predicted and ground-truth intervals by temporal IoU and reports F1 at thresholds $0.3/0.5/0.7$. Segment localization (SegLoc) evaluates temporal overlap without considering labels, whereas segment detection (SegDet) additionally requires an exact match of the direction-aware label set.

5.2 MAIN RESULTS

Table 2 shows that existing models struggle on CAMCHOREO, whereas CAMDISTILL and CAMINJECT lead by a wide margin. Scaling Qwen3-VL from 4B to 235B increases frame-level micro F1 from 24.2 to only 33.4. The strongest closed-source model, Gemini-3.1-Pro, reaches 43.3. In contrast, CAMDISTILL achieves 67.5 with the 4B backbone and 67.8 with the 8B backbone; CAMINJECT obtains comparable results. Both approaches therefore exceed the strongest baseline by more than 20 micro-F1 points.

The geometry-only baseline clarifies where the difficulty lies. At IoU 0.5, it obtains 63.9 SegLoc but only 2.9 SegDet. Estimated pose can indicate *when* camera behavior changes, but cannot identify some movements, such as Zoom and Focus Shift. More generally, all models perform substantially better on SegLoc than on SegDet. The central challenge is therefore not merely locating temporal boundaries, but recovering the complete set of motion labels. Cam-Motion-7B also transfers poorly: CameraBench clip-level tuning overfits its base MLLM and erodes instruction-following, yielding valid outputs for only 19 of 4,229 clips. Its scores, computed over these 19 alone, are not comparable to other rows. This reflects both the format gap and the cost of narrow task-specific tuning.

The comparison between CAMDISTILL and CAMINJECT isolates the effect of replacing direct teacher features with distilled ones. CAMINJECT retains the 3D model at inference, whereas CAMDISTILL predicts camera tokens from the frozen MLLM features and removes the teacher. Nevertheless, their results are nearly identical. With the 4B backbone, both reach 67.5 micro F1 and differ by only 0.4 SegDet at IoU 0.5. With the 8B backbone, CAMDISTILL trails CAMINJECT by

Table 2. Results on CAMCHOREO. Frame-level micro/macro precision, recall, and F1 are evaluated every 0.1 seconds. SegLoc measures temporal overlap; SegDet additionally requires an exact direction-aware label set. Both are reported at IoU $0.3/0.5/0.7$. Per column, best is in **bold** and second-best is underlined.

	Frame-Level (%)						Segment-Level F1 (%)					
	Micro			Macro			SegLoc @IoU			SegDet @IoU		
Model	P	R	F1	P	R	F1	0.3	0.5	0.7	0.3	0.5	0.7
VGGT-Ω	21.7	30.2	25.3	23.3	26.2	17.2	73.6	63.9	48.5	3.0	2.9	2.7
Gemini-3.1-Pro	51.4	37.4	43.3	40.5	23.0	27.4	82.8	76.1	61.2	25.1	23.7	19.8
GPT-5.4	43.8	33.5	38.0	31.2	19.2	21.3	82.8	76.1	62.1	23.9	22.4	18.9
InternVL3.5-8B	31.9	18.7	23.6	16.9	6.6	7.1	38.4	21.0	8.5	7.2	4.0	1.7
Qwen2.5-VL-7B	27.9	17.8	21.8	8.4	4.1	4.0	62.6	39.7	18.2	9.9	6.4	3.0
Cam-Motion-7B	12.6	11.2	11.9	3.4	2.4	2.6	53.2	38.0	20.3	0.0	0.0	0.0
Qwen3-VL-4B	32.4	19.3	24.2	14.9	7.1	7.8	69.7	55.6	38.8	14.4	11.6	8.6
Qwen3-VL-8B	36.6	23.0	28.3	17.3	7.9	9.1	76.2	67.7	52.5	16.5	15.0	12.5
Qwen3-VL-235B	39.7	28.8	33.4	22.0	13.9	15.4	79.1	66.1	47.8	17.9	15.6	12.0
Qwen3.5-4B	46.0	14.7	22.3	20.6	5.1	6.9	52.1	46.9	38.2	16.8	15.4	12.9
Qwen3.5-9B	50.5	13.7	21.5	21.9	4.5	6.5	43.8	38.3	29.7	14.9	13.3	10.7
Qwen3.6-35B	46.4	29.3	35.9	22.3	12.2	14.3	80.1	68.8	51.2	19.8	17.9	14.2
CAMDISTILL-4B	73.5	62.4	67.5	63.8	54.5	57.7	86.4	80.4	66.8	39.7	38.2	33.5
CAMINJECT-4B	73.0	62.8	67.5	63.5	54.7	57.6	**86.7**	**80.8**	66.8	40.1	38.6	33.9
CAMDISTILL-8B	73.8	62.6	67.8	63.6	54.9	57.9	86.5	80.4	66.7	**40.5**	38.8	34.1
CAMINJECT-8B	**74.0**	**63.5**	**68.3**	**64.9**	**56.3**	**59.2**	86.5	**80.8**	**67.0**	40.3	**39.0**	**34.2**

Table 3. Camera information on Qwen3-VL-4B. We add pose-as-text prompting (PromptInject), supervised fine-tuning (SFT), and our CAMDISTILL and CAMINJECT to the base model. PromptInject and CAMINJECT run VGGT-Ω at inference, and latency and peak memory are measured on a single H100. The 8B backbone shows the same trends (Table 8).

Method	Micro F1	Macro F1	SegLoc@0.5	SegDet@0.5	Latency (s/clip)↓	Peak Mem (GB)↓
Qwen3-VL-4B	24.2	7.8	55.6	11.6	**10.1**	**18.3**
+PromptInject	23.1	10.7	67.3	12.1	16.8	24.0
+SFT	62.2	51.4	79.0	33.6	**10.1**	**18.3**
+CamDistill	**67.5**	**57.7**	80.4	38.2	10.2	20.1
+CamInject	**67.5**	57.6	**80.8**	**38.6**	16.0	23.1

only 0.5 micro F1. Thus, CAMDISTILL preserves almost all of the benefit of direct feature injection without requiring the 3D model at inference.

5.3 ANALYSIS AND ABLATIONS

Effect of camera information. As shown in Table 3, feeding the teacher's per-frame pose as textual prompt (PromptInject) helps localization (SegLoc@0.5 $55.6 \rightarrow 67.3$) but not recognition (micro F1 $24.2 \rightarrow 23.1$), and still runs VGGT-Ω at inference (latency $10.1 \rightarrow 16.8$ s). SFT is far more effective at 62.2 micro F1, but task supervision alone leaves the frozen encoder unable to separate motions that differ only in parallax or perspective. Distilling the teacher's camera representation into GCTE closes much of this gap, adding 5.3 micro-F1, 6.3 macro-F1, and 4.6 SegDet@0.5 over SFT. The larger macro gain shows the geometric signal especially helps rare, geometry-dependent classes, and Figure 4 attributes it to the supervision itself, since performance peaks at a nonzero distillation weight. Crucially, CAMDISTILL reaches this accuracy without the teacher at inference. It matches CAMINJECT within 0.4 SegDet but adds only 0.1 s and 1.8 GB over the base model, against CAMINJECT's 5.9 s and 4.8 GB. CAMDISTILL thus attains injection-level accuracy at essentially the base model's inference cost, and the 8B backbone shows the same pattern (Table 8).

Generalization to external benchmarks. We next test whether CAMDISTILL learns transferable camera-motion cues rather than merely adapting to the output format of CAMCHOREO. CameraBench (Lin et al., 2026) evaluates clip-level recognition on real web videos using mAP, while CMVQA (Feng et al., 2026) evaluates multiple-choice reasoning on synthetic clips using accuracy. We follow each benchmark's official evaluation script and task protocol, so improvements provide evidence of cross-task transfer. As shown in Table 4, CAMDISTILL-8B improves over Qwen3-VL-8B by 20.7 mAP on CameraBench and 16.6 accuracy points on CMVQA. It also outperforms

Table 4. External generalization. mAP on CameraBench and accuracy on CMVQA, two external camera-motion benchmarks.

Model	CameraBench	CMVQA
Qwen2.5-VL-7B	31.0	24.8
Qwen3-VL-8B	39.6	23.5
Cam-Motion-7B	49.7	29.7
CAMDISTILL-8B	<u>60.3</u>	<u>40.1</u>
CAMINJECT-8B	**62.9**	**42.4**

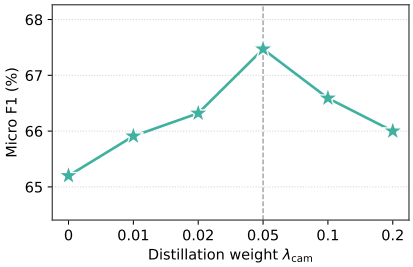


Figure 4. Distillation-weight sensitivity. CAMDISTILL-4B peaks at $\lambda_{\text{cam}} = 0.05$.

Table 5. Ablations of CAMDISTILL-4B. Each block varies one design factor, namely feature-layer region, module depth, token position, and teacher, while the others stay at the default. Segment metrics use IoU 0.5.

Component	Setting	Micro F1	Macro F1	SegLoc	SegDet
Feature-layer region	Early $(0, 3, 6, 9)$	66.1	56.0	80.0	37.3
	Late $(14, 17, 20, 23)$	65.4	54.9	79.6	36.8
	Uniform $(4, 9, 13, 18)$	66.8	56.5	79.9	37.6
	Early–middle $(1, 5, 9, 13)$	**67.5**	**57.7**	**80.4**	**38.2**
Module depth	3 layers $(1, 7, 13)$	66.6	56.2	79.8	37.4
	4 layers $(1, 5, 9, 13)$	**67.5**	**57.7**	**80.4**	**38.2**
	5 layers $(1, 4, 7, 10, 13)$	67.2	57.5	80.3	38.2
Token position	After visual tokens	64.7	54.5	79.5	36.9
	Before visual tokens	**67.5**	**57.7**	**80.4**	**38.2**
Teacher	VGGT	65.8	55.7	80.2	37.6
	VGGT-Ω	**67.5**	**57.7**	**80.4**	**38.2**

CameraBench-tuned Cam-Motion-7B on both benchmarks. The remaining gap to CAMINJECT is small, consistent with the modest information loss introduced by distillation.

Sensitivity to the distillation weight. Figure 4 shows that performance peaks at $\lambda_{\text{cam}} = 0.05$. With a smaller weight, the geometric supervision is too weak to add much beyond task training. With a larger weight, feature imitation competes with the language-modeling objective. The distillation loss is therefore most effective as a moderate auxiliary signal rather than the dominant target.

Design ablations. Table 5 varies one GCTE design choice at a time on the 4B backbone. *Feature layers*: early-to-middle features perform best, suggesting that they retain geometric detail while providing sufficient contextual abstraction. Earlier features contain less context, whereas later features are increasingly semantic. *Module depth*: four alternating blocks achieve the best trade-off. Three blocks are insufficient to reproduce the teacher representation, while a fifth provides no meaningful gain. *Token position*: placing camera tokens before the visual tokens improves micro F1 by 2.8 points, indicating that they are most useful as conditioning context rather than appended summaries. *Teacher*: replacing VGGT with VGGT-Ω improves micro F1 by 1.7 points, even though the teacher is absent at inference. Together with Figure 4, these results attribute the gains to the camera-specific design and supervision rather than to an arbitrary increase in model capacity.

6 CONCLUSION

We study camera motion as a temporally grounded, compositional problem, in which a model must localize when each movement occurs and recover the movements that co-occur within a shot. Building the CAMCHOREO benchmark for this task shows that such temporal and compositional structure is the rule rather than the exception in real video, and that current MLLMs struggle on it because their visual encoders lack the geometric grounding it requires. To close this gap, we introduce CAMDISTILL, which distills the geometry of a 3D foundation model into lightweight camera tokens during training and discards the teacher at inference. It matches the accuracy of direct geometric injection while adding almost no inference cost. We hope CAMCHOREO and CAMDISTILL encourage modeling camera motion as a time-varying, compositional signal.

REPRODUCIBILITY STATEMENT

We provide the information needed to reproduce the benchmark and experiments. Appendix C describes dataset construction and filtering, Appendix D specifies the taxonomy and annotation protocol, and Appendix E defines the evaluation metrics. Appendix G lists the optimization settings, tapped layers, and hardware for both backbones. Appendix K gives the complete GCTE equations, and Appendix F provides the training and inference prompt. We will release the CAMCHOREO annotations, evaluation code, prompts, trained checkpoints, and permitted video identifiers or download scripts.

ETHICS STATEMENT

CAMCHOREO consists of publicly available single-shot YouTube clips labeled by trained expert annotators and is intended solely for research on camera-motion understanding. The separate training videos are collected from Tencent Video, a publicly accessible online video platform. Broader implications and limitations are discussed in Appendices A and B.

AI USE STATEMENT

We used generative AI tools only as general-purpose assistants for editing prose and for minor coding support (e.g., plotting and data-processing scripts). Generative AI was not used to generate research ideas, experimental results, or data annotations. All AI-assisted text and code were reviewed and verified by the authors, who take full responsibility for the final content of this work.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, et al. Recammaster: Camera-controlled generative rendering from a single video. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14834–14844. IEEE, 2025a.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025b.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025c.

Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

Siyan Dong, Shuzhe Wang, Shaohui Liu, Lulu Cai, Qingnan Fan, Juho Kannala, and Yanchao Yang. Reloc3r: Large-scale training of relative camera pose regression for generalizable, fast, and accurate visual localization. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16739–16752. IEEE, 2025.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pp. 834–849. Springer, 2014.

Haoan Feng, Sri Harsha Musunuri, and Guan-Ming Su. Geometry-guided camera motion understanding in videollms. *arXiv preprint arXiv:2603.13119*, 2026.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

Moyang Li, Zihan Zhu, Marc Pollefeys, and Daniel Barath. Droid-slam in the wild. *arXiv preprint arXiv:2603.19076*, 2026.

Xiaozhe Li, Kai Wu, Siyi Yang, YiZhan Qu, Guohua Zhang, Zhiyu Chen, Jiayao Li, Jiangchuan Mu, Xiaobin Hu, Wen Fang, et al. Can video generation replace cinematographers? research on the cinematic language of generated video. *arXiv preprint arXiv:2412.12223*, 2024b.

Zhiqiu Lin, Siyuan Cen, Daniel Jiang, Jay Karhade, Hewei Wang, Chancharik Mitra, Yu Tong Tiffany Ling, Yuhan Huang, Rushikesh Zawar, Xue Bai, et al. Towards understanding camera motions in any video. *Advances in Neural Information Processing Systems*, 38, 2026.

Hongbo Liu, Jingwen He, Yi Jin, Dian Zheng, Yuhao Dong, Fan Zhang, Ziqi Huang, Yinan He, Weichao Chen, Yu Qiao, et al. Shotbench: Expert-level cinematic understanding in vision-language models. *Advances in Neural Information Processing Systems*, 38:129987–130019, 2026.

Jakob Isak Nielsen, Edvin Kau, and Richard Raskin. *Camera movement in narrative cinema: towards a taxonomy of functions*. Department of Inf. & Media Studies, University of Aarhus, 2007.

Sundar Pichai, Demis Hassabis, and Koray Kavukcuoglu. A new era of intelligence with gemini 3, 2025. URL https://blog.google/intl/en-africa/company-news/outreach-and-initiatives/a-new-era-of-intelligence-with-gemini-3/.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Tomás Soucek and Jakub Lokoc. Transnet v2: An effective deep network architecture for fast shot transition detection. In *Proceedings of the 32nd ACM international conference on multimedia*, pp. 11218–11221, 2024.

Raymond Spottiswoode. *A grammar of the film: An analysis of film technique*. Univ of California Press, 1959.

Yunlong Tang, Junjia Guo, Hang Hua, Susan Liang, Mingqian Feng, Xinyang Li, Rui Mao, Chao Huang, Jing Bi, Zeliang Zhang, et al. Vidcomposition: Can mllms analyze compositions in compiled videos? In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8490–8500. IEEE, 2025.

Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jiahao Wang, Yufeng Yuan, Rujie Zheng, Youtian Lin, Jian Gao, Lin-Zhuo Chen, Yajie Bao, Yi Zhang, Chang Zeng, Yanxi Zhou, et al. Spatialvid: A large-scale video dataset with spatial annotations. *arXiv preprint arXiv:2509.09676*, 2025a.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025b.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-ω. *arXiv preprint arXiv:2605.15195*, 2026a.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709. IEEE, 2024.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025c.

Xinran Wang, Songyu Xu, Shan Xiangxuan, Yuxuan Zhang, Muxi Diao, Xueyan Duan, Kongming Liang, Zhanyu Ma, et al. Cinetechbench: A benchmark for cinematographic technique understanding and generation. *Advances in Neural Information Processing Systems*, 38, 2026b.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, volume 2026, pp. 10481–10497, 2026c.

Hang Wu, Yujun Cai, Haonan Ge, Hongkai Chen, Ming-Hsuan Yang, and Yiwei Wang. Refineshot: Rethinking cinematography understanding with foundational skill evaluation. *arXiv preprint arXiv:2510.02423*, 2025a.

Hang Wu, Yujun Cai, Zehao Li, Haonan Ge, Bowen Sun, Junsong Yuan, and Yiwei Wang. Camreasoner: Reinforcing camera movement understanding via structured spatial reasoning. *arXiv preprint arXiv:2602.00181*, 2026.

Jianlong Wu, Wei Liu, Ye Liu, Meng Liu, Liqiang Nie, Zhouchen Lin, and Chang Wen Chen. A survey on video temporal grounding with multimodal large language model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Jinbo Xing, Long Mai, Cusuh Ham, Jiahui Huang, Aniruddha Mahapatra, Chi-Wing Fu, Tien-Tsin Wong, and Feng Liu. Motioncanvas: Cinematic shot design with controllable image-to-video generation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–11, 2025.

Jihan Yang, Zifan Zhao, Xichen Pan, Shusheng Yang, Junyi Zhang, Bingyi Kang, Hu Xu, and Saining Xie. Cambrian-p: Pose-grounded video understanding. *arXiv preprint arXiv:2605.22819*, 2026.

Mehmet Burak Yilmaz, Elen Lotman, Andres Karjus, and Pia Tikka. An embodiment of the cinematographer: emotional and perceptual responses to different camera movement techniques. *Frontiers in Neuroscience*, 17:1160843, 2023.

Gongjie Zhang, Wenhao Li, Quanhao Qian, Jiuniu Wang, Deli Zhao, Shijian Lu, and Ran Xu. On the generalization capacities of mllms for spatial intelligence. *arXiv preprint arXiv:2603.06704*, 2026.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T Freeman. Structure and motion from casual videos. In *European Conference on Computer Vision*, pp. 20–37. Springer, 2022.

Duo Zheng, Yanyang Li, Liwei Wang, et al. Learning from videos for 3d world: Enhancing mllms with 3d vision geometry priors. *Advances in neural information processing systems*, 38:20560–20586, 2026.

A Broader Implications and Reproducibility

Broader implications. Camera-motion understanding describes the behavior of the observer, not only the content of the observed scene. Because camera state affects depth cues, visibility, and the interpretation of object motion, camera-aware representations may benefit spatial reasoning, action understanding, video retrieval, and controllable video generation. These applications also require caution. Cinematic labels reflect production conventions and can remain ambiguous across domains, so downstream systems should preserve uncertainty rather than treat every prediction as an objective description of authorial intent.

Reproducibility. The training set is disjoint from the independently collected benchmark. The appendices specify the taxonomy, annotation instructions, prompts, frame- and segment-level metrics, validation-based model-selection protocol, teacher alignment, and inference-cost protocol. We will release the benchmark annotations, permitted video identifiers or download scripts, evaluation code, and checkpoints where licensing allows.

B Limitations

CamChoreo is restricted to single-shot clips and therefore does not cover multi-shot videos or interactions among camera motion, editing, and long-form narrative structure. Extending the task and benchmark to edited, multi-shot video is left to future work.

C Benchmark Construction Pipeline

Figure 5 summarizes the complete curation pipeline, from source discovery and automatic filtering to class balancing and human quality control.

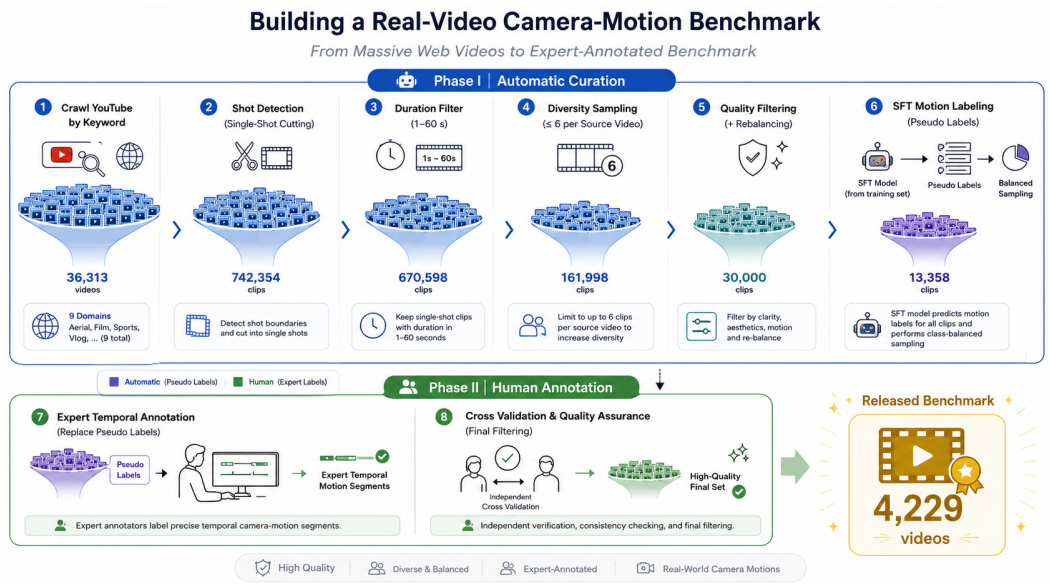


Figure 5. CamChoreo construction pipeline. *Phase I: automatic curation.* We collect YouTube videos from nine domains, split them into shots, filter for duration, diversity, and quality, and use pseudo labels only to balance the candidate pool. This process reduces 36,313 source videos to 13,358 candidate clips. *Phase II: human annotation.* Expert annotators replace all pseudo labels with temporal camera-motion annotations. Independent review and final filtering produce 4,229 released clips.

Source collection. We query YouTube using a curated keyword set that covers all nine content domains, with roughly a dozen queries per domain. Examples include *cinematic footage* and *film clip 4k* for Film & TV; *wildlife documentary* and *aerial nature footage* for Documentary & Nature; *FPV racing drone* and *drone orbit building* for Aerial & Drone; *walking vlog* and *handheld walking*

footage for Vlog & Selfie; and *product commercial* and *car advertisement* for Commercial & Ad. We use analogous queries for Sports & Action, Gaming & Animation, Tutorial & Education, and Synthetic & Rendered.

Filtering funnel. Figure 5 contains eight stages. **(1)** We retain source videos between 2 s and 30 min. **(2)** TransNetV2 (Soucek & Lokoc, 2024) detects shot boundaries and divides each video into single-shot clips. **(3)** We keep clips between 1 and 60 s. **(4)** At most six clips are retained from each source video to improve content diversity. **(5)** We re-encode all clips with a uniform `ffmpeg` profile and score them on clarity, aesthetics, motion intensity, motion quality, and content quality. Clips below threshold on any dimension are removed, while a controlled fraction of near-static clips is restored to preserve the Static prior. **(6)** An SFT camera-motion model trained on the separate 43,438-video training set assigns *pseudo* labels. These labels are used only to construct a balanced candidate pool that up-weights rare movements such as Zoom, Roll, Arc, and Focus Shift. **(7)** Five annotators with film- and media-related backgrounds independently replace the pseudo labels with precise temporal annotations. **(8)** Cross-review resolves disagreements by consensus; clips without consensus, as well as doubtful or empty clips, are removed. The resulting benchmark contains 4,229 clips.

D Taxonomy and Annotation Rules

Table 6. Camera motion taxonomy used in CamChoreo. cw/ccw abbreviate clockwise/counterclockwise.

Type	Direction	Annotation cue
Static	None	Camera position and orientation remain essentially unchanged; imperceptible micro-jitter is allowed.
Unstable	None	Irregular visible shaking without a stable direction; if a stable direction exists, annotate that motion instead.
Pan	left/right	Horizontal rotation around a fixed camera position; foreground and background move similarly with little parallax.
Tilt	up/down	Vertical rotation around a fixed camera position; frame shifts vertically without translation parallax.
Truck	left/right	Lateral camera translation; foreground/background parallax is visible.
Pedestal	up/down	Vertical camera translation; perspective and horizon height change.
Dolly In/Out	None	Forward/backward camera translation with depth parallax; distinct from zoom.
Zoom In/Out	None	Focal-length change with approximately uniform image scaling and no depth parallax.
Roll	cw/ccw	Rotation around the optical axis; horizon tilts.
Arc	cw/ccw	Camera orbits around a subject or scene center.
Follow	None	Camera tracks a moving subject; requires reasoning about subject-centered motion.
Focus Shift	None	Focus plane changes while camera motion may be absent.

Annotation protocol. The taxonomy contains 12 movement types and 20 direction-aware labels. Directions follow the physical camera motion rather than the apparent background flow: Pan Left/Right denotes rotation of the viewing direction, Truck Left/Right translation of the camera center in its local frame, and Tilt/Pedestal Up/Down the corresponding physical rotation/translation. Each movement also receives a coarse speed attribute (`zero`, `slow`, `medium`, or `fast`), although speed is not evaluated in this work. Annotators apply seven main rules. (i) Each segment contains all basic movements with perceptible magnitude and clear intent, while minor compensatory motion is ignored. (ii) Direction is constrained by movement type: Pan and Truck use left/right, Tilt and Pedestal use up/down, Arc and Roll use clockwise/counterclockwise, and all remaining types use `null`. A movement with two directional components is represented by two elements, such as Dolly In + Tilt up. (iii) Static permits only imperceptible micro-jitter, while clearly visible directionless shake is labeled Unstable. (iv) Dolly is distinguished from Zoom through depth parallax and perspective change, rather than uniform image scaling. (v) Focus Shift, including rack and follow focus, is treated as a basic movement. (vi) Follow and Arc are annotated together with their

underlying primitive, such as Dolly In, Truck, or Pan. Arc requires the camera to move along a clear curved trajectory around one or more identifiable subjects through at least $45°$; weak curvature below this threshold or motion without a locked subject is treated as a minor adjustment and is not labeled Arc. Arc clockwise/counterclockwise is defined by the camera trajectory around the subject as viewed from above. (vii) Subject motion is not labeled as camera motion. For example, a walking person does not imply camera movement unless the background perspective or frame boundaries also change. Segment boundaries are placed at 0.1-second resolution whenever the active motion phase changes.

E Evaluation Protocol Details

We formalize the frame- and segment-level metrics summarized in the main text.

Setup. Each video is represented as a sequence of non-overlapping temporal segments that fully covers $[0, T]$, and every segment carries a *set* of camera-movement labels. Directional movements are encoded jointly with their direction, for example `Pan_left`, while non-directional movements use only the type name. This produces the 20-label space $\mathcal{C}$ used throughout the paper. Metrics are computed on videos present in both the predictions and the ground truth (GT).

E.1 Frame-Level Metrics

Sampling. We sample each timeline at intervals of $\Delta = 0.1\,\text{s}$. At timestamp t, let $Y_t \subseteq \mathcal{C}$ and $\hat{Y}_t \subseteq \mathcal{C}$ denote the GT and predicted label sets of the segments covering t, where a segment is active when $s \le t < e$. Because the GT segments cover $[0, T]$, evaluation samples the full GT timeline; a missing predicted segment yields an empty predicted label set and therefore false negatives. Because labels are evaluated densely over time, a boundary error affects multiple timestamps and is reflected in the recognition score.

Micro precision/recall/F1. We accumulate multi-label counts over all sampled frames,

$$\text{TP} = \sum_t |Y_t \cap \hat{Y}_t|, \quad \text{FP} = \sum_t |\hat{Y}_t \setminus Y_t|, \quad \text{FN} = \sum_t |Y_t \setminus \hat{Y}_t|, \tag{5}$$

and define $P = \text{TP}/(\text{TP}+\text{FP})$, $R = \text{TP}/(\text{TP}+\text{FN})$, and $\text{F1} = 2PR/(P+R)$. Micro averaging is instance-weighted and is therefore dominated by frequent classes.

Macro precision/recall/F1. We instead compute per-class counts $\text{TP}_c, \text{FP}_c, \text{FN}_c$ for each $c \in \mathcal{C}$ (a frame contributes to class c as TP if $c \in Y_t \cap \hat{Y}_t$, FP if $c \in \hat{Y}_t \setminus Y_t$, FN if $c \in Y_t \setminus \hat{Y}_t$), form the per-class $P_c, R_c, \text{F1}_c$, and average them equally over $\mathcal{C}$. Macro averaging is class-weighted and surfaces rare classes such as Roll and Arc. A direction-agnostic variant that collapses each composite label to its type isolates direction errors.

E.2 Segment-Level Metrics

Temporal IoU and matching. This axis operates on whole segments and *decouples* localization from recognition. For a GT segment g and predicted segment p, the temporal IoU is

$$\text{IoU}(g, p) = \frac{\max\big(0,\ \min(g^{\text{end}}, p^{\text{end}}) - \max(g^{\text{start}}, p^{\text{start}})\big)}{|g| + |p| - |g \cap p|}. \tag{6}$$

Given a threshold τ, we form the pairwise temporal-IoU matrix and use Hungarian matching to obtain a one-to-one assignment between ground-truth and predicted segments. Assigned pairs with $\text{IoU} \ge \tau$ are retained as matches. Let N_G and N_P be the number of GT and predicted segments.

Localization (SegLoc). A predicted segment is a true positive iff it is matched with $\text{IoU} \ge \tau$, *regardless of labels*. With M_{loc} matched pairs, $P_{\text{loc}} = M_{\text{loc}}/N_P$, $R_{\text{loc}} = M_{\text{loc}}/N_G$, and SegLoc-F1 $= 2P_{\text{loc}}R_{\text{loc}}/(P_{\text{loc}} + R_{\text{loc}})$. This measures pure temporal segmentation quality.

Detection (SegDet). Segment detection additionally requires the predicted and GT label sets to match exactly, $L_p = L_g$. Let M_{det} denote the number of matched pairs satisfying this condition. We define $P_{\text{det}} = M_{\text{det}}/N_P$, $R_{\text{det}} = M_{\text{det}}/N_G$, and compute SegDet-F1 analogously. Because $M_{\text{det}} \leq M_{\text{loc}}$, SegDet-F1 cannot exceed SegLoc-F1 at the same threshold. SegDet assigns no partial credit within a segment: a direction error or a missing co-occurring movement invalidates the match. It is therefore substantially stricter than frame-level F1.

Thresholds. All segment-level metrics are reported at $\tau \in \{0.3, 0.5, 0.7\}$ without averaging, exposing the localization-tightness trade-off. Matching is deterministic, using Hungarian assignment on the temporal-IoU matrix.

F Training and Inference Prompt

We use an identical prompt at training and inference. The model receives a video, the system prompt below, and a short user instruction, and is required to return only JSON. The evaluator tolerates minor formatting variations but enforces the closed taxonomy, valid directions, chronological ordering, and non-overlapping temporal segments.

System Prompt

You are a senior film cinematographer. After watching the video, determine which camera movements make up this video, locate their time spans, and output structured JSON.
Core principle: judge only the motion of the camera (lens) itself, not the motion of objects within the frame. People walking or cars driving inside the frame do not mean that the camera is moving. Observe whether the *background and frame edges* move.

Basic movement (required, array). Each element is {`"type", "direction", "speed"`}, with `type` drawn from the closed set below.
Static / non-steady. `Static`: camera position and orientation essentially unchanged (barely visible micro-jitter allowed); `speed=zero` if completely still, else `slow`. `Unstable`: clearly perceptible irregular shaking with no stable direction; if a sustained direction exists (e.g. the background sweeps left), label that movement instead (e.g. Pan).
Rotation vs. translation. `Pan` (fixed camera, horizontal rotation; no depth parallax) vs. `Truck` (lateral camera translation; obvious parallax). `Tilt` (fixed camera, vertical rotation) vs. `Pedestal` (vertical translation; horizon height and pitch change).
Depth. `Dolly In`/`Dolly Out` (camera moves forward/backward; near and far regions scale at different rates because of parallax) vs. `Zoom In`/`Zoom Out` (focal-length change, camera fixed; uniform scaling, no parallax).
Other. `Roll` (rotation about the optical axis; tilting horizon), `Arc` (camera orbits a centered subject), `Follow` (camera tracks a moving subject; background changes continuously), `Focus Shift` (focus moves across depth layers). Arc and Follow must also annotate the underlying basic movements (e.g. Truck, Pan, Dolly In).

Direction rules. `Truck`→left/right, `Pedestal`→up/down, `Pan`→left/right, `Tilt`→up/down, `Arc`/`Roll`→clockwise/counterclockwise; all other types use `null`.
Speed rules. `zero` (completely still) / `slow` (confirmable only on careful inspection) / `medium` (clearly perceived) / `fast` (rapid, with a sense of speed).
Compound movement. When several motions occur simultaneously and are all observable, output all items (e.g. Dolly In while Pan). Output the `segments` array ordered by time, following the format below, and output *only* JSON.

Output Format: Example JSON

```
{
  "segments": [
    {
      "start_time": 0.0,
      "end_time": 2.5,
      "basic_movement": [
        {"type": "Truck", "direction": "right", "speed": "medium"},
        {"type": "Dolly In", "direction": null, "speed": "slow"}
      ],
      "confidence": "high"
    },
    {
      "start_time": 2.5,
      "end_time": 5.0,
      "basic_movement": [
        {"type": "Static", "direction": null, "speed": "zero"}
      ],
      "confidence": "high"
    }
  ]
}
```

User Prompt

Analyze the camera movement in this video and output JSON following the system prompt rules. Output only JSON.

G Additional Implementation Details

The camera module uses QK-normalized scaled dot-product attention, pre-normalization, residual connections, and small LayerScale coefficients. The first-frame and shared subsequent-frame queries are initialized from a zero-mean Gaussian with standard deviation 10^{-3}. For each video, the teacher cache stores an $S \times 2048$ tensor, where S is the number of teacher frames. Before computing the loss, these features are aligned to the MLLM temporal grid. CamInject loads the same teacher features at inference and maps them to the LLM hidden size with a two-layer projector of approximately 8M parameters. CamDistill accesses the cache only during training.

Training data. SFT, CamDistill, and CamInject are trained on 43,438 clips from the Tencent Video platform, with 1,000 clips held out for validation. The training annotations use the same 12 movement types, 20 direction-aware labels, 0.1-second temporal resolution, and annotation protocol as CamChoreo (Appendix D). The data source is the main difference: the training clips come from Tencent Video, whereas the benchmark clips come from YouTube. The two sets are disjoint. We additionally screen them with CLIP embeddings and manually inspect high-similarity pairs; no duplicate clip is retained.

Evaluation inputs. Qwen-family open-source models, Cam-Motion-7B, SFT, CamDistill, and CamInject use 5 FPS with at most 100 frames. They share the same English prompt, deterministic decoding, parser, and evaluator. Gemini receives the video through its video interface with a requested 5-FPS rate. GPT-5.4 does not accept video through the evaluated API and therefore receives uniformly sampled timestamped frames (5 FPS). InternVL3.5 uses 32 uniformly sampled frames because its inference interface accepts a fixed frame count. External CameraBench and CMVQA results use the benchmarks' official scripts and protocols without our temporal JSON prompt.

Training configuration. Table 7 gives the complete setup for both backbones. We fully fine-tune the language model while freezing the vision encoder and visual projector. We train GCTE jointly with the language model. For CamInject, the teacher projector is also trainable. GCTE adds approximately 110.2M trainable parameters to the 4B backbone and 216.3M to the 8B backbone. Block i attends to the i-th selected ViT layer, so module depth equals the number of tapped layers.

When the number of teacher frames differs from the number of MLLM temporal groups, we align the teacher sequence using two-frame average pooling when possible, followed by adaptive pooling or nearest-neighbor interpolation when necessary. Because the 3D model provides only supervision for CAMDISTILL, we extract its camera tokens once and cache them offline. The teacher is not invoked in the optimization loop, so distillation adds little runtime overhead beyond standard SFT. For CAMINJECT, the teacher can alternatively be run online for each batch.

Table 7. Training hyperparameters. Shared by CAMDISTILL and CAMINJECT unless a per-backbone value is given.

Setting	Value
Model and optimization	
Backbone	Qwen3-VL-4B / 8B-Instruct
Trainable parameters	full LLM; GCTE module
Frozen parameters	vision encoder, visual projector
Precision / attention	bfloat16 / FlashAttention-2
Memory	gradient checkpointing, DeepSpeed ZeRO-2
GPUs	8× H20
Per-device batch size	2
Gradient accumulation	4 (effective batch size 64)
Epochs	2
Learning rate (4B / 8B)	2×10^{-5} / 1.5×10^{-5}
LR schedule	cosine, warm-up ratio 0.05
Weight decay	0.01
Max sequence length	16,384
Training time (4B / 8B)	$\sim$7 / $\sim$10 hours
Video sampling	
Frame rate / count	5 FPS, 4–100 frames
Max pixels per frame	100,352
Camera-token module and distillation	
GCTE depth M (4B / 8B)	4 / 8
Tapped ViT layers (4B)	$\{1, 5, 9, 13\}$ over 24 layers
Tapped ViT layers (8B)	$\{1, 3, 5, 7, 9, 11, 13, 15\}$
Camera-token dim d_c	1024 (concatenated feature 2048)
Distillation weight λ_{cam}	0.05
Teacher	3D foundation model (VGGT / VGGT-Ω), cached

Training curves. Figure 6 reports the SFT and distillation losses for both backbones. Both objectives decrease smoothly, and the 4B and 8B curves nearly overlap. The training-set cosine-distance loss falls from approximately 1 to 0.04, showing that GCTE fits the cached teacher targets during optimization.

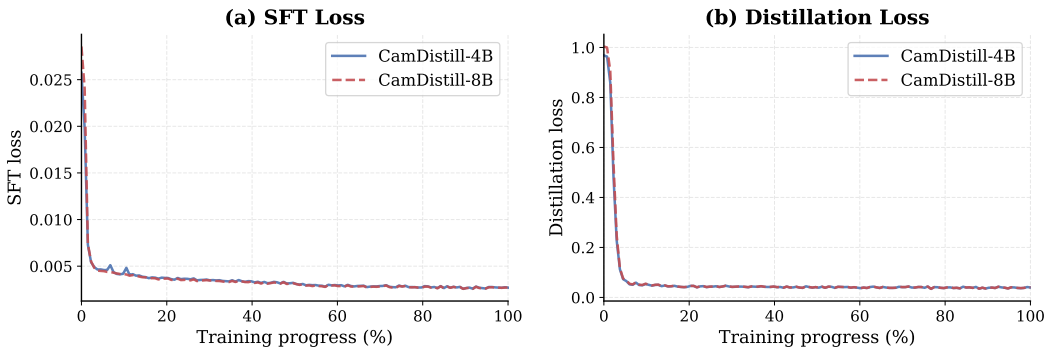


Figure 6. Training loss curves. **(a)** Next-token SFT loss and **(b)** camera-token distillation loss for CAMDISTILL-4B (solid) and CAMDISTILL-8B (dashed) over normalized training progress. Both backbones converge to similar final values (SFT $\approx$ 0.003, distillation $\approx$ 0.04), indicating stable optimization across model scales. Curves are logged directly without additional smoothing.

H CAMERA POSE ESTIMATION METHODS AND TEACHER CHOICE

Our camera tokens are distilled from a 3D foundation model that estimates camera geometry. We briefly review candidate estimators and explain our teacher choice.

Classical structure-from-motion and SLAM systems, such as COLMAP (Schonberger & Frahm, 2016) and DROID-SLAM (Teed & Deng, 2021), recover trajectories through feature matching and geometric optimization. These pipelines can be slow and brittle under low texture or pure rotation. They are also sensitive to dynamic subjects, which are common in film, television, and vlog footage: when a moving person occupies much of the frame, feature matching may attribute subject motion to the camera.

Feed-forward geometry transformers instead learn scene-level priors. DUSt3R (Wang et al., 2024), VGGT (Wang et al., 2025b), and VGGT-Ω (Wang et al., 2026a) jointly estimate camera parameters, depth, and point maps in a single forward pass without bundle adjustment. Permutation-equivariant models such as π^3 (Wang et al., 2026c) improve multi-view consistency and long-sequence stability. ViPE (Huang et al., 2025) jointly estimates depth and pose in low-texture and high-motion scenes, Reloc3r (Dong et al., 2025) focuses on relative-pose regression, and DROID-SLAM in the Wild (Li et al., 2026) improves SLAM robustness in dynamic environments. Large annotated resources such as SpatialVID (Wang et al., 2025a) further support progress in video geometry.

We select **VGGT / VGGT-Ω** as teachers for three reasons. First, their feed-forward inference avoids per-scene optimization. Second, their joint reasoning over cameras and 3D structure is better suited to shots dominated by dynamic subjects than matching-based SfM. Third, they produce one compact 2048-dimensional camera token per frame, providing a direct geometry-aligned target for GCTE.

I GEOMETRY-ONLY BASELINE DETAILS

The geometry-only baseline (§5) converts VGGT-Ω camera poses into CAMCHOREO-style segment labels using the same deterministic pipeline for all benchmark clips. From consecutive poses it computes camera-local translation $(\Delta x, \Delta y, \Delta z)$, local yaw/pitch/roll, world speed, and trajectory curvature. These eight signals are z-normalized for segmentation.

Segmentation. Frames below both a translation-speed threshold of 0.005 and an angular-speed threshold of 0.3° are grouped into static intervals of at least three frames. Their boundaries are fixed, and the remaining intervals are segmented with PELT change-point detection using an RBF cost, penalty 3.0, and minimum segment length three. Segments shorter than three frames are merged into the longer neighbor, and adjacent segments with identical label sets are merged.

Threshold classification. Within each segment, local translation and angular velocities are averaged and each axis is thresholded independently. The evaluated pipeline uses a translation threshold τ_t=0.02 in the pose encoder's local units and a rotation threshold τ_r=0.5°:

Condition	Label
$\Delta z > \tau_t$ / $< -\tau_t$	Dolly In / Dolly Out
$\Delta x > \tau_t$ / $< -\tau_t$	Truck Right / Truck Left
$\Delta y > \tau_t$ / $< -\tau_t$	Pedestal Down / Pedestal Up
yaw $> \tau_r$ / $< -\tau_r$	Pan Left / Pan Right
pitch $> \tau_r$ / $< -\tau_r$	Tilt Down / Tilt Up
roll $> \tau_r$ / $< -\tau_r$	Roll CCW / Roll CW
no axis exceeds its threshold	Static

Multiple axes can exceed their thresholds simultaneously, producing compound camera motion. Arc is approximated when lateral translation and yaw have absolute Pearson correlation at least 0.7 with sufficient amplitude; Follow requires sustained forward motion and moderate trajectory curvature. Unstable is detected at the video level when the high-frequency FFT energy ratio of raw lateral motion exceeds 0.3, using 0.2 of the spectrum as the low-frequency cutoff and at least eight frames. The baseline cannot predict Zoom, which depends on focal-length change rather than extrinsics, or Focus Shift, which is optical rather than geometric. Its hand-designed rules are intended as an interpretable pose-only reference, not a competitive learned model.

J METHOD COMPARISON ON THE QWEN3-VL-8B BACKBONE

Table 8 repeats the comparison from Table 3 with the Qwen3-VL-8B backbone. The pattern is consistent with the 4B results. Pose-as-text prompting (+PromptInject) provides little improvement in frame-level micro F1, SFT produces a large gain, and both camera-token methods improve further. CAMDISTILL remains close to CAMINJECT while avoiding the 3D teacher at inference, retaining the low latency of SFT with only a modest memory increase.

Table 8. Method comparison on the Qwen3-VL-8B backbone. Same protocol as Table 3. Quality metrics are micro/macro type-and-direction F1 and segment localization/detection F1 at IoU 0.5. Latency (s/clip, batch 1) and peak GPU memory measure inference cost on a single H100. `+PromptInject` and CAMINJECT run the VGGT-Ω teacher at inference, whereas `+SFT` and CAMDISTILL do not. Per quality column, best is in **bold** and second-best is underlined; ↓ lower is better.

Method	Micro F1	Macro F1	SegLoc@0.5	SegDet@0.5	Latency (s/clip)↓	Peak Mem (GB)↓
Qwen3-VL-8B	28.3	9.1	67.7	15.0	12.8	27.9
+PromptInject	28.1	12.2	74.5	18.6	21.5	33.7
+SFT	64.1	51.9	79.7	34.9	12.8	27.9
+CamDistill	67.8	57.9	80.4	38.8	12.8	31.6
+CamInject	**68.3**	**59.2**	**80.8**	**39.0**	20.1	33.0

K DETAILED GCTE BLOCK STRUCTURE

GCTE alternates two attention operations within each of its M blocks. A *frame-wise cross-attention* lets each frame's camera token query the frozen visual tokens of that frame, and a *global camera self-attention* then lets the per-frame camera tokens exchange information within the video. Both are standard pre-norm transformer sublayers with LayerNorm (LN), QK-normalized attention, LayerScale-gated residuals, and a feed-forward layer, and both write only to the camera tokens, so the pretrained visual stream is unchanged. This section expands the abstract $\mathrm{CrossAttn}$ and $\mathrm{SelfAttn}$ maps of Section 4 into their sublayer form.

Frame-wise cross-attention. In block m, the camera token $c_i^{(m-1)}$ of frame i queries that frame's frozen features $x_i^{(\ell_m)}$ at the tapped vision layer ℓ_m. With multi-head cross-attention MHCA (query from the camera token, keys and values from the frame features) and LayerScale vectors γ_1, γ_2,

$$u_i^{(m)} = c_i^{(m-1)} + \gamma_1 \odot \mathrm{MHCA}\big(\mathrm{LN}(c_i^{(m-1)}),\ \mathrm{LN}(x_i^{(\ell_m)})\big), \tag{7}$$

$$\tilde{c}_i^{(m)} = u_i^{(m)} + \gamma_2 \odot \mathrm{FFN}\big(\mathrm{LN}(u_i^{(m)})\big). \tag{8}$$

The keys and values are read-only, so the visual tokens $x_i^{(\ell_m)}$ are never updated.

Global camera self-attention. The post-cross-attention tokens of a video then attend to one another through multi-head self-attention MHSA, with LayerScale vectors γ_3, γ_4,

$$v_i^{(m)} = \tilde{c}_i^{(m)} + \gamma_3 \odot \mathrm{MHSA}\big(\mathrm{LN}(\tilde{c}_{1:T}^{(m)})\big)_i, \tag{9}$$

$$c_i^{(m)} = v_i^{(m)} + \gamma_4 \odot \mathrm{FFN}\big(\mathrm{LN}(v_i^{(m)})\big). \tag{10}$$

The attention is masked block-diagonally, so camera tokens from different videos in a batch do not interact. After the final block, the frame-level and temporally contextualized states are concatenated into the token distilled against the teacher,

$$z_i = \big[\tilde{c}_i^{(M)};\ c_i^{(M)}\big] \in \mathbb{R}^{2d_c}. \tag{11}$$

Algorithm 1 summarizes the full forward pass.

Implementation notes. Each camera branch has dimension $d_c = 1024$. Concatenating the final block's post-cross-attention and post-self-attention states gives the 2048-dimensional token z_i, which matches the cached teacher token. The two camera queries are initialized from zero-mean Gaussians ($\sigma = 10^{-3}$) and all linear layers with Xavier initialization. During training the teacher token supervises z_i through the distillation loss, and at inference the teacher branch is removed so that only the projected z_i enters the LLM.

Algorithm 1 Forward pass of GCTE.

1: Initialize c_1 with the first-frame camera query and $c_{2:T}$ with the shared non-first-frame camera query.
2: **for** $m = 1$ to M **do**
3: Select frozen vision feature layer ℓ_m.
4: **for** each frame i **do**
5: **frame-wise cross-attention:** update c_i by cross-attention with $Q = c_i$ and $K, V = x_i^{(\ell_m)}$; keep $x_i^{(\ell_m)}$ unchanged.
6: **end for**
7: Save the post-frame-wise cross-attention tokens in the last block as the frame-level branch.
8: **global camera self-attention:** for each video independently, update $c_{1:T}$ by self-attention over camera tokens only.
9: **end for**
10: Concatenate the final post-frame-wise cross-attention and post-global camera self-attention tokens to obtain $z_i = [c_i^{\text{frame}}; c_i^{\text{global}}]$.
11: Project z_i to the LLM hidden size and insert it before frame i's visual tokens.

Position encoding. The decoder uses multimodal RoPE (M-RoPE), which assigns each token a (t, h, w) coordinate. A camera token receives the temporal index of its frame and the spatial center of that frame's visual patch grid. The decoder therefore interprets it as part of the corresponding frame rather than as an additional time step. Because its coordinate is fixed, prepending or appending the token changes sequence order but not its M-RoPE position.

L Computational Complexity

We analyze only the inference overhead introduced by CamDistill, because its 3D teacher is absent at test time. Let T be the number of frames, P the number of frozen visual tokens per frame, d_v the vision width, d_c the camera-token width, d_l the LLM width, and M the number of GCTE blocks.

GCTE blocks. In each block, frame-wise cross-attention projects the TP frozen visual tokens to keys and values and lets one camera query per frame attend to its P visual tokens. Including projections, attention interactions, and the camera-token feed-forward update, its cost is

$$O\big(TPd_v d_c + TPd_c + Td_c^2\big)\,. \tag{12}$$

Global camera self-attention operates on only the T camera tokens. Its projections, pairwise attention, and feed-forward update cost

$$O\big(Td_c^2 + T^2 d_c\big)\,. \tag{13}$$

Across M blocks, GCTE therefore adds

$$O\big(M\left(TPd_v d_c + TPd_c + Td_c^2 + T^2 d_c\right)\big)\,. \tag{14}$$

For fixed hidden widths, this overhead is linear in the number of visual tokens TP, apart from self-attention over the much shorter sequence of T camera tokens. The expression includes the visual key/value projections omitted by an attention-interaction-only analysis.

LLM sequence overhead. CamDistill inserts exactly one camera token per frame. The LLM sequence length grows from $TP + L$ to $TP + T + L$, where L is the text length. Relative to the visual sequence, the token-count increase is $T/(TP) = 1/P$. This does not make decoder attention free, but it is much smaller than adding dense geometry tokens for every visual patch. The measured system-level effect is reported in Table 3: latency changes from 10.1 to 10.2 seconds per clip on the 4B backbone, while peak memory increases from 18.3 to 20.1 GB. We therefore describe CamDistill as having negligible measured latency overhead, with a modest memory increase, rather than as cost-free.

Summary. CamDistill is efficient for two reasons. First, GCTE replaces the teacher's global attention over all visual tokens with single-query cross-attention and self-attention over only T camera tokens. Second, it increases the LLM sequence length by only a fraction $1/P$. Once the teacher

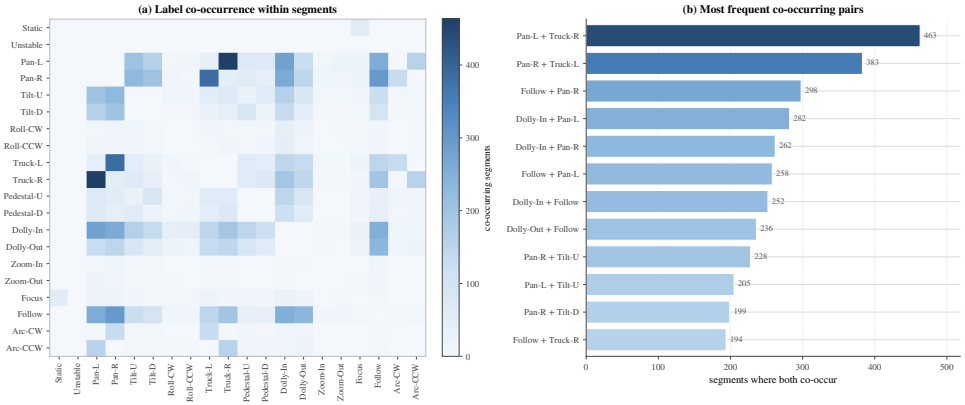


Figure 7. Camera-motion co-occurrence in CAMCHOREO. Left: direction-aware co-occurrence matrix. Right: most frequent label pairs.

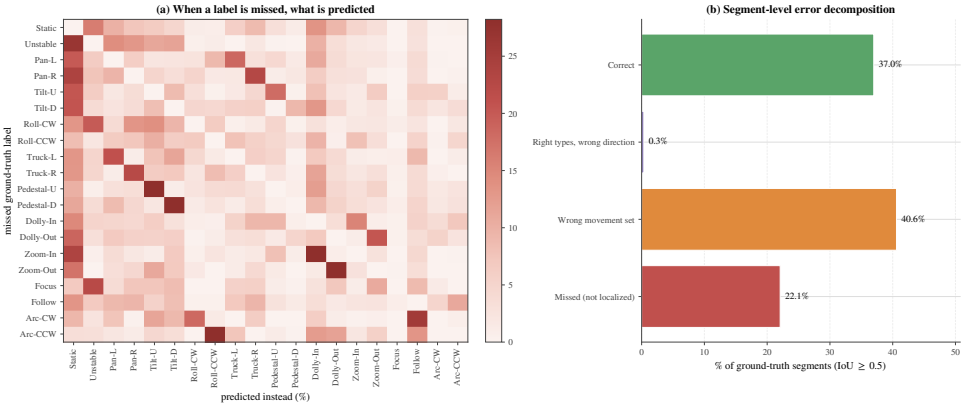


Figure 8. Error analysis of CAMDISTILL-8B. Left: row-normalized frame-level confusions. Right: segment-level error decomposition after IoU ≥ 0.5 matching.

is removed, the resulting inference cost is close to that of the SFT backbone, consistent with the latency and memory measurements in Table 3.

M COMPOSITIONALITY AND ERROR ANALYSIS

Co-occurrence. Camera motion in CAMCHOREO is strongly compositional: 44.2% of segments contain compound camera motion with at least two simultaneous movement primitives. Figure 7 reveals recurring structures rather than arbitrary combinations. Pan and Truck often occur in opposite directions during reveals or orbiting shots, Follow commonly accompanies Pan or Dolly, and Pan+Tilt represents coordinated rotation. The unusual opposing Dolly–Zoom combination appears in eight segments.

Error decomposition. At temporal IoU ≥ 0.5 (Figure 8), CAMDISTILL-8B recovers 37.0% of ground-truth segments with the exact label set. A further 40.6% are localized correctly but contain an incomplete or incorrect set of movements, while 22.1% are missed. Only 0.3% have the correct movement type but the wrong direction. The remaining confusions are physically plausible: distinguishing Dolly from Zoom and Pan from Truck requires parallax or lens evidence that is not apparent from image-plane motion alone.

N PER-CLASS DIFFICULTY, COMPLEXITY, AND TEMPORAL PRECISION

We next use CAMDISTILL-8B predictions to identify where the difficulty of CAMCHOREO is concentrated.

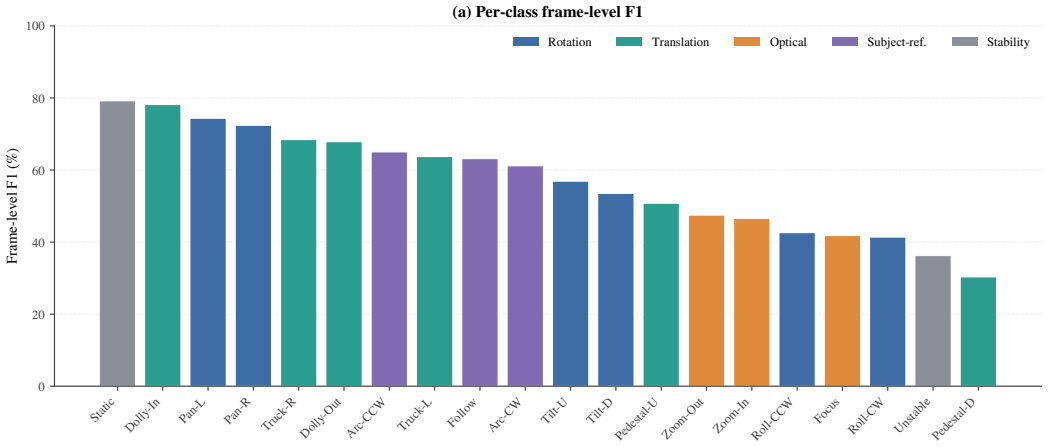


Figure 9. Per-class frame-level F1 of CAMDISTILL-8B, sorted by score and colored by motion family.

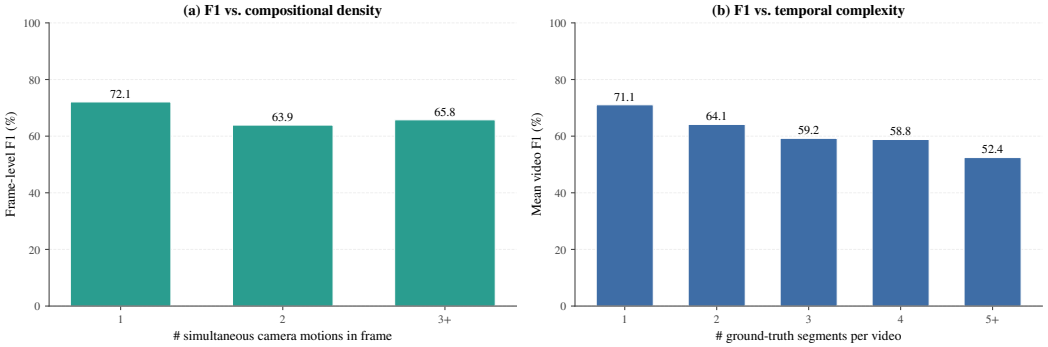


Figure 10. Performance versus task complexity. (a) Frame-level F1 decreases as more camera motions co-occur; (b) mean per-video F1 decreases as the clip is split into more temporal segments.

Per-class difficulty and the long tail. Figure 9 reports frame-level F1 for all 20 direction-aware labels. Performance is uneven and broadly follows class frequency: the more frequent half of the labels average 67.5 F1, compared with 46.6 for the rarer half. Common and geometrically salient movements such as Static, Dolly, Pan, and Truck are recognized reliably. Rare optical and rotational movements such as Zoom, Roll, and Focus Shift remain difficult because they have fewer training examples and depend on subtle cues rather than large image-plane displacement. Arc, although equally rare, is recognized more reliably, since its curved orbiting trajectory produces distinctive image-plane motion. This class imbalance motivates reporting macro averages alongside micro averages.

Difficulty grows with composition and temporal structure. Figure 10 isolates the two properties central to our task. In panel (a), frame-level F1 is 72.1 with one active movement, compared with 63.9 for two and 65.8 for three or more, showing that compound frames are harder than single-motion frames without implying a monotonic trend within the compound groups. In panel (b), mean per-video F1 decreases monotonically from 71.1 for clips with one segment to 52.4 for clips with five or more. These trends confirm that both temporal structure and motion composition contribute substantially to the difficulty of CAMCHOREO.

Temporal precision and segmentation behavior. For predicted segments matched at temporal IoU ≥ 0.5, the boundaries are precise (Figure 11). The median absolute start and end offsets are 0.00 s and 0.04 s, and approximately 91% of matched boundaries fall within 0.5 s of the annotation. At the clip level (Table 9), CAMDISTILL-8B predicts an average of 1.92 segments, compared with 2.03 in the ground truth. It predicts the exact number of segments for 65.7% of clips, under-segments 21.9%, and over-segments 12.4%. The model is therefore slightly conservative: it is more likely to merge adjacent phases than to introduce spurious boundaries. This behavior is also visible in the qualitative examples in Section O.

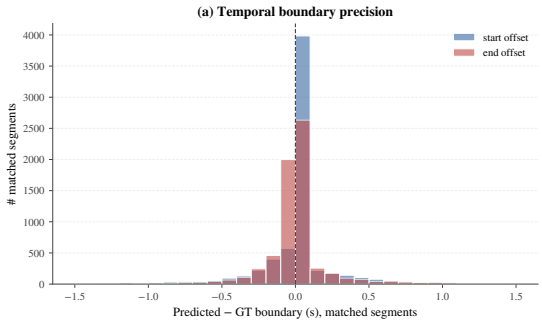


Figure 11. Boundary precision. Signed offset between matched predicted and ground-truth segment boundaries.

Table 9. Segmentation and boundary statistics of CAMDISTILL-8B on CAMCHOREO.

Statistic	Value
Mean GT segments / video	2.03
Mean pred. segments / video	1.92
Exact segment count	65.7%
under- / over-seg.	21.9% / 12.4%
Median $\|\Delta\|$ start / end	0.00 / 0.04 s
Within 0.5 s (start / end)	91.0 / 91.3%

O QUALITATIVE RESULTS

O.1 REPRESENTATIVE PREDICTIONS

Figures 12–17 show example predictions of CAMDISTILL-8B on CAMCHOREO, drawn from clips that span a range of temporal and compositional complexity. In each panel, sampled frames appear above the ground-truth (GT) and predicted (Pred) segment timelines. Segments matched by temporal IoU share a color, unmatched segments are gray, and every segment is labeled with its movements. Across these examples, the model recovers the dominant movements and the overall temporal structure, including compositional cases such as a simultaneous Roll and Pedestal Up or a Truck+Pan+Arc orbit. Its remaining errors are mostly merged adjacent phases or a movement dropped from a densely compositional segment, consistent with the error analysis in Appendix M.

O.2 FAILURE CASES

CAMDISTILL still breaks down on the hardest clips, and Figure 18 shows four representative errors. They fall into a few recurring modes. First, the model confuses movements that look similar in the image plane but differ geometrically, reading a lateral Truck as a Pan and a forward Dolly as a Zoom; both distinctions require the parallax cues that its encoder captures poorly. Second, it misses subject-referenced motion, labeling a hand-held Follow shot as Static or a plain Pan. Third, it mistakes irregular Unstable footage for Roll and over-segments it into several short intervals. Across all four, dense compositions are only partially recovered and adjacent phases are often merged. These modes match the quantitative error analysis in Appendix M, where most residual errors are incomplete or incorrect label sets rather than boundary errors.

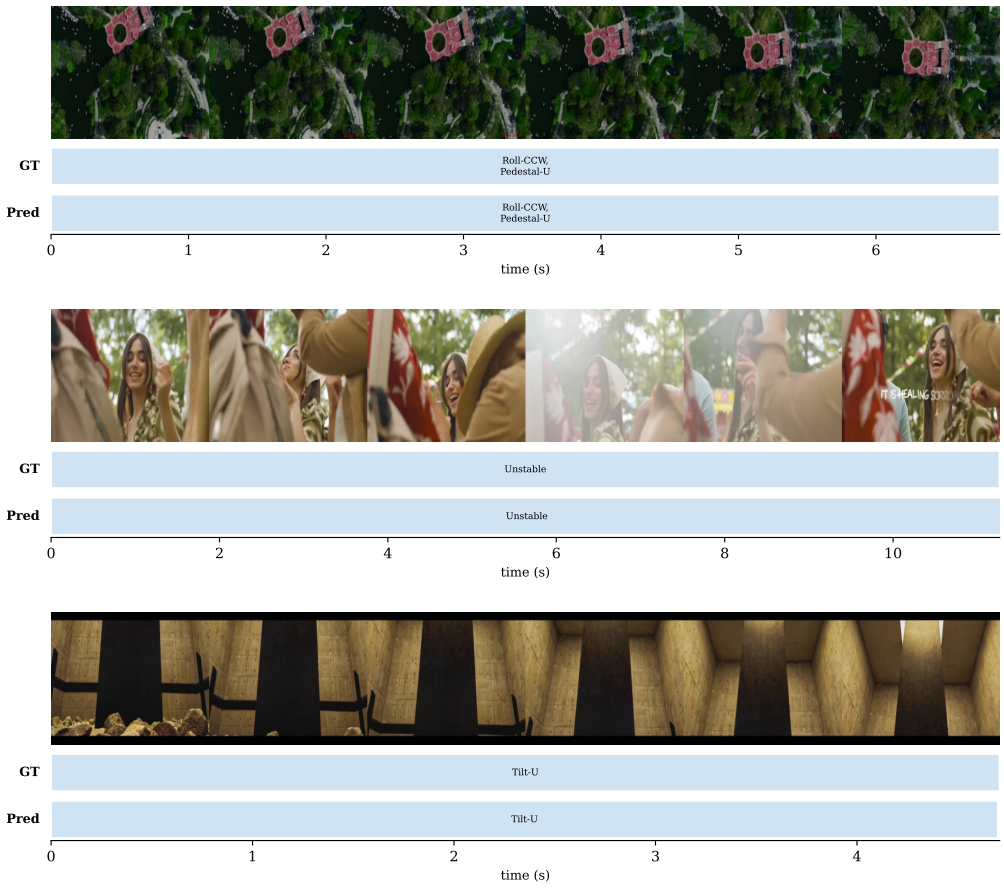


Figure 12. Qualitative predictions of CAMDISTILL-8B on CAMCHOREO (part 1). Matched GT and Pred segments (by temporal IoU) share a color, unmatched segments are gray, and each segment lists its movements.



Figure 13. Qualitative predictions on CAMCHOREO (part 2). Conventions as in Figure 12.

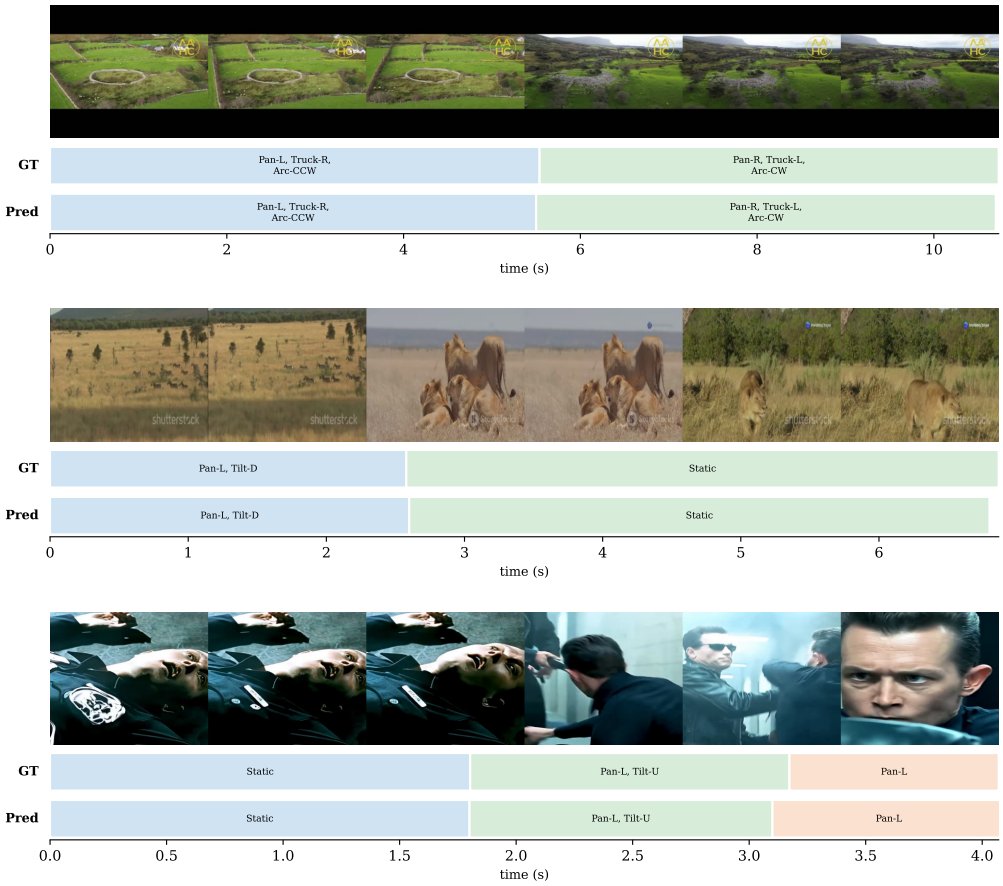


Figure 14. Qualitative predictions on CAMCHOREO (part 3). Conventions as in Figure 12.



Figure 15. Qualitative predictions on CAMCHOREO (part 4). Conventions as in Figure 12.

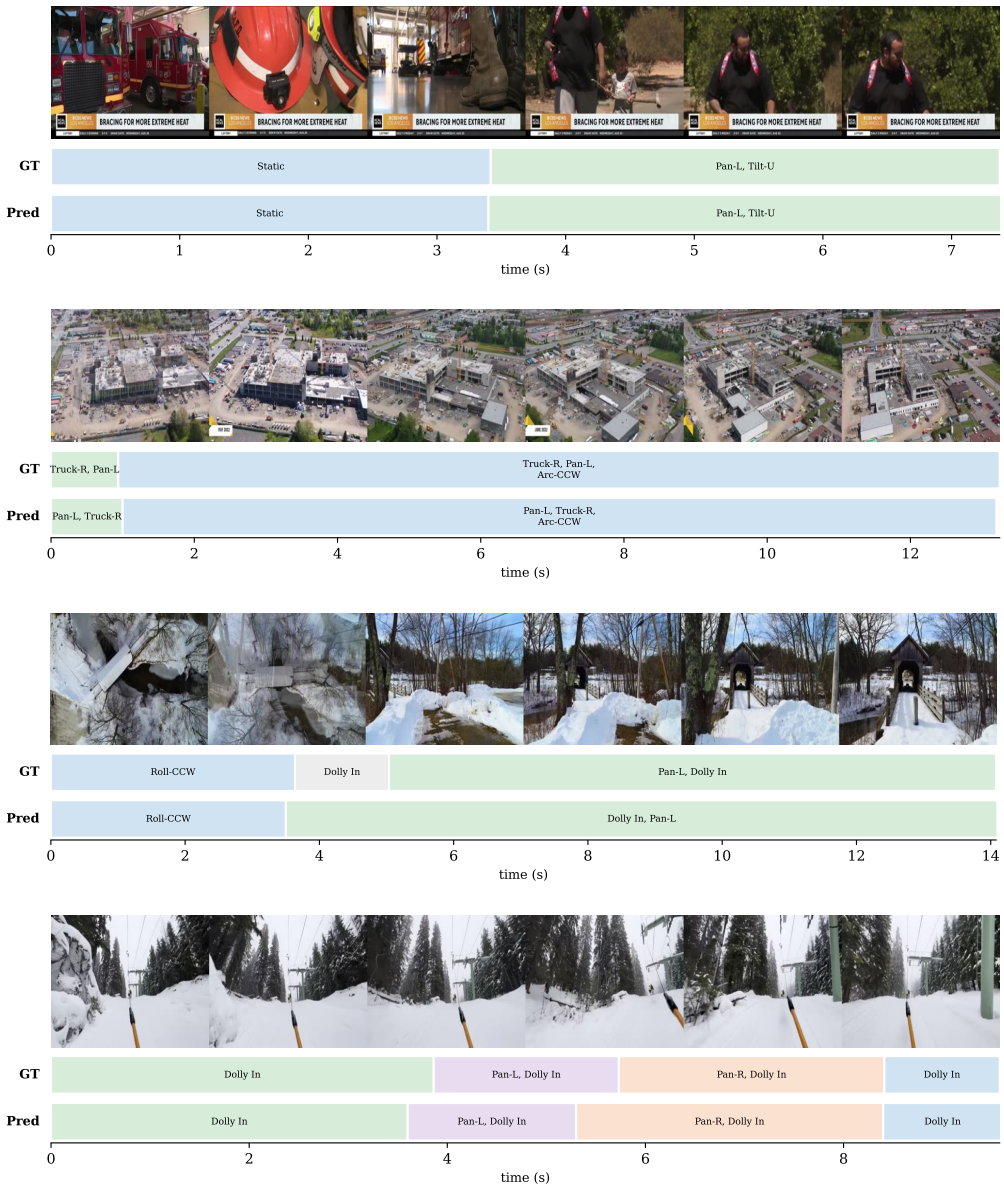


Figure 16. Qualitative predictions on CAMCHOREO (part 5). Conventions as in Figure 12.

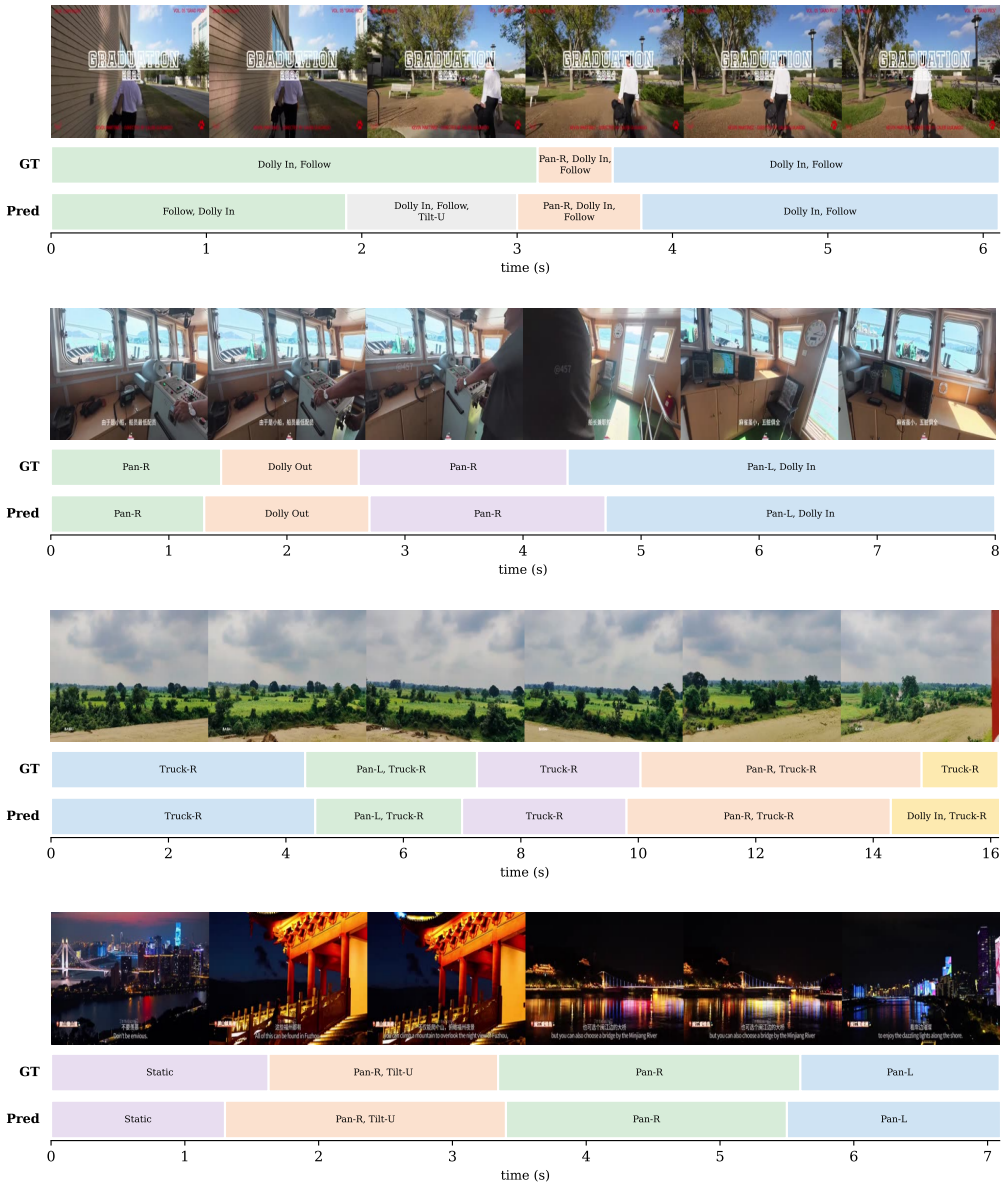


Figure 17. Qualitative predictions on CAMCHOREO (part 6). Conventions as in Figure 12.

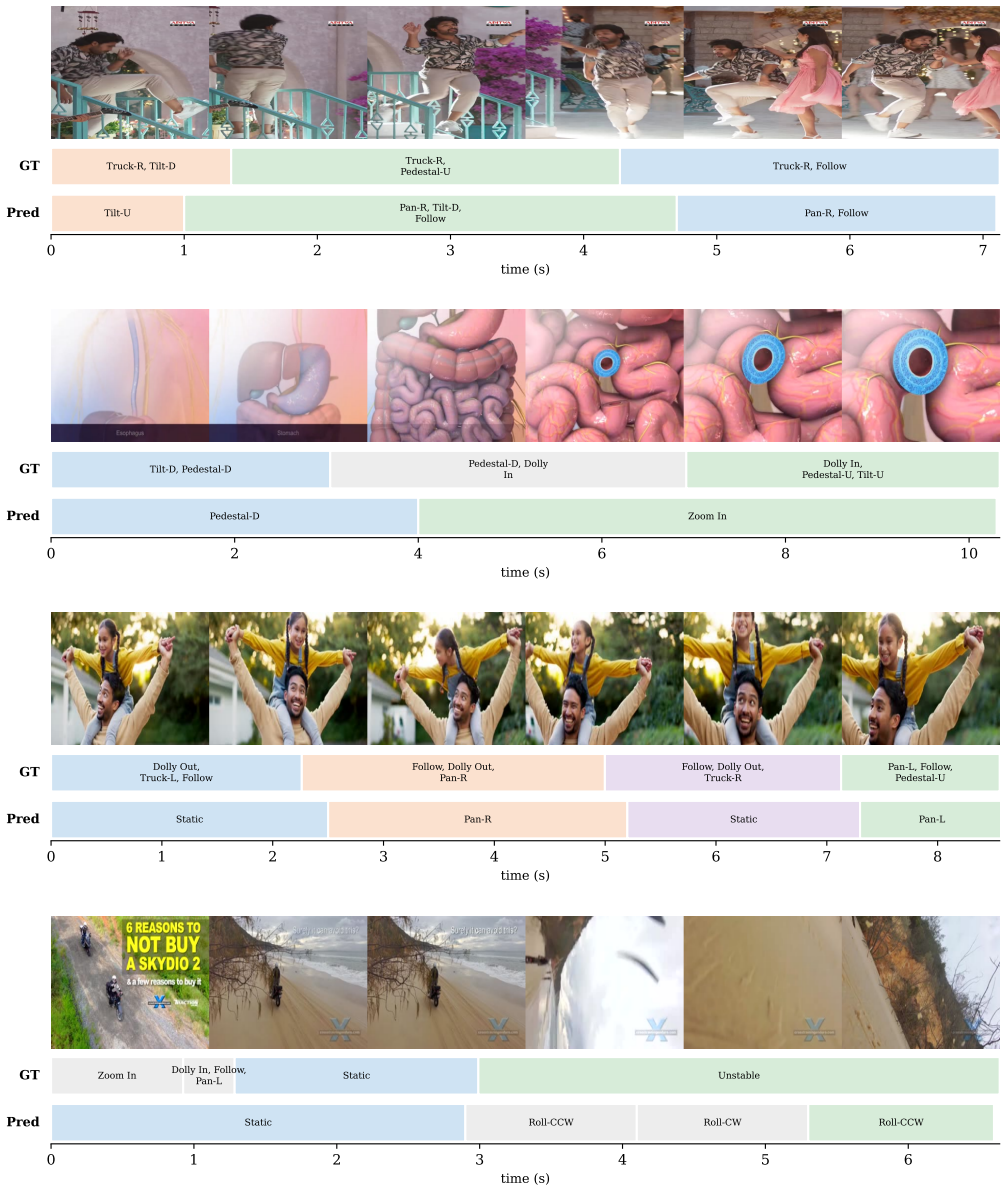


Figure 18. Failure cases of CAMDISTILL-8B. Top to bottom: a lateral Truck predicted as Pan; a forward Dolly predicted as Zoom; a hand-held Follow predicted as Static or Pan; and Unstable footage predicted as Roll and over-segmented into short intervals. Conventions as in Figure 12.