Which Way Did It Move? Diagnosing and Overcoming Directional Motion Blindness in Video-LLMs

Jongseo Lee[1*] **Hyuntak Lee**[1*] **Sunghun Kim**[1] **Sooa Kim**[1] **Jihoon Chung**[2] **Jinwoo Choi**[1†]
[1]Kyung Hee University [2]Princeton University
{jong980812, takhyun03, rlatjdgns0816, suhakim12, jinwoochoi}@khu.ac.kr
jc5933@princeton.edu

Abstract

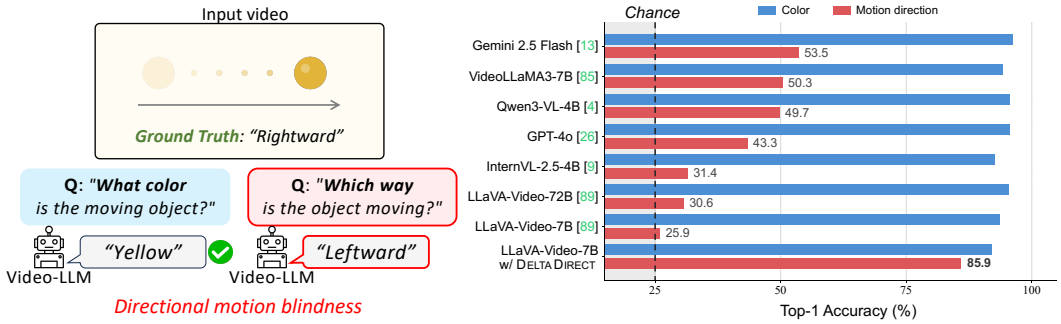
Video Large Language Models (Video-LLMs) have made rapid progress on temporal video understanding, yet many fail at a basic perceptual primitive: signed image-plane motion direction. On simple videos of a single object moving left, right, up, or down, most Video-LLMs perform near chance, with above-chance cases largely attributable to prediction biases rather than genuine direction understanding. We call this failure *directional motion blindness*. We localize the failure by tracing motion direction information through the Video-LLM pipeline. Motion direction remains linearly accessible from the vision encoder, projector, and LLM hidden states, but the readout fails to bind this signal to the correct verbal answer option, revealing a *direction binding gap*. Although synthetic motion direction instruction tuning reduces this gap on the source domain, motion direction concept vector analysis shows that visual complexity weakens the signal magnitude and limits out-of-domain generalization. We introduce MoDirect, a dataset family for motion direction instruction tuning and evaluation, and `DeltaDirect`, a diagnosis-driven, projector-level objective that predicts normalized 2-D motion vectors from adjacent-frame feature deltas. On MoDirect-SynBench, instruction tuning with `DeltaDirect` improves motion direction accuracy from 25.9% to 85.9%. On MoDirect-RealBench, `DeltaDirect` improves real-world motion direction accuracy by 21.4 points over the vanilla baseline without real-world tuning data, while preserving standard video-understanding performance. Our project page is available at https://jong980812.github.io/which-way-did-it-move/.

1 Introduction

Recent work on Video-LLMs has made rapid progress in temporal understanding [23, 40, 48, 59, 64] with different benchmarks analyzing a model's temporal capability by testing its memory [58], ability to identify complex human action or object movements [48, 62, 64], identifying the order of stitched videos [48, 91], or understanding long-form videos [17, 51, 94]. Despite this progress, we find that many Video-LLMs [3, 11, 37, 39, 43, 44, 73, 88] fail at a much simpler temporal primitive: motion direction understanding. Consider a seemingly trivial synthetic video in which a yellow circle moves from left to right, as shown in Figure 1. Can a Video-LLM correctly tell which way it moves: left→right, right→left, bottom→top, or top→bottom? To our surprise, most Video-LLMs show accuracy close to random chance (25%), with Gemini2.5 Flash [13] only reaching 53.5% accuracy. Humans identify such motion directions instantly and effortlessly, reflecting the fundamental role of motion direction in visual perception and physical navigation [6, 20, 53]. We refer to this systematic failure to distinguish signed motion directions as *directional motion blindness*.

[*]Equally contributed first authors.
[†]Corresponding author.

