

Geometry-Guided Camera Motion Understanding in VideoLLMs

Haoan Feng¹, Sri Harsha Musunuri², Guan-Ming Su²,
¹University of Maryland, College Park, ²Dolby Laboratories Inc.

hfengac@umd.edu, harsha.musu@gmail.com, guanmingsu@ieee.org

Contents

A Camera Motion Taxonomy and Constraints	1
A.1 Atomic Motion Primitives	1
A.2 Incompatibility constraints and label-set canonicalization	2
B Dataset Construction Details	2
B.1 Pose-to-label mapping	3
B.2 Re-balancing and class distribution	4
C CameraMotionVQA Protocol	4
C.1 Prompt templates and answer normalization	4
C.2 Distractor sampling algorithm	4
D Additional Experimental Results	5
D.1 Classifier capacity ablation	5
D.2 Per-label breakdown and confusion analysis	6
E Full Prompts and additional Qualitative Examples	7
E.1 Full prompt templates	7
E.2 More qualitative examples	8
F. Additional Discussion	11
F.1. Synthetic-to-real gap	11
F.2. Camera intrinsics and zoom	12
F.3. Joint training vs. plug-and-play	12
F.4. VGGT dependency and practical deployment	12
G Proposed Data Augmentation Strategy	12
H LLM-as-Judge Evaluation Protocol	13
H.1 Preliminary Results	13

A. Camera Motion Taxonomy and Constraints

We adopt the camera motion taxonomy proposed in CameraBench [3] and refer readers to that work for a comprehensive definition of motion primitives, annotation guidelines, and constraint design. CameraBench formalizes camera motion as a set of atomic cinematographic operations

(*e.g.*, pan, tilt, dolly, roll, arc) with clear geometric interpretation and explicit rules to avoid contradictory labels.

In this work, we use a controlled subset of 15 primitives tailored to short (1-second) within-shot segments, focusing on the extrinsic parameter changes of the camera. Specifically, we focus on principal rotation and translation operations that can be reliably identified at this temporal granularity, while preserving the incompatibility structure defined in CameraBench (*e.g.*, opposing directions along the same degree of freedom are mutually exclusive, and `static` cannot co-occur with any non-static motion).

Our taxonomy is therefore not a redefinition, but a task-specific instantiation of the CameraBench framework, designed to support constrained multi-label benchmarking and systematic diagnosis of camera motion recognition in VideoLLMs.

A.1. Atomic Motion Primitives

Our benchmark instantiates a compact, implementation-friendly subset of $K=15$ atomic motion primitives, summarized in Tab. 1. Each primitive corresponds to a distinct geometric degree of freedom of the camera motion: three rotations (yaw, pitch, roll), three translations (X, Y, Z), a coupled rotation–translation pattern (`arc`), and a `static` state. This axis-aligned parameterization keeps the label space interpretable, physically grounded, and directly compatible with 3D camera representations.

The primitives are designed to be identifiable at the 1-second, within-shot granularity used in our benchmark, and they follow standard cinematographic terminology. Across different axes, primitives may co-occur (*e.g.*, `pan left + tilt up`, or `dolly in + pan right`), motivating a multi-label rather than single-label formulation. We also include `arc` to capture common cinematographic motion where yaw rotation is coupled with lateral translation, which is difficult to describe as a single pure-axis primitive. Finally, `static` denotes the absence of dominant camera motion above predefined thresholds and serves as a diagnostic case for distinguishing true camera motion from object-induced or scene-induced apparent motion in VideoLLMs.

Table 1. **Atomic camera motion primitives.** We list the $K=15$ primitives used in our benchmark, grouped by geometric degree of freedom (rotations, translations, coupled motion, and static). Opposing directions within each axis are mutually exclusive, forming the basis of our incompatibility constraints, while primitives across different axes may co-occur under a constrained multi-label formulation.

Axis	Primitive	Operational Definition	Mutual exclusion
Yaw	pan left / pan right	Horizontal rotation about vertical axis	pan left \leftrightarrow pan right
Pitch	tilt up / tilt down	Vertical rotation about horizontal axis	tilt up \leftrightarrow tilt down
Roll	roll cw / roll ccw	In-plane rotation about optical axis	roll cw \leftrightarrow roll ccw
Y	truck left / truck right	Lateral translation parallel to image plane	truck left \leftrightarrow truck right
Z	crane up / crane down	Vertical translation of camera center	crane up \leftrightarrow crane down
X	dolly in / dolly out	Translation along view axis	dolly in \leftrightarrow dolly out
Coupled	arc cw / arc ccw	Coupled yaw + lateral translation	arc cw \leftrightarrow arc ccw
None	static	No dominant motion above thresholds	static \leftrightarrow any non-static

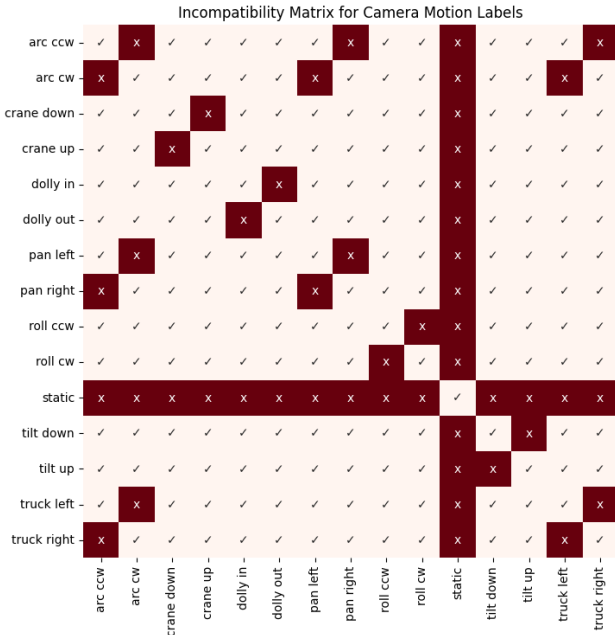


Figure 1. **Incompatibility matrix for constrained multi-label camera motion.** Entry $M_{ij}=1$ indicates that primitives i and j are mutually exclusive and cannot co-occur in the same 1-second segment. The axis structure produces block-wise exclusivity (*e.g.*, left/right pairs), and *static* is incompatible with all non-static primitives. Row/column ordering matches the primitive index order used in our dataset and experiments.

A.2. Incompatibility constraints and label-set canonicalization

Camera motion primitives are not independent. We enforce geometric exclusivity constraints using a symmetric incompatibility matrix $M \in \{0, 1\}^{K \times K}$, where $M_{ij}=1$ indicates that primitives i and j cannot co-occur in the same 1-second segment. Fig. 1 visualizes M ; the row/column ordering matches the primitive index ordering used throughout the dataset, training, and evaluation. Within each de-

gree of freedom, opposing directions are mutually exclusive (*e.g.*, pan left vs. pan right), and *static* is mutually exclusive with any non-static primitive. These constraints prevent contradictory supervision and yield a well-defined target space for constrained multi-label recognition (and the corresponding VQA-style evaluation).

To make labels deterministic and comparable across data generation, training logs, and evaluation scripts, we canonicalize each segment’s label set as follows: (i) remove duplicates; (ii) drop any label set that violates M (*i.e.*, contains a conflicting pair); (iii) sort remaining labels by a fixed global order (the primitive index order used in Fig. 1); and (iv) enforce the cardinality of at least one and at most three primitives per segment. The last step reflects our benchmark design assumption that 1-second clips rarely contain more than three dominant, distinguishable camera motions, and it ensures a bounded and interpretable output space for both classifiers and VideoLLM prompting.

B. Dataset Construction Details

We construct **CameraMotionDataset** and **CameraMotionVQA** to provide controlled, geometry-aware supervision and standardized evaluation for camera motion understanding in VideoLLMs. As summarized in Tab. 2, existing benchmarks primarily focus on broad cinematographic techniques, perceptual composition attributes, or real-world camera motion without explicit geometric supervision. In contrast, our dataset is built from synthetic videos with known camera parameters, segmented into fixed 1-second within-shot clips, and annotated with constrained multi-label motion primitives. As shown in the main paper, this design enables (i) precise primitive-level supervision, (ii) constraint-consistent evaluation via a multiple-choice VQA protocol, and (iii) controlled studies of geometry-derived camera cues and motion-token distillation. Below, we detail the data source, segmentation strategy, label generation process, and balancing procedure.

Table 2. Positioning of CameraMotionDataset and CameraMotionVQA against related benchmarks and cinematic datasets. We contrast task focus, temporal unit, label structure (including whether explicit constraints or QA protocols are provided), camera-motion granularity, intended use, and whether explicit camera parameters (geometry access) are available.

Work	Primary focus	Temporal unit	Label type (constraints / QA)	Motion granularity	Intended use	Geometry access
CameraBench [3]	Camera motion primitives in diverse real-world videos; taxonomy + label-then-caption annotations.	Short video clips (avg. ~5–6s); shot-aligned / manually segmented.	Multi-label binary primitive tags (~50); plus captions and paired yes/no VQA skill probes.	Primitive-level; directional primitives (translation / rotation / zoom / tracking, etc.).	Mixed: benchmark suite (classification, retrieval, captioning, VQA) + small-scale training for motion-aware VLMs.	No: real videos (no ground-truth camera parameters).
CineTechBench [7]	Broad cinematographic techniques (scale, angle, composition, movement, lighting, color, focal length) with expert annotation.	Movie images + short movie clips/shots (~600 images, 120 clips).	Manual technique labels; question-answer pairs and annotated descriptions for understanding; separate camera-movement generation evaluation.	Coarse-to-mid: technique categories; camera movement is one dimension among 7.	Diagnostic benchmark (understanding + generation evaluation).	No: no explicit camera parameters provided.
VidComposition [6]	Composition understanding in compiled videos (cinematography, character, narrative, scene, making).	Compiled videos (avg. ~20 min) segmented into coherent sections.	Human-annotated multiple-choice QA (982 videos / 1706 questions) across 15 sub-tasks, including camera movement / angle / shot size perception.	Coarse-to-mid: composition attributes; camera motion appears as a perceptual attribute in QA.	Diagnostic benchmark for MLLMs on compiled-video composition.	No: no explicit camera parameters provided.
CineScale2 [4]	Cinematic camera features (camera angle + camera level) on movie frames/images.	Frames / images (~24.7k).	Single-label categorical annotations per attribute (angle: 5 classes; level: 6 classes).	N/A for motion (static camera setup attributes only).	Training-scale dataset for recognition of camera angle/level; film-style analysis.	No: no explicit camera parameters provided.
CameraMotionDataset	Camera motion primitive dataset for controlled supervision and motion-token distillation.	Within-shot 1s segments (fixed-length; shot-consistent).	Constrained multi-label primitive annotations with axis-wise incompatibility constraints.	Primitive-level; directional; axis-wise constraints; supports compound motions within 1s.	Supervision dataset (training / distillation / controlled analysis).	Yes: synthetic camera parameters available (controlled rendering).
CameraMotionVQA	Camera-motion sensitivity benchmark for VideoLLM evaluation via structured prompting.	Same 1s within-shot segments mapped to QA format.	Standardized multiple-choice VQA protocol derived from ground-truth primitives; constraint-consistent answer space.	Primitive-level reasoning over directional motions (single and compound).	Diagnostic benchmark for motion-aware understanding and prompting.	Implicit (inherits synthetic geometry from CameraMotionDataset; not directly exposed to models).

B.1. Pose-to-label mapping

Each 1-second segment contains $T=15$ camera extrinsics $\{\mathbf{E}_t\}_{t=1}^T$, where $\mathbf{E}_t = [\mathbf{R}_t \mid \mathbf{t}_t] \in \mathbb{R}^{3 \times 4}$. Translations are expressed in world units (meters in the renderer), and rotations are represented as 3×3 matrices. We assume a right-handed coordinate system where the camera forward axis corresponds to the negative x -axis, and yaw rotation occurs about the world vertical axis.

The pose sequence is mapped to primitive labels using two signals: (i) net translation expressed in the initial camera frame and (ii) accumulated inter-frame rotation. All angles are computed in degrees, and Algorithm 1 summarizes the full procedure.

Rotation statistics. For consecutive frames, we compute the relative rotation $\mathbf{R}_{\Delta,t} = \mathbf{R}_t \mathbf{R}_{t-1}^\top$ and convert it to an axis-angle representation. We accumulate signed changes for yaw (pan), pitch (tilt, estimated from the forward-vector change), and roll. Let $\Delta_{\text{pan}}, \Delta_{\text{tilt}}, \Delta_{\text{roll}}$ denote the total signed changes over the segment, and $\Sigma_{\text{pan}}, \Sigma_{\text{tilt}}, \Sigma_{\text{roll}}$ de-

note the accumulated absolute changes.

Static / rotation-dominant case. Let d_{trans} be the total translation distance (sum of inter-frame translations). If $d_{\text{trans}} < 0.05$, we treat the segment as rotation-dominant. We then assign `static` if $\Sigma_{\text{pan}} < 0.2$ and $\Sigma_{\text{tilt}} < 0.2$; otherwise we assign `pan left/right` based on the sign of Δ_{pan} when $\Sigma_{\text{pan}} > 0.2$, and `tilt up/down` based on the sign of Δ_{tilt} when $\Sigma_{\text{tilt}} > 0.2$.

Translation-dominant case. If translation is significant, we compute the net translation in the initial camera frame:

$$\Delta \mathbf{t}_{\text{cam}} = \mathbf{R}_1^\top (\mathbf{t}_T - \mathbf{t}_1),$$

with components (x, y, z) corresponding to forward, lateral, and vertical motion, respectively. We use an adaptive threshold $t_{\text{move}} = \max(0.3 \cdot \|\Delta \mathbf{t}_{\text{cam}}\|_\infty, 0.5)$ to decide whether a component is dominant.

To distinguish `arc` from near-straight translation, we estimate path curvature

$$\kappa = \frac{\sum_t \|\mathbf{f}_t - \mathbf{f}_{t-1}\|_2}{\sum_t \|\mathbf{t}_t - \mathbf{t}_{t-1}\|_2 + \epsilon},$$

Algorithm 1 Pose-to-label mapping

Require: Extrinsics $\{(\mathbf{R}_t, \mathbf{t}_t)\}_{t=1}^T$

- 1: $\mathcal{Y} \leftarrow \emptyset$
- 2: Compute d_{trans} (total translation distance) and $(\Delta_{\text{pan}}, \Delta_{\text{tilt}})$, $(\Sigma_{\text{pan}}, \Sigma_{\text{tilt}})$ (signed / abs rotation stats)
- 3: **if** $d_{\text{trans}} < 0.05$ **then** \triangleright rotation-dominant / static
- 4: **if** $\Sigma_{\text{pan}} < 0.2$ **and** $\Sigma_{\text{tilt}} < 0.2$ **then**
- 5: **return** {static}
- 6: **end if**
- 7: **if** $\Sigma_{\text{pan}} > 0.2$ **then**
- 8: $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{pan right}\}$ **if** $\Delta_{\text{pan}} > 0$ **else** $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{pan left}\}$
- 9: **end if**
- 10: **if** $\Sigma_{\text{tilt}} > 0.2$ **then**
- 11: $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{tilt down}\}$ **if** $\Delta_{\text{tilt}} > 0$ **else** $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{tilt up}\}$
- 12: **end if**
- 13: **else** \triangleright translation-dominant
- 14: Compute $\Delta \mathbf{t}_{\text{cam}}$ and t_{move}
- 15: Compute curvature κ from forward vectors
- 16: **if** $\kappa > 9 \times 10^{-4}$ **then**
- 17: $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{arc cw}\}$ **if** $\Delta_{\text{pan}} > 0$ **else** $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\text{arc ccw}\}$
- 18: **end if**
- 19: Update \mathcal{Y} with dolly in/out (forward-dominant), truck left/right (lateral-dominant), and crane up/down (vertical-dominant)
- 20: **end if**
- 21: **return** Canonicalize(\mathcal{Y}) (Sec. A.2)

where \mathbf{f}_t is the camera forward vector in world coordinates. If $\kappa > 9 \times 10^{-4}$, we classify the segment as `arc cw/ccw` based on the sign of Δ_{pan} , and optionally add auxiliary `tilt` or `dolly` if their corresponding rotation/translation magnitudes exceed thresholds. Otherwise, we assign axis-aligned translation primitives according to dominant components: `dolly in/out` from x , `truck left/right` from y , and `crane up/down` from z .

Roll. We additionally assign `roll cw/ccw` only when forward motion is present and $\Sigma_{\text{roll}} > 0.5$, which avoids spurious roll labels from minor numerical jitter.

To verify the correctness of the automatic mapping, we randomly sampled 720 segments and collected primitive annotations using the same 15-label set. Annotators were instructed to select the correct option from an MCQ-style VQA benchmark and to prioritize dominant motion over minor jitter. The resulting agreement accuracy is 93%, supporting the reliability of the pose-to-label mapping procedure.

B.2. Re-balancing and class distribution

The full pose-to-label conversion over all rendered videos produces 542,504 1-second segments. As shown in Fig. 2(a), due to the scripted camera trajectories in the synthetic generator, the resulting label distribution is highly imbalanced. In particular, simple directional primitives such as `tilt down` and `arc ccw` occur far more frequently than rarer motions such as `roll cw/ccw`. Without re-balancing,

this imbalance would bias classifiers toward dominant primitives and obscure diagnostic conclusions about motion sensitivity in VideoLLMs.

To mitigate this effect, we apply per-label-set downsampling. For each unique canonical label set (Sec. A.2) (e.g., `tilt up` and `pan left`), we cap the number of segments to at most 200 samples. If a label set contains fewer than 200 segments, all instances are retained. This strategy preserves the diversity of primitive combinations while preventing over-representation of frequent motion patterns.

After re-balancing, the final dataset contains **12,274** segments. Fig. 2 visualizes the label-set frequency distribution before and after downsampling. The post-balanced distribution is substantially flatter, enabling more controlled evaluation of primitive-level recognition and structured prompting.

C. CameraMotionVQA Protocol

C.1. Prompt templates and answer normalization

Each 1-second clip in CameraMotionDataset is converted into a 4-way multiple-choice VQA instance by pairing the video with a fixed prompt template and four candidate label sets (one correct and three distractors). We use the following prompt string (shown verbatim) with lettered options:

```
<video>
Identify the camera motion depicted in
the video using standard cinematographic
terminology.
Options:
(A) <optA>
(B) <optB>
(C) <optC>
(D) <optD>
```

The reference answer is stored as `Answer: <letter>`.

Option text. For readability, we verbalize `cw/ccw` as `clockwise/counterclockwise` in options, while preserving all other primitive names (e.g., `pan left`, `dolly in`). Multi-primitive options are formatted as a comma-separated list in canonical order (Sec. A.2).

Answer parsing. At evaluation time, we normalize model outputs by extracting the first occurrence of a valid choice letter in $\{A,B,C,D\}$ (case-insensitive), optionally preceded by the token `Answer:`. If no valid letter is found, model output will be parsed to try to match the ground truth; otherwise, the prediction is marked as invalid. This normalization ensures a consistent, model-agnostic scoring interface for VideoLLMs.

C.2. Distractor sampling algorithm

As shown in Algorithm 2, a naïve distractor strategy (sampling arbitrary label sets) can make questions either triv-

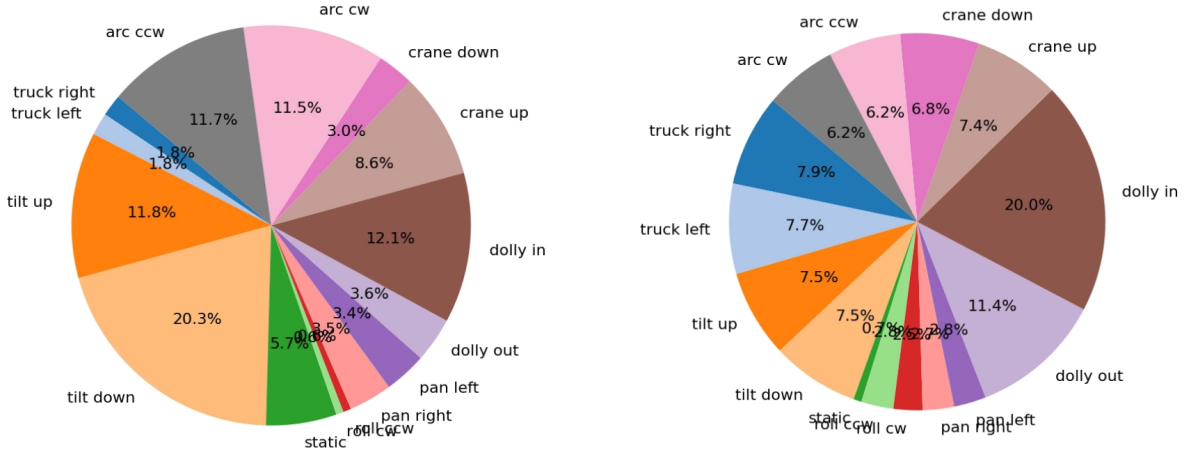


Figure 2. **Label-set distribution before and after re-balancing.** We cap each canonical primitive combination at 200 segments. The original distribution is highly skewed toward common directional motions, while the rebalanced dataset provides a more uniform coverage across primitive combinations.

ial (e.g., mismatched label cardinality) or invalid (e.g., violating incompatibility constraints). We therefore sample distractors from a precomputed pool \mathcal{Y} of *canonicalized, constraint-valid* label sets (Sec. A.2), stratified by label cardinality $c \in \{1, 2, 3\}$.

Complexity-matched distractors. Given a ground-truth label set y^* with complexity $c = |y^*|$, we sample distractors using the following rule, matching the implementation in our dataset generator:

- If $c = 1$: sample 3 distractors from single-label sets and 1 distractor from two-label sets.
- If $c = 2$: sample 2 distractors from two-label sets, 1 from single-label sets, and 1 from three-label sets.
- If $c = 3$: sample 2 distractors from two-label sets and 1 from three-label sets (then insert y^* to ensure one correct option).

This yields a mixture of easy and moderately hard negatives while avoiding degenerate cases where the correct answer is identifiable solely by the number of primitives.

Constraint validity. All candidates (including distractors) are drawn from \mathcal{Y} , which contains only incompatibility-consistent label sets. As a result, every option is valid under the same constraint system as the ground truth, preventing models from exploiting constraint violations as shortcuts.

D. Additional Experimental Results

D.1. Classifier capacity ablation

We study the effect of classifier capacity on camera-motion recognition. The classifier is a lightweight Transformer en-

Algorithm 2 Distractor sampling for CameraMotionVQA (4-way MCQ)

Require: GT label set y^* ; pools $\mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3$ of valid label sets with $|y| = 1, 2, 3$

- 1: $c \leftarrow |y^*|$; $\mathcal{O} \leftarrow \emptyset$
- 2: **if** $c = 1$ **then**
- 3: $\mathcal{O} \leftarrow \text{Sample}(\mathcal{Y}_2, 1) \cup \text{Sample}(\mathcal{Y}_1, 3)$
- 4: **else if** $c = 2$ **then**
- 5: $\mathcal{O} \leftarrow \text{Sample}(\mathcal{Y}_1, 1) \cup \text{Sample}(\mathcal{Y}_3, 1) \cup \text{Sample}(\mathcal{Y}_2, 2)$
- 6: **else**
- 7: $\mathcal{O} \leftarrow \text{Sample}(\mathcal{Y}_2, 2) \cup \text{Sample}(\mathcal{Y}_3, 1)$
- 8: **end if**
- 9: Replace one element in \mathcal{O} with y^* to ensure exactly one correct option
- 10: Verbalize cw/ccw \rightarrow clockwise/counterclockwise; shuffle and assign letters A–D
- 11: **return** Options $\{(A, y_A), (B, y_B), (C, y_C), (D, y_D)\}$ and correct letter

coder operating on frozen VideoLLM vision features. We vary three architectural parameters: the number of encoder blocks (2–6), the number of attention heads (4 or 8), and the hidden dimension (256–896).

Fig. 3 plots accuracy as a function of model size. Across all configurations, performance varies within a relatively narrow range (about 0.71–0.79), indicating that increased classifier capacity does not consistently translate to higher accuracy. The best results are obtained with small models (20–50M parameters), while larger configurations with wider hidden dimensions or additional layers do not provide reliable improvements.

Increasing the number of attention heads yields a modest gain (typically 1–2%), but the overall trend shows early saturation: models larger than roughly 50M parameters do not improve performance and sometimes degrade slightly. This behavior is consistent with a representation bottleneck in

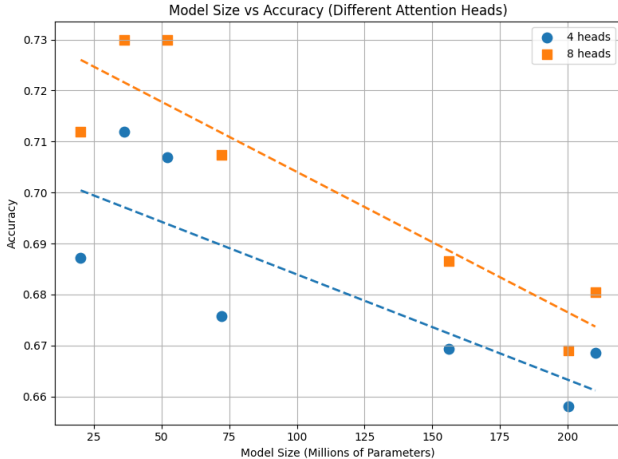


Figure 3. **Classifier capacity ablation.** Accuracy versus classifier size (M parameters) for varying Transformer depth, head count, and hidden dimension. Overall performance saturates with small models; using 8 attention heads yields a modest but consistent gain over 4 heads.

the frozen vision features, but it may also reflect limited supervision after re-balancing (Sec. B.2), where the effective training set size is relatively small for fitting larger classifiers. A potential direction to better exploit higher-capacity heads is to increase data diversity via augmentation, *e.g.*, temporal replay/resampling within the 1-second clip, synthetic camera roll perturbations, and scale-preserving random crops (to simulate focal-length and framing variation) while keeping the motion label unchanged.

D.2. Per-label breakdown and confusion analysis

We provide per-label analysis for the three camera-motion recognition pipelines used throughout the paper: *VGGT classifier* (teacher cues), *VGGT-Q-Former* (distilled student cues), and *Q-Former probing* (diagnostic baseline on frozen VideoLLM vision features). Following Tab. 2 in the main paper, we report class-averaged and frequency-weighted scores: the overall **Macro-F1 / Weighted-F1** are **0.87 / 0.92** (VGGT classifier), **0.83 / 0.87** (VGGT-Q-Former), and **0.69 / 0.74** (Q-Former probing). These aggregate gaps are reflected consistently in the per-primitive results in Tab. 3.

Per-label breakdown. Tab. 3 reports F1 for each primitive. VGGT-derived cues yield uniformly strong performance across most translation and rotation primitives, with several classes exceeding 0.9 F1 (*e.g.*, *dolly in/out*, *crane up/down*, *truck left/right*, *roll cw*). Distilling VGGT cues into a compact Q-Former incurs a moderate but systematic drop (typically 0.03–0.06 F1 per class), while still substantially outperforming probing. This aligns with the intended accuracy–throughput trade-off: the

Table 3. **Per-primitive F1 scores.** VGGT classifier achieves the best accuracy and consistently outperform the other two methods. A similar trend is visible between VGGT-Q-Former and Q-Former probing. However, the VGGT classifier requires running the 3D backbone, leading to higher computational and memory overhead. VGGT-Q-Former reuses frozen VideoLLM vision features and trades a modest accuracy drop for substantially higher throughput. Compared to probing, both VGGT-derived variants improve nearly all primitives, with the largest gains on ambiguous motions (*pan*, *tilt*) and the *static* class.

Primitive	VGGT Classifier	VGGT-Q-Former	Q-Former Probing
arc ccw	0.88	0.83	0.68
arc cw	0.94	0.90	0.75
crane down	0.94	0.89	0.74
crane up	0.93	0.88	0.72
dolly in	0.94	0.90	0.78
dolly out	0.90	0.85	0.75
pan left	0.83	0.76	0.58
pan right	0.85	0.80	0.66
roll ccw	0.91	0.86	0.71
roll cw	0.97	0.92	0.83
static	0.65	0.44	0.27
tilt down	0.94	0.90	0.78
tilt up	0.89	0.83	0.69
truck left	0.93	0.88	0.74
truck right	0.92	0.88	0.77
Macro F1	0.87	0.83	0.69

VGGT classifier achieves the best recognition accuracy but requires running the 3D backbone, whereas VGGT-Q-Former reuses frozen VideoLLM visual features at inference time and is therefore significantly more efficient.

The most challenging class is *static*, which has the lowest F1 across all methods. This is expected because *static* is defined by the *absence* of dominant motion above thresholds, and minor camera jitter or object-induced apparent motion can easily trigger false positives. Similarly, *pan* and *tilt* are moderately harder than most translation primitives, as they are more easily confounded with arc-like camera paths and with subtle changes in viewpoint.

Confusion analysis. Fig. 4 reveals that residual errors of the VGGT classifier are *highly structured* and dominated by a few recurring confusion patterns rather than broadly distributed noise.

First, there is a pronounced tendency to over-predict *dolly out* as a false positive across multiple ground-truth classes: both *crane up* and *crane down* are frequently mislabeled as *dolly out* (*e.g.*, 15 and 9 counts, respectively), and similarly for *truck left/right* (11 counts each) and *tilt up/down* (8–7 counts). This suggests that, in some segments, outward depth change acts as an “attrac-

tor” class, likely because multiple camera trajectories contain a depth component and the mapping reduces a continuous pose trajectory to discrete primitives.

Second, we observe systematic confusions among coupled motions and axis-aligned primitives. In particular, `arc cw/ccw` is often confused with `dolly out` and with horizontal motion primitives (e.g., `pan right`), consistent with the fact that `arc` combines yaw rotation with lateral translation and may also include a depth component depending on the trajectory. For example, `arc cw` is frequently predicted as `dolly out` or `pan right`, while `arc ccw` shows confusion with `truck right` and `dolly out`.

Third, the largest single off-diagonal entry corresponds to `dolly in` being misclassified as `pan right` (16 counts), indicating that certain trajectories exhibit concurrent yaw change and forward motion where the dominant primitive is ambiguous at 1-second granularity. Similar interactions appear between `tilt` and translation primitives, where pitch changes co-occur with camera motion and lead to mixed predictions.

Finally, `static` errors are sparse but concentrated: when misclassified, `static` is most often predicted as `dolly out` (3 counts), consistent with threshold-boundary cases where small residual translation exceeds the “no dominant motion” criterion.

Overall, the confusion matrix indicates that the remaining failure modes are driven primarily by (i) discretization of continuous camera trajectories into primitives and (ii) co-occurring multi-axis motions where one component (notably depth change) is over-selected. This motivates both the constrained multi-label formulation (to allow co-occurrence) and future improvements in pose-to-label mapping (e.g., better dominance criteria for separating depth change from vertical/lateral translation in short clips).

E. Full Prompts and additional Qualitative Examples

E.1. Full prompt templates

We evaluate how explicit camera-motion cues affect *Videollm* generation using three prompts that share the same video input (a sequence of uniformly sampled frames) and differ only in whether and how motion information is communicated. The *motion-header prompt* follows the structured prompting design in the main paper (Sec. 3.5), where a per-second motion list is prepended as a compact textual scaffold [3].

Baseline prompt (generic description). This is the default VideoLLM usage pattern: the model receives frames and a generic captioning instruction, without any explicit cinematography cues.

```
Here are [N] consecutive video frames.
They are evenly sampled at a frame rate
```

```
of [r] FPS.
```

```
Describe the video clip using clear and
concise language. Make your description
in one paragraph.
```

In practice, this prompt often yields content-correct but camera-vague descriptions, and may conflate camera motion with object motion.

Structured prompt (filmmaker-style instruction). Motivated by the qualitative prompting protocol, we reformulate the instruction in filmmaking language to explicitly request cinematographic attributes (lighting, framing, composition) and to encourage temporal linking across frames.

```
Here are [N] consecutive video frames.
They are evenly sampled at a frame rate
of [r] FPS.
```

```
Describe this video using the filmmaker’s
language, highlighting the lighting,
framing, video composition, and
especially camera usage that connects
different frames. For example: ``At the
beginning, <video content>; then <camera
motion>, <video content>; ...; finally,
<camera motion>, <video content>``. Make
your description in a paragraph.
```

This structure improves the *intent* to discuss camera usage, but it still requires the VideoLLM to infer motion direction and temporal evolution solely from its visual representation.

Camera motion injected structured prompt. We inject predicted motion primitives as a short header before the same filmmaker-style instruction. Following Sec. 3.5 of the main paper, a shot with S one-second segments is serialized as a list Per-second camera motion: $[m_1, \dots, m_S]$, where each m_s is a canonicalized set of primitives (e.g., `static` or `pan left + tilt up`).

```
Here are [N] consecutive video frames.
They are evenly sampled at a frame rate
of [r] FPS.
```

```
Per-second camera motion: [m1, m2, ...,
mS].
```

```
Describe this video using the filmmaker’s
language, highlighting the lighting,
framing, video composition, and
especially camera usage that connects
different frames. For example: ``At the
beginning, <video content>; then <camera
motion>, <video content>; ...; finally,
<camera motion>, <video content>``. Make
your description in a paragraph.
```

This modification is *training-free*: it does not change VideoLLM weights or architecture, but conditions generation on an explicit motion prior.

We design the motion header to be *compact*, *structured*, and *physically grounded* (primitive taxonomy with constraints). This choice is motivated by two empirical observations. First, probing shows that camera motion is

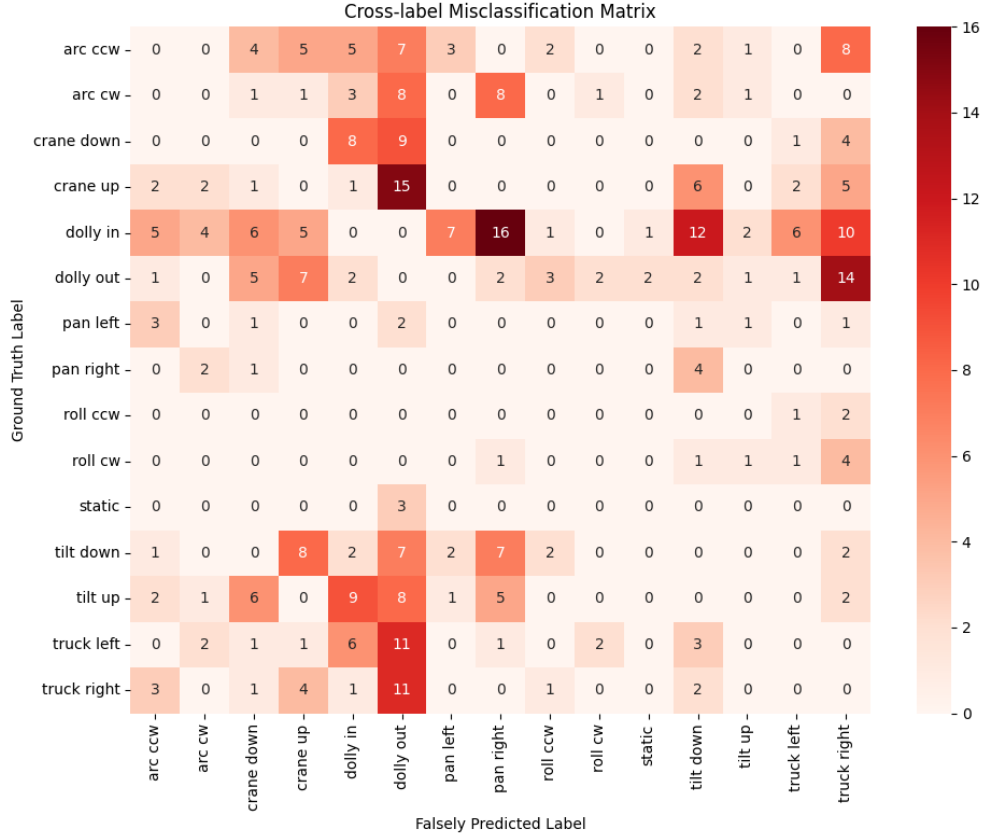


Figure 4. **Cross-label confusion for the VGGT classifier.** Rows denote ground-truth primitives and columns denote falsely predicted primitives (higher intensity indicates more frequent confusion). The label order matches Tab. 3 and the dataset primitive indices.

only weakly recoverable from frozen VideoLLM vision features, making direction and temporal consistency brittle without additional cues. Second, camera motion is intrinsically compositional and time-indexed; a per-second list matches the segment granularity of our benchmark and provides a stable alignment target for the language model. As discussed in the main paper, providing explicit per-second motion labels encourages temporally grounded narratives, reduces motion-direction hallucinations, and biases the model toward continuity-aware reasoning (*e.g.*, linking subject changes to camera transitions) rather than producing generic cinematography statements.

E.2. More qualitative examples

To further illustrate the effect of camera-motion prompting, we provide additional qualitative examples from *CineTech-Bench* [7]. For each clip, we compare the outputs produced by the three prompt variants described in Sec. E.1: *baseline*, *structured*, and our *motion-header* prompt.

Figures 5 (Clip2) and 6 (Clip3) show sampled frames from two example clips. Together with the example shown in the main paper (Clip1), we present three representative

cases covering different camera motions and scene types. For each clip, we report the generated descriptions under the three prompt settings, resulting in nine outputs in total (Tab. 4, 5, 6). Key phrases related to camera motion are highlighted for comparison.

Across the three examples, the baseline prompt primarily focuses on scene content and rarely provides reliable descriptions of camera motion. The structured prompt encourages the model to discuss cinematographic aspects, but the inferred motion cues are often ambiguous or incorrect. For instance, in Clip1, the structured prompt incorrectly describes the motion as a *whip pan to the right*, while the injected motion header leads the model to correctly describe alternating *pan-left* and *pan-right* movements that match the camera dynamics. In Clip2, the structured prompt identifies a rolling camera but does not clearly specify its direction or temporal progression, whereas the motion-header prompt explicitly grounds the description in a *clockwise roll* combined with *trucking* and *dolly-out* movements, producing a more precise and temporally coherent narrative. In Clip3, the structured prompt relies on loosely defined cinematographic terms such as “push-in” or “tracking for-



Figure 5. Clip2 sampled frames from CineTechBench [7].



Figure 6. Clip3 sampled frames from CineTechBench [7].

Table 4. Qualitative comparison of model-generated descriptions for Clip1 under different prompt templates. Camera motion-related phrases are highlighted in bold.

Baseline prompt	Structured prompt (filmmaker-style instruction)	Camera motion injected structured prompt
<p>This video portrays a dramatic musical performance scene. It begins with a conductor, dressed in black, gesturing with precision and intensity, likely cueing the start of a musical piece. His facial expression is focused and assertive, signaling a moment of high tension. The camera quickly pans with motion blur, transitioning to a drummer mid-performance, energetically striking the drums while accompanied by a bassist. The motion blur appears again between frame changes, emphasizing swift camera movement or cuts to maintain rhythmic pacing. The conductor reappears in subsequent frames, continuing to direct with expressive hand movements, culminating in another quick blur before returning to the drummer, now deeply immersed in an intense and possibly climactic moment. The alternating shots and dynamic movement suggest a tightly edited sequence synchronized with musical tempo, evoking a sense of urgency and artistic fervor typical of a performance climax.</p>	<p>At the beginning, the video opens with a medium close-up of the conductor, dramatically lit with focused, warm stage lighting that casts soft shadows and isolates him against the dimly lit auditorium, emphasizing his stern facial expression and sharp, deliberate hand gestures. Then, a rapid whip pan to the right (evident from motion blur) bridges the visual transition, dynamically redirecting the viewer's focus to the drummer. The camera settles into a medium-wide shot of the drum set, capturing the drummer mid-performance under similarly warm, directional lighting that accentuates the shimmer of the cymbals and the energy of his movements. The framing includes part of the upright bassist, preserving ensemble context while maintaining the drummer as the focal point. Another whip pan follows, visually echoing the earlier motion and swiftly returning the viewer to the conductor. As the video cuts back, the conductor appears more animated in his movements, with tight framing and consistent lighting maintaining visual continuity. A final whip pan transitions back to the drummer, who now performs with increased intensity, reinforcing the escalating rhythm and emotional tension through repeated visual interplay. The sequence's camera work—marked by fluid whip pans—mirrors the musical tempo, while the lighting and composition maintain dramatic cohesion, heightening the confrontation and synchrony between conductor and performer.</p>	<p>At the beginning, under warm, low-key stage lighting, a static medium closeup frames the conductor from stage right, isolating him against a softly lit auditorium background as he cues the ensemble with precise hand gestures. Then, a pan-left motion bridges the space, transitioning fluidly into a medium shot of the drummer, dynamically lit from above to emphasize the percussive energy and motion blur of his performance. Next, a pan-right swiftly returns the focus to the conductor, capturing his sharp directional cues in a near-symmetrical composition that mirrors the earlier frame, reinforcing continuity through motion. A static shot then holds on the conductor, now mid-gesture, in tighter framing that heightens dramatic tension while maintaining spatial coherence. Finally, a pan-left reintroduces the drummer in a more intensified moment—his kinetic energy visually amplified through rhythmic lighting and a centered composition that anchors the visual climax. The sequence skillfully employs lateral camera movement and repeated framing motifs to synchronize the visual rhythm with the musical performance, creating a cohesive and propulsive cinematic language.</p>

Table 5. Qualitative comparison of model-generated descriptions for Clip2 under different prompt templates. Camera motion-related phrases are highlighted in bold.

Baseline prompt	Structured prompt (filmmaker-style instruction)	Camera motion injected structured prompt
<p>The clip shows a wounded man lying motionless on the forest floor among ferns and leaves, his face bruised and smeared with blood, suggesting he has been injured or unconscious. The camera gradually shifts its framing across the scene, revealing more of his body and the surrounding foliage as his head slowly tilts to the side. After remaining still for several moments, he begins to stir slightly, moving his head and hand as if regaining consciousness. Toward the end of the clip, another person enters from the left side, reaching toward him and leaning over his body, apparently checking on him or attempting to help him while the injured man weakly reacts.</p>	<p>At the beginning, the shot opens in a dim, cool-toned forest, lit by soft bluish ambient light. A tight overhead close-up frames a wounded man lying among ferns and damp leaves, his face near the center while foliage fills the surrounding space. As the sequence progresses, the camera slowly drifts laterally while rolling, shifting the man's face from a frontal view to a slightly oblique angle. The framing gradually widens from a centered close-up to a looser, off-center composition, revealing more of his shoulder and the forest floor. The continuous roll with gentle tracking slides his head toward the lower part of the frame while foliage expands across the upper area, creating a drifting, hovering sensation. Toward the end, the camera dollies slightly outward while maintaining the roll, widening the overhead view as another person enters from the left and leans toward the injured man. The shot settles into a loose overhead two-shot, both figures partially visible among the ferns, with the slow rolling camera motion unifying the sequence into a calm but tense visual moment.</p>	<p>The shot opens in a dim, cool-toned forest with soft bluish lighting. A wounded man lies among ferns and damp leaves, framed in a tight overhead close-up that centers on his face and upper torso. The camera rolls slowly clockwise, keeping the close framing while rotating the horizon so the foliage appears to drift around him, creating a disorienting sense of instability. Midway through, the motion becomes more complex: the camera continues the clockwise roll while tracking left and craning up, gradually widening the view and revealing more of the forest floor. Soon after, the camera maintains the roll and begins a gentle dolly out, expanding the overhead composition. As the frame widens, another figure enters from the left, kneels beside the injured man, and reaches toward him. The shot ends in a wider overhead view, with both figures visible among the ferns as the slow rolling, outward-moving camera connects the sequence into a somber, atmospheric moment.</p>

ward,” which describe similar motion but lack a consistent taxonomy and omit the camera roll entirely. In con-

trast, the motion-header prompt enforces explicit primitives such as *dolly-in* and *roll clockwise*, yielding a more uni-

Table 6. Qualitative comparison of model-generated descriptions for Clip3 under different prompt templates. Camera motion-related phrases are highlighted in bold.

Baseline prompt	Structured prompt (filmmaker-style instruction)	Camera motion injected structured prompt
<p>A man wearing a long dark coat walks forward through a grand, modern throne room with floor-to-ceiling windows overlooking a futuristic city. Several guards in coordinated red uniforms stand symmetrically on both sides of the room, holding spears and watching as he approaches a large, curved ceremonial throne positioned at the center. Other armored attendants and officials stand nearby, observing quietly. The man slows as he reaches the throne, turns slightly, then lowers himself into the seat, gripping the armrests and settling into a commanding posture while the guards remain lined up behind him, reinforcing the formal and authoritative atmosphere of the scene.</p>	<p>At the beginning, a slightly canted medium-wide establishing shot inside a grand, glass-walled hall, where bright natural daylight floods in from the floor-to-ceiling windows behind the characters, producing strong backlighting and soft silhouettes. In the foreground, a man in a long dark coat walks toward a large circular throne-like structure positioned at the center of the frame, while symmetrical rows of red-armored guards stand rigidly in the background, forming a visual corridor that guides the viewer’s eye inward. As the shot progresses, the camera slowly tracks forward behind him, gradually straightening its angle and tightening the framing, turning the initial off-axis composition into a more centered, ceremonial perspective. The circular throne becomes a dominant compositional element, framing the character as he approaches; the guards remain evenly spaced on both sides, reinforcing the symmetry and depth of the mise-en-scène. Continuing the forward push-in, the camera subtly lowers and stabilizes into a frontal alignment as the man reaches the throne, placing him precisely on the central axis of the architecture and the guard formation. He then turns and lowers himself into the seat, the movement captured in a steady medium shot that emphasizes the ritualistic nature of the action. Finally, as he settles into the throne and grips the armrests, the camera completes its push-in to a centered, balanced composition: the character sits framed by the curved throne behind him, flanked symmetrically by the guards and the luminous cityscape beyond the windows, creating a powerful throne-room tableau under strong backlighting.</p>	<p>At the beginning, a slightly canted wide shot reveals a grand, glass-walled hall flooded with soft daylight from floor-to-ceiling windows, the bright skyline forming a luminous backdrop while a group of red-armored guards stands in symmetrical formation. A man in a dark coat walks toward a sculptural throne framed by two curved metallic blades. As the camera dollies in while rolling clockwise, the off-axis framing creates a subtle sense of motion and tension, gradually tightening the composition around the man’s back and the throne. The dolly-in with clockwise roll continues, bringing the throne into clearer alignment behind him while the guards remain evenly spaced on both sides, their red armor catching rim light from the windows. The lighting stays high-key and backlit, outlining figures with soft highlights while the foreground remains slightly shadowed, adding depth. As the camera keeps dolling in and rolling clockwise, the frame slowly levels and the throne becomes centered in the composition. The man turns and lowers himself toward the seat, the curved blades of the throne visually framing his body. Finally, the dolly-in with clockwise roll resolves into a centered medium shot, where he settles into the throne and grips the armrests, with the guards standing symmetrically behind him and the bright cityscape still glowing through the windows, completing the camera’s smooth approach and stabilizing the composition around his newly assumed position of authority.</p>

fied and geometrically grounded description. These examples highlight that while structured prompting alone encourages camera-aware language, providing explicit motion primitives substantially improves directional correctness, terminological consistency, and temporal grounding in VideoLLM-generated descriptions.

F. Additional Discussion

F.1. Synthetic-to-real gap

Our benchmark is deliberately built on synthetic data from MultiCamVideo [1] to obtain ground-truth camera extrinsic parameters that are unavailable in real-world datasets. This design choice provides precise, physically grounded labels free from annotation noise, but introduces a domain gap with respect to real-world video characteristics such as lens distortion, motion blur, rolling shutter artifacts, and non-uniform lighting.

We view this synthetic benchmark as a *diagnostic tool* rather than a claim of real-world coverage: it isolates camera motion understanding from confounds that plague manual annotation (e.g., visual illusions caused by object mo-

tion, subjective labeling of ambiguous clips). Limited by time and computational resources, evaluation and exploration of real-world applications are not covered in this work. To mitigate the gap, several strategies can be pursued in future work:

- **Real-world evaluation.** Benchmarks such as CameraBench [3] provide human-annotated camera motion labels for real videos, albeit at coarser granularity and with camera motions not fully defined in the camera coordinate system. Testing our pipeline on CameraBench would quantify how well the VGGT-based classifier generalizes beyond the synthetic domain. We note that VGGT itself is pretrained on diverse real-world data, which may facilitate transfer.
- **Domain adaptation via augmentation.** We outline a data augmentation strategy (Sec. G) that introduces realistic visual effects (zoom, motion blur, camera shake) to narrow the appearance gap.
- **Mixed training.** Future work can combine our synthetic labels with pseudo-labels from real videos (e.g., generated by running SfM on in-the-wild footage) to improve robustness.

We emphasize that the primary contribution of this work is to *inspire the VLM community* to consider explicit embedding or incorporation of camera motion information. The synthetic benchmark serves as a proof of concept demonstrating: (1) current VideoLLMs have a measurable camera-motion blindness; (2) geometric cues from 3DFMs can substantially improve recognition; and (3) structured prompting is a viable injection mechanism. These findings are independent of the data domain and motivate further investigation on real-world data.

F.2. Camera intrinsics and zoom

Our current taxonomy covers 15 primitives derived entirely from camera extrinsic parameters (translation and rotation), intentionally omitting intrinsic changes such as zoom (focal-length variation). This is a notable limitation, as zoom is a fundamental cinematographic technique that is frequently confused with dolly motion [3].

The omission is a deliberate scope choice: Multi-CamVideo provides precise extrinsic trajectories but does not vary focal length across frames, making it unsuitable for zoom annotation. Extending the taxonomy to include zoom requires either:

- **Synthetic data with intrinsic variation.** Rendering engines can simulate zoom by programmatically varying the virtual camera’s focal length. Combined with the existing extrinsic trajectories, this would yield a richer motion space. We outline a zoom simulation strategy via post-hoc cropping in Sec. G.
- **Real-world intrinsic estimation.** Modern 3DFMs and SfM pipelines can jointly estimate camera intrinsics from video; however, disentangling true optical zoom from digital zoom or dolly-induced scale change remains challenging.

We note that our pipeline architecture is agnostic to the number of motion primitives: the multi-label classifier and incompatibility matrix can be extended to include zoom-in and zoom-out as additional labels once appropriate training data is available. In the data augmentation section (Sec. G), we propose a series of augmentations that cover zoom-in and zoom-out simulation (while not reflecting true optical changes). Training on the augmented data does not require any architectural change and can be a practical step toward improving robustness to zoom-like effects in real videos.

F.3. Joint training vs. plug-and-play

A natural question is whether directly feeding VGGT features into a VideoLLM and jointly training on camera motion data would yield better results than our plug-and-play approach. We chose the latter for several reasons:

- **Model agnosticism.** Our pipeline works with any VideoLLM without modifying its weights. Joint training would require per-model adaptation and access to the

original training infrastructure.

- **No VideoLLM fine-tuning cost.** Fine-tuning a 7B+ VideoLLM on camera motion data risks catastrophic forgetting of general video understanding capabilities, requiring careful multi-task training.
- **Modularity and interpretability.** The structured prompt provides an explicit, human-readable interface between camera cues and the VideoLLM, making it straightforward to verify, debug, or override motion predictions.

Recent work such as VLM-3R [2] demonstrates that trainable 3D-enhanced VLMs can achieve strong spatial-temporal understanding by fusing camera tokens during training. We view our plug-and-play approach and trainable approaches as complementary: ours provides an immediate, low-cost solution, while joint training may achieve higher ceilings given sufficient compute and data. Limited by computational resources, the joint-training approach is not adopted and evaluated in this work, while a systematic comparison between these paradigms is an important direction for future work.

F.4. VGGT dependency and practical deployment

The strongest variant of our pipeline relies on VGGT (1.2B parameters), which adds computational overhead. We address this concern through three observations:

- **Distillation reduces cost substantially.** The VGGT-Q-Former distillation achieves $5.3\times$ throughput improvement at 39% peak memory, with only a 10-point drop in instance-level accuracy. This demonstrates distillation as a promising path to making the pipeline practical for applications that require real-time or large-scale inference.
- **The 3DFM ecosystem is rapidly improving.** LiteVGGT [5] already reduces VGGT’s cost through geometry-aware token merging. As lighter 3DFMs emerge, our pipeline directly benefits without architectural changes. Moreover, camera awareness can complement the VLM and VLA communities’ ongoing efforts to incorporate 3D geometry into video understanding.
- **Camera cues are computed offline.** For applications like media indexing, retrieval, or batch captioning, VGGT inference can be amortized as a one-time preprocessing step, after which only the lightweight classifier runs online.

G. Proposed Data Augmentation Strategy

To improve robustness and bridge the synthetic-to-real gap, we propose the following augmentation strategies applicable during classifier training. Each augmentation must respect compatibility with existing camera motion labels (Fig. 7).

1. **Zoom-in / Zoom-out simulation.** Since the source data lacks focal-length variation, we simulate zoom effects



Figure 7. A visualization of the proposed data augmentations. For each augmentation, the left column shows the starting frames and the right column shows the augmented ending frames, because the augmentation is applied to the entire segment.

via progressive center cropping (zoom-in) or padding with border extrapolation (zoom-out) across frames in a segment. The crop ratio is linearly interpolated from 1.0 to a target ratio sampled in $[0.6, 0.9]$ for zoom-in, and reversed for zoom-out. *Compatibility*: zoom-in and dolly-in should not co-occur as labels; an updated incompatibility matrix enforces this constraint.

- Reverse play.** Temporally reversing a segment swaps directional labels (e.g., pan-left \leftrightarrow pan-right, dolly-in \leftrightarrow dolly-out), effectively doubling the training set with consistent labels. *Compatibility*: all directional primitives must be symmetrically swapped.
- Random center cropping.** Cropping with a ratio sampled in $[0.6, 0.9]$ simulates varying fields of view and removes border artifacts. Since the crop is spatially centered and consistent across frames, it does not alter camera motion labels. *Compatibility*: compatible with all motion primitives.

These augmentations are designed to be applied *jointly* with the original camera motion labels, requiring no re-annotation. The augmentation pipeline can be integrated into the training dataloader with minimal overhead. We also considered **camera shake simulation** (per-frame random translations and roll rotations with temporal smoothing) and **motion blur via frame blending** (α -compositing of adjacent frames). However, we did not implement these two augmentations: the degree of shake and blur are difficult to label in a categorical framework, and more importantly, detecting their presence in real footage is efficiently handled by conventional signal-processing methods, making model-based prediction less beneficial for these attributes compared to semantic camera motion primitives.

H. LLM-as-Judge Evaluation Protocol

To enable systematic evaluation of camera-aware video descriptions beyond qualitative inspection, we propose an LLM-as-judge protocol [8]. This protocol uses a large language model to score VideoLLM-generated descriptions along multiple cinematographic dimensions. Fig. 8 provides the full prompt template, which includes detailed scoring criteria for each dimension. The LLM evaluates each description based on its alignment with ground-truth camera motion labels, temporal coherence, reproducibility

Table 7. **LLM-as-judge scores for three prompting strategies.** Scores range from 1 (poor) to 5 (excellent). Injecting camera motion labels (P3) yields the largest gains in cinematographic accuracy and temporal continuity.

Prompt	CA	TC	RD	NM	LQ	Final
P1: Baseline	1	2	1	3	4	1.75
P2: Filmmaker	3	3	4	5	5	3.65
P3: + Motion labels	4	4	3	4	5	3.85

of camera work, narrative tone, and language quality. This approach enables scalable, consistent evaluation that complements human judgment and provides quantitative metrics for future research.

Each generated description is evaluated on five dimensions, each scored from 1 (poor) to 5 (excellent):

- Cinematographic Accuracy (CA).** Does the description correctly identify camera motion types, shot scales, and framing? Are technical terms (pan, tilt, dolly, truck, crane, static) used accurately and consistently with the ground-truth motion labels?
- Temporal Continuity (TC).** Does the description follow the temporal progression of the video? Are camera motions described in the correct chronological order? Does it capture transitions between different motion phases?
- Reproducibility Detail (RD).** Could a cinematographer recreate the described camera work from the description alone? Are directions, durations, and spatial relationships specified precisely enough?
- Narrative Mood & Tone (NM).** Does the description convey the artistic intent and emotional quality of the camera work? Does it connect camera choices to narrative effect?
- Language Quality (LQ).** Is the description fluent, well-structured, and free of contradictions? Does it use professional cinematographic vocabulary appropriately?

H.1. Preliminary Results

To validate the evaluation protocol, we apply it to descriptions generated by GPT-4o from a single example clip (Fig. 7 in the main text). We compare the three prompting strategies used in the qualitative analysis:

- P1. Baseline:** “Describe this video in detail in a paragraph.”
- P2. Filmmaker’s language:** Request a description using cinematographic terminology (lighting, framing, composition, camera usage) with a temporal structure template.
- P3. Filmmaker’s language + motion labels:** Same as P2, with ground-truth per-second camera motion labels prepended (e.g., [static, pan-left, pan-right, static, pan-left]).

Several observations emerge. The baseline prompt (P1) produces fluent but cinematographically vague descriptions,

LLM-as-Judge Prompt Template

You are an expert cinematographer evaluating a video description. You are given:

- Ground-truth per-second camera motion labels: [{motion.labels}]
- The generated video description: [{description}]

Score the description from 1 to 5 on each of the following dimensions:

1. Cinematographic Accuracy (CA)

Assess whether the description correctly and precisely identifies camera angle, framing, lens focal length, camera movement (direction and speed), lighting type and direction, and composition.

- 5: All key elements correct, using industry-standard terminology.
- 4: Mostly accurate with minor omissions or ambiguity.
- 3: Some inaccuracies or generalizations; partially correct terminology.
- 2: Several incorrect elements; relies on non-technical language.
- 1: Inaccurate or vague; lacks meaningful cinematographic detail.

2. Temporal Continuity (TC)

Measure how clearly the description communicates the sequence of events, camera transitions, and rhythmic or pacing logic throughout the scene.

- 5: Seamless, accurate timeline; per-shot motion and transitions are logically and temporally clear.
- 4: Mostly clear sequence with minor inconsistencies or vague transitions.
- 3: Partial timeline or implied transitions; lacks precise continuity.
- 2: Confusing or incorrect sequence; missing or inaccurate motion description.
- 1: Lacks chronological structure; incoherent scene progression.

3. Reproducibility Detail (RD)

Evaluate whether the description provides sufficient technical detail for a filmmaker to recreate the scene, including shot scale, camera positioning, blocking, movement mechanics, lighting setup, frame duration, and scene geometry.

- 5: Fully reproducible; contains all necessary technical elements.
- 4: Mostly reproducible; a few minor missing parameters.
- 3: Some useful information; key elements (e.g., camera height, lighting angles) are missing.
- 2: Vague and incomplete; only general scene description present.
- 1: Not reproducible; lacks all relevant production-level detail.

4. Narrative Mood & Tone (NM)

Measure the description's ability to convey the emotional atmosphere, dramatic intention, and cinematic tone (e.g., tension, melancholy, exuberance), as well as the subjective energy of the scene.

- 5: Strong emotional clarity aligned with visual style and editing rhythm.
- 4: Clear tone with some expressive nuance.
- 3: Functional tone description; lacks depth or resonance.
- 2: Tone is underdeveloped or inconsistent.
- 1: No discernible mood or emotional quality.

5. Language Quality & Fluency (LQ)

Assess overall clarity, precision, and fluency of the writing, with a focus on cinematographic vocabulary, cohesion, and technical correctness in grammar and syntax.

- 5: Polished, technically fluent, and lexically precise.
- 4: Clear and correct with minor stylistic or lexical weaknesses.
- 3: Mostly clear; occasional awkward phrasing or imprecise terminology.
- 2: Grammatically flawed or inconsistent in tone or register.
- 1: Unclear, ungrammatical, or non-professional.

For each dimension, provide: (1) Score (1--5), and (2) One-sentence justification.

Finally, compute the weighted average:

$$\text{Final} = 0.30 \cdot \text{CA} + 0.25 \cdot \text{TC} + 0.25 \cdot \text{RD} + 0.10 \cdot \text{NM} + 0.10 \cdot \text{LQ}$$

Figure 8. The full prompt template for the LLM-as-judge evaluation protocol.

scoring lowest on accuracy and reproducibility. Requesting filmmaker's language (P2) substantially improves reproducibility and narrative tone, but camera motion directions remain incorrect (e.g., describing a "whip pan to the right"

when ground truth is pan-left), limiting the accuracy and temporal continuity scores. Injecting ground-truth motion labels (P3) corrects these directional errors and improves temporal alignment, raising the final score from 3.65 to

3.85. The largest per-dimension gain appears in cinematographic accuracy (+1) and temporal continuity (+1), directly validating the core hypothesis of our structured prompting approach: providing explicit camera motion cues enables more precise and faithful video descriptions.

These preliminary results, while based on a single example, demonstrate that the evaluation protocol can meaningfully discriminate between prompting strategies of varying informativeness, and that the structured motion header is the key factor driving improvement. We leave large-scale evaluation across the full dataset to future work.

References

- [1] 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. *arXiv preprint arXiv:2503.11647*, 2025. 11
- [2] Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Huaizhi Qu, Dilin Wang, Zhicheng Yan, et al. Vlm-3r: Vision-language models augmented with instruction-aligned 3d reconstruction. *arXiv preprint arXiv:2505.20279*, 2025. 12
- [3] Zhiqiu Lin, Siyuan Cen, Daniel Jiang, Jay Karhade, Hewei Wang, Chancharik Mitra, Tiffany Ling, Yuhan Huang, Sifan Liu, Mingyu Chen, et al. Towards understanding camera motions in any video. *arXiv preprint arXiv:2504.15376*, 2025. 1, 3, 7, 11, 12
- [4] Mattia Savardi, András Bálint Kovács, Alberto Signoroni, and Sergio Benini. Cinescale2: a dataset of cinematic camera features in movies. *Data in Brief*, 51:109627, 2023. 3
- [5] Zhijian Shu, Cheng Lin, Tao Xie, Wei Yin, Ben Li, Zhiyuan Pu, Weize Li, Yao Yao, Xun Cao, Xiaoyang Guo, et al. Litevsgt: Boosting vanilla vsgt via geometry-aware cached token merging. *arXiv preprint arXiv:2512.04939*, 2025. 12
- [6] 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 *CVPR*, pages 8490–8500, 2025. 3
- [7] Xinran Wang, Songyu Xu, Xiangxuan Shan, Yuxuan Zhang, Muxi Diao, Xueyan Duan, Yanhua Huang, Kongming Liang, and Zhanyu Ma. Cinetechbench: A benchmark for cinematographic technique understanding and generation. *arXiv preprint arXiv:2505.15145*, 2025. 3, 8, 9
- [8] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023. 13