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



(a) (b)

Figure 1: **Directional motion blindness in Video-LLMs.** (a) Given a simple synthetic video of a yellow circle moving from left to right, recent Video-LLMs correctly identify the object's color but answer the wrong motion direction. (b) Across Video-LLMs, appearance recognition is high, yet signed motion direction accuracy remains much lower, often near chance.

To understand where this failure arises, we trace motion direction information across the Video-LLM pipeline. We find that the signal remains linearly accessible throughout the pipeline: motion direction is decodable from the visual encoder, the projector output, and intermediate LLM states. The breakdown emerges at the readout position, i.e., the final token representation used to predict the answer: although a motion direction signal is still linearly accessible, the model does not reliably connect it to the correct verbal response [52, 56]. We call this mismatch a *direction binding gap*, and observe the same pattern across multiple Video-LLMs (Figure 3). This suggests that directional motion blindness is not simply a failure of visual perception, but a failure to make an available motion signal usable for language-level readout.

We therefore introduce MoDirect, a dataset family for signed image-plane motion direction, with three subsets: MoDirect-Inst, MoDirect-SynBench, and MoDirect-RealBench. Its synthetic part contains four domains under a controlled 2×2 design over foreground type and background type: Primitive-on-Syn, Cutout-on-Syn, Primitive-on-Real, and Cutout-on-Real. Primitive videos contain rendered geometric primitives such as circles, triangles, and squares; Cutout videos contain segmented real-world object images [45]. Synthetic videos use uniform-color backgrounds, whereas Real videos use natural scene backgrounds [92]. From the simplest synthetic domain, Primitive-on-Syn, we construct MoDirect-Inst for instruction tuning. We use all four synthetic domains as MoDirect-SynBench for controlled synthetic evaluation, and use MoDirect-RealBench for real-video evaluation.

Instruction tuning on MoDirect-Inst substantially improves on the source domain motion direction recognition. However, models that perform well on Primitive-on-Syn still degrade on more complex domains, especially Cutout-on-Real. This indicates that next-token prediction on motion direction instructions can improve answer binding on the source domain, Primitive-on-Syn, but does not necessarily produce a strong domain-invariant motion direction signal.

We next ask why this generalization gap persists. Using difference-in-means [1, 38, 52, 60, 68], we extract motion direction concept vectors from the readout position of each LLM layer for each domain and motion direction. After instruction tuning, motion direction concept vectors become well aligned across domains, despite being trained only on Primitive-on-Syn; for example, at certain layers, their cross-domain cosine similarity is much higher than in the baseline without motion direction instruction tuning. Yet their magnitude drops sharply on complex domains, mirroring the observed accuracy drop. Moreover, restoring only the magnitude of the motion direction concept vector recovers much of the lost accuracy. These results suggest that the out-of-domain failure is not due to a missing motion direction geometry, but to a *magnitude deficit*: the model learns the motion direction structure, but the signal is too weak to be reliably read out across domains. Please refer to Section 4 for more details.

This diagnosis motivates `DeltaDirect`, a training-only projector-level objective that strengthens signed displacement cues at the visual-language interface. Rather than adding motion tokens or a motion-specific encoder at inference time, `DeltaDirect` uses analytically available 2-D motion vectors from MoDirect-Inst as supervision during training. Specifically, `DeltaDirect` predicts normalized 2-D motion vectors from adjacent-frame projector-feature deltas, encouraging the projector output to carry a stronger signed displacement signal before it enters the LLM. The aux-

iliary branch is discarded after training, leaving the test-time input format, token sequence, model architecture, and decoding procedure unchanged.

We evaluate `DeltaDirect` across synthetic, real-world, and general video understanding benchmarks. On MoDirect-SynBench, instruction tuning on MoDirect-Inst improves over the zero-shot model, and adding `DeltaDirect` further improves average motion direction accuracy by 6.6 points, with the largest gain of 11.5 points on Cutout-on-Real. To test real-world transfer, we construct MoDirect-RealBench, a motion direction benchmark curated from Something-Something-V2 (SSv2) [21], TOMATO [64], and KTH [63]. Without seeing real-world videos during tuning, `DeltaDirect` improves accuracy by 21.4 points over the vanilla baseline. Beyond motion direction-specific benchmarks, `DeltaDirect` preserves or improves performance on standard video understanding benchmarks [23, 39, 48, 51, 58, 70, 86], suggesting positive transfer rather than overfitting to synthetic motion.

Finally, when applied during full fine-tuning, `DeltaDirect` alone yields strong improvements on both motion direction and general video benchmarks, indicating that motion-vector supervision can serve as a useful training signal for future Video-LLMs.

In this work, our major contributions are threefold:

- We identify *directional motion blindness* in Video-LLMs: despite strong appearance recognition, recent models fail to distinguish basic signed motion directions such as left, right, up, and down. We diagnose this failure as a *direction binding gap*, where motion direction information is linearly decodable but not reliably bound to the correct verbal response.
- We show that instruction tuning closes the direction binding gap on the source domain but leaves an out-of-domain generalization gap. Through concept vector analysis, we trace this gap to a magnitude deficit in shared motion direction representations across domains.
- We introduce MoDirect, a controlled dataset family for motion direction instruction tuning and evaluation, and propose `DeltaDirect`, a training-only auxiliary objective that predicts motion vectors from adjacent-frame feature deltas. `DeltaDirect` improves motion direction understanding while preserving or improving general video understanding.

2 Related Work

Video-LLMs and temporal-motion benchmarks. Video-LLMs extend image-based MLLMs to temporal understanding through video instruction tuning, unified image-video architectures, and long-video modeling [3, 4, 12, 13, 26, 43, 44, 46, 49, 50, 73, 85]. Existing benchmarks evaluate temporal memory, action and event understanding, temporal ordering, and long-form video reasoning [14, 39, 48, 51, 64]. Motion direction appears in several temporal or motion-centric benchmarks, often alongside speed, order, rotation, and broader motion reasoning [16, 23, 41, 70, 77, 86, 95]. These benchmarks are valuable for real-world evaluation, but real videos entangle apparent direction with camera motion, occlusion, object identity, scene layout, scale change, and event semantics. We instead separate diagnosis from transfer: controlled synthetic videos *isolate signed image-plane direction*, and real-world benchmarks test whether the intervention transfers.

Mechanistic analysis of LLMs. A growing line of work analyzes internal representations rather than final outputs alone. Probing and difference-in-means analyses show that model states can encode information that is not reliably verbalized [52, 56]. For Video-LLMs, recent work begins to trace layer-wise information flow and temporal bottlenecks inside the model [29, 40]. Our analysis is complementary: we isolate motion direction as a single perceptual primitive and ask *whether the signal is present, where it weakens, and how it couples to verbal readout*.

Motion representations and supervision. Conventional video models have explicitly modeled motion through two-stream architectures, frame differences, pose sequences, optical flow, and trajectories for action recognition [31, 32, 34–36, 42, 65, 71], flow distillation [15, 66, 96], and self-supervised learning [25, 79]. More recently, Video-LLMs have incorporated motion cues through motion-vector tokenization [28], compressed-video representations [90], optical flow [47, 54], and motion-appearance disentanglement [84]. `DeltaDirect` instead uses motion vectors only as training-time supervision on projector-feature deltas, without changing the test-time input or decoding pipeline.

3 Where Does Directional Motion Blindness Arise in Video-LLMs?

We investigate a Video-LLM, LLaVA-Video [89], from visual representations to the answer token, and test whether the failure is due to missing supervision, insufficient prompting, visual encoding,

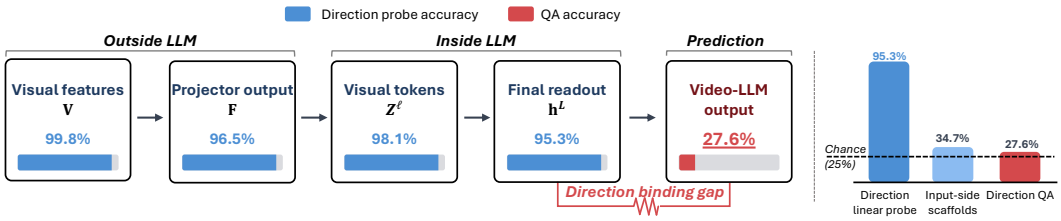


Figure 2: **Direction is decodable, but not converted into the answer.** On Primitive-on-Syn in MoDirect-SynBench, motion direction remains linearly decodable throughout LLaVA-Video-7B [89]. However, video-QA accuracy stays near chance, even with input-side scaffolds. We refer to this mismatch as the *direction binding gap*.

projection, or the final binding between a perceived direction and the corresponding answer option. We also investigate several Video-LLMs and draw the same conclusions. For more results and details, please refer to Appendix D.

We organize the analysis around three research questions. **(i)** Can the failure be explained by missing direction supervision or insufficient input-side scaffolding? (Section 3.2) **(ii)** Is the signed direction erased by the vision encoder, vision-language projector, or the intermediate LLM layers? (Section 3.3) **(iii)** If direction remains accessible, can the final readout token bind it to the prompt-specific answer option? (Section 3.4) We find that motion direction remains linearly accessible across the vision encoder, projector, and LLM hidden states, but the final readout token fails to encode the mapping from direction to answer option. We call this mismatch a direction binding gap.

3.1 Experimental Setup

Datasets. We introduce MoDirect-SynBench, a controlled synthetic benchmark for signed image-plane motion direction. MoDirect-SynBench follows a 2×2 design over foreground type and background type: Primitive-on-Syn, Cutout-on-Syn, Primitive-on-Real, and Cutout-on-Real. Each video contains a single foreground object moving in one of four signed directions: left, right, up, down. We randomize start and end positions within the valid frame region and use held-out start–end trajectories for evaluation, reducing static-position shortcuts. In every analysis on MoDirect-SynBench, we train a linear probe on the training split and evaluate it on held-out videos.

Task. We use a four-way multiple-choice question (MCQ) format as the primary evaluation protocol. The four options correspond to left→right, right→left, bottom→top, and top→bottom. Crucially, we randomize the order of answer options for each example. Therefore, the correct answer letter is not a fixed alias of the motion direction; the model must bind the perceived direction to the prompt-specific option text. This design enables single-token, logit-based evaluation and reduces free-form generation confounds, while directly testing whether the model can bind the perceived direction to the correct prompt-specific option [29, 39, 61].

Video-LLM pipeline. A Video-LLM processes a video through three stages. Given T sampled frames, a vision encoder produces visual features $\mathbf{V} \in \mathbb{R}^{T\times M\times D_v}$, where M denotes the number of patch tokens per frame and D_v denotes the embedding dimension. A projector maps these features into the LLM embedding space, producing visual tokens $\mathbf{F} \in \mathbb{R}^{T\times N\times D}$, where $N \leq M$ reflects spatial pooling and D is the LLM hidden dimension. An L-layer LLM consumes the visual tokens with text tokens and produces an answer. Inside the LLM, we track the hidden states at the visual-token positions across layers. Let $\mathbf{z}_t^\ell \in \mathbb{R}^D$ denote the spatially pooled hidden state of the visual-token positions at frame t and layer $\ell \in \{1,\ldots,L\}$.

We refer to the hidden state at the last token position used for next-token prediction as the *readout token*, and denote $\mathbf{h}^\ell \in \mathbb{R}^D$ for the readout token at layer ℓ. We call $\mathbf{h}^L$ the *final readout token*; this is the representation the Language Model (LM) head maps to output vocabulary logits.

3.2 Can Data or Prompting Explain the Failure?

Before inspecting model internals, we rule out two external explanations. First, motion direction supervision is scarce in existing video instruction data. Using keyword pre-filtering followed by semantic classification, we estimate that only 0.91% of LLaVA-Video-178K [89] examples are direction-dependent; human verification further indicates that these estimates are upper bounds (Appendix D.1). This scarcity motivates controlled motion direction supervision, but it does not localize

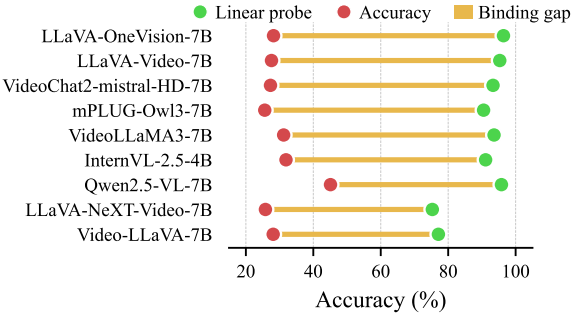


Figure 3: **The direction binding gap is universal across Video-LLMs.**

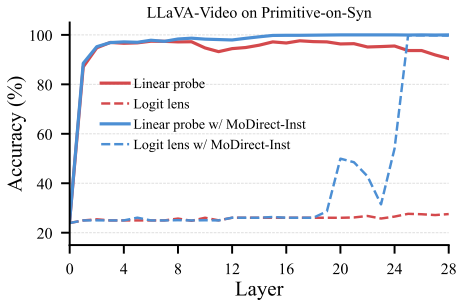


Figure 4: **Instruction tuning closes the direction binding gap on the source domain.**

where the failure occurs. Second, simple input-side scaffolds do not solve the task. We test visual boundary cues, step-by-step location reasoning [30, 74], and coordinate-grid prompts [7, 10, 82]; the best combination reaches only 34.7% on Primitive-on-Syn as shown in Figure 2. Thus, external factors alone are insufficient, and we turn to representation-level diagnostics.

3.3 Is Motion Direction Erased by the Vision Encoder or Projector?

We next ask whether the failure is caused by visual processing. We use linear probing as a diagnostic tool [22, 38, 56]: if a linear classifier can recover the signed direction from a frozen representation, then that representation contains direction information in a linearly accessible form. This does not imply that the model can directly use the signal for generation; rather, it tests whether the visual-temporal evidence is present before the final readout.

Vision encoder preserves direction. We train a four-way linear probe on the frozen vision-encoder output $\mathbf{V}$. The probe recovers motion direction with 99.8% accuracy (Figure 2), indicating that the vision encoder preserves a near-perfect direction signal.

Projector also preserves direction. We repeat the same probe on the projector output $\mathbf{F}$. Accuracy remains high at 96.5%, indicating that the vision-language projection step does not erase the signed direction signal. Thus, directional motion blindness is not explained by a lack of linearly accessible direction information in either the vision encoder or the projector. The failure must arise from *how this signal is used or read out by the LLM*.

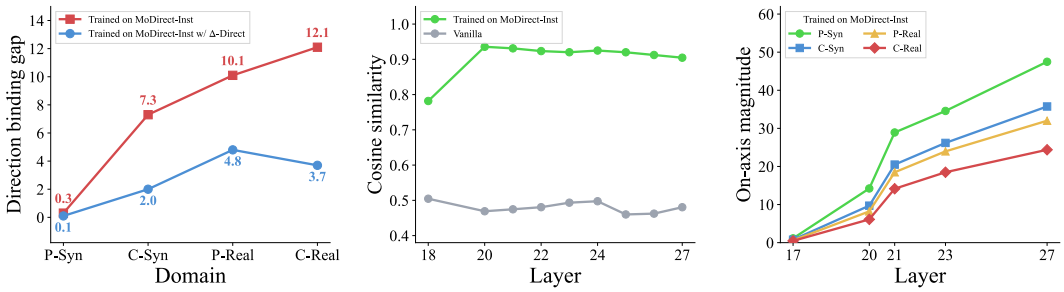
3.4 Can the Readout Bind Motion Direction to the Answer Option?

Direction information reaches the LLM input in a linearly accessible form. We now ask whether it remains accessible inside the LLM and whether the readout token binds this direction signal to the correct prompt-specific answer option.

Direction remains decodable inside the LLM. We probe two internal representations across LLM layers: the visual tokens $\mathbf{z}_t^\ell$ and the readout token $\mathbf{h}^\ell$. As shown in Figure 2, visual-token probes achieve $98.1\% \pm 1.3$, and readout token probes achieve $95.3\% \pm 2.4$. These results show that the LLM does not simply discard the direction signal.

The bottleneck is answer-option binding. The key question is whether the internally decodable direction signal reaches the correct *answer option*. Because the MCQ option order is randomized, the answer letter is not a fixed alias of the motion direction; the model must bind the perceived direction to the option text in the current prompt. Yet the model achieves only 27.6% MCQ accuracy, near the 25% chance level (Figure 2). This shows that the model carries the direction signal internally but fails to bind it to the prompt-specific answer option.

Direction binding gap. Taken together, these results show that the bottleneck lies not in representing motion direction, but in converting that representation into the correct answer option. The direction signal remains accessible across the vision encoder, the projector, and the readout state, yet it is not reliably bound to the prompt-specific answer option [56, 57, 67]. We call this mismatch the *direction binding gap*. As shown in Figure 3, this gap persists across architectures and scales, indicating a shared structural limitation rather than a model-specific artifact. We further verify in Appendix D.3 that the same failure appears in open-ended direction-answer generation, confirming that the phenomenon is not an artifact of the MCQ format. This diagnosis motivates Section 4, where we test whether controlled motion direction instruction tuning can close the gap.



(a) OOD binding gap (b) Cross-domain geometry (c) Direction signal magnitude

Figure 5: **Shared orientation, weak magnitude. (a)** Instruction tuning closes the binding gap on Primitive-on-Syn, but the gap reopens on OOD domains; `DeltaDirect` narrows it across domains. **(b)** Direction concept vector orientations align across domains after instruction tuning, with late-layer cosine similarity exceeding 0.9. **(c)** Despite this alignment, concept-vector magnitude decreases with visual complexity, revealing the *magnitude deficit* behind the OOD binding gap.

4 What Instruction Tuning Solves and What It Does Not

Section 3 localizes directional motion blindness to answer-option binding: direction is linearly accessible inside the model, but the readout does not reliably link it to the correct answer option. We now test the most direct remedy: *motion direction instruction tuning*. We fine-tune on the simplest synthetic domain and ask whether the learned binding transfers to more complex visual conditions.

4.1 Setup: Training on MoDirect-Inst, Testing on MoDirect-SynBench

Training data. We fine-tune a Video-LLM on MoDirect-Inst, a synthetic instruction dataset built from the Primitive-on-Syn domain. MoDirect-Inst contains 100K training videos with diverse motion direction-related QA formats, including direction MCQ, open-ended direction, and static appearance questions (e.g., object color and shape, to preserve general visual recognition).

Evaluation data. We use MoDirect-SynBench as our testbed. MoDirect-SynBench includes Primitive-on-Syn, the source domain used to construct MoDirect-Inst, and three OOD domains: Primitive-on-Real, Cutout-on-Syn, and Cutout-on-Real. These OOD domains introduce natural scene backgrounds [92], realistic object cutouts [45], or both. This split allows us to distinguish source domain binding from domain-invariant direction understanding.

Training. We use LLaVA-Video-7B [89] as the backbone, keep the vision encoder and original LLM weights frozen, and update only the projector and LoRA [24] adapters in the LLM. We provide more training details in Appendix A.3.

4.2 Source Domain: Answer-Option Binding Emerges

We evaluate whether instruction tuning closes the direction binding gap on the held-out Primitive-on-Syn test set. We repeat the layer-wise probing analysis from Section 3. To diagnose answer-option binding, we use the *logit lens* [29, 55], which projects each intermediate readout token $\mathbf{h}^{\ell}$ through the LM head without training an external classifier. This measures whether the LM head can recover the correct answer letter at each layer. As shown in Figure 4, in the vanilla model, direction is highly decodable from the readout token (—), but its logit lens accuracy stays near chance (- -), matching its low MCQ accuracy of 27.6%. After instruction tuning, logit lens accuracy (- -) rises in the late layers and converges to direction probing accuracy (—) in the final layers. MCQ accuracy correspondingly increases to 99.7%. Thus, direction-focused instruction tuning establishes a source domain binding pathway from the internal direction signal to the correct answer option.

4.3 Out-of-Domain: The Binding Gap Reopens

We next ask whether this learned binding transfers to more complex visual conditions. As shown in Figure 5 (a), instruction tuning nearly closes the gap on the source domain Primitive-on-Syn test set (0.3 points), but the gap reopens on OOD domains, e.g., 12.1 points on Cutout-on-Real. Although direction remains decodable from the final readout token, MCQ accuracy drops as object and background complexity increase. This suggests that next-token supervision learns source domain answer binding, but does not produce a direction signal strong enough for reliable OOD readout.

4.4 Why Does the Binding Gap Reopen? Shared Orientation but Insufficient Magnitude

To understand the OOD failure, we analyze *motion direction concept vectors* using difference-in-means [1, 38, 52, 60, 68] at the readout state after instruction tuning. For each domain A, direction class d, and layer ℓ, we define $\mathbf{v}^{\ell}_{d,A} = \mathbb{E}[\mathbf{h}^{\ell} \mid y = d, A] - \mathbb{E}[\mathbf{h}^{\ell} \mid A]$. Here we compute the expectation over the MoDirect-SynBench samples. We decompose each vector into its unit orientation $\hat{\mathbf{v}}^{\ell}_{d,A} = \mathbf{v}^{\ell}_{d,A}/\|\mathbf{v}^{\ell}_{d,A}\|$ and magnitude $\|\mathbf{v}^{\ell}_{d,A}\|$. We use "orientation" here to distinguish the representation-space axis from the physical motion direction.

Orientation aligns across domains. To test whether the representation space orientation of each motion direction is shared across domains, we compare unit motion direction concept vectors across domains. For each layer ℓ, direction $d \in \{\text{left}, \text{right}, \text{up}, \text{down}\}$, and unordered pair of domains (A, B), we compute $\cos\left(\hat{\mathbf{v}}^{\ell}_{d,A}, \hat{\mathbf{v}}^{\ell}_{d,B}\right)$. Since MoDirect-SynBench has four domains, this gives $\binom{4}{2} = 6$ domain pairs and therefore $4 \times 6 = 24$ cosine similarity values per layer. The cosine similarity reported in Figure 5 (b) is the average over these values. After instruction tuning, this average rises sharply in the late layers, indicating that the model learns a shared motion direction concept vector orientation for each motion direction across domains, despite being trained on Primitive-on-Syn.

Magnitude drops under visual complexity. The aligned orientation alone is insufficient. As shown in Figure 5 (c), the same late layers exhibit a magnitude drop under increasing visual complexity; in particular, Cutout-on-Real shows a clear magnitude deficit relative to Primitive-on-Syn. This mirrors the OOD MCQ accuracy drop and suggests a *magnitude deficit*: although domains share a motion direction concept vector orientation for each motion direction, the corresponding signal magnitude is too weak in complex domains to support reliable answer-option binding.

Restoring magnitude recovers accuracy. As a diagnostic intervention, for each OOD sample with ground-truth motion direction d, we rescale $\mathbf{h}^{\ell}$ along the corresponding unit concept-vector orientation $\hat{\mathbf{v}}^{\ell}_{d,A}$ to match the average Primitive-on-Syn magnitude, leaving the orthogonal component unchanged. We observe that this intervention improves OOD MCQ accuracy, with the largest gain of 15.5 points on Cutout-on-Real. These results suggest that the OOD binding gap is not caused by a missing shared orientation across domains, but by *insufficient signal magnitude*.
This motivates a training objective (Section 5) that does not merely teach the answer token, but makes the projector output carry a stronger signed displacement signal before it enters the LLM.

5 `DeltaDirect`: Strengthening Motion Signals at the VL Interface

The preceding sections suggest a specific intervention point. Directional motion blindness is not caused by the complete absence of direction evidence: direction is linearly accessible from the vision encoder, projector output, and LLM hidden states (Section 3). However, this signal is not reliably bound to the correct answer option. Moreover, motion direction instruction tuning closes the binding gap on the source domain, but visual complexity weakens the magnitude of the shared concept-vector orientation, reopening the gap in out-of-domains (Section 4).

These observations lead to three design requirements. First, supervision should act at the projector output, where visual tokens enter the LLM, to encourage the visual representation passed to the LLM to preserve signed displacement information. Second, it should target motion-induced change rather than static appearance, so we supervise adjacent-frame feature deltas, which suppress time-invariant appearance content and emphasize temporal changes. Third, it should supervise signed displacement directly, rather than only through the answer word, so we use normalized 2-D motion vectors as explicit supervision.

To meet these requirements, we introduce `DeltaDirect`, a training-only auxiliary objective illustrated in Figure 6. It supervises adjacent-frame feature deltas at the projector output using synthetic 2-D motion vectors, alongside the standard next-token prediction objective. The auxiliary branch is discarded after training, leaving the test-time input format, token sequence, model architecture, and decoding procedure unchanged.

5.1 Motion-Induced Feature Deltas

`DeltaDirect` operates on the projector output $\mathbf{F} \in \mathbb{R}^{T \times N \times D}$. For each adjacent frame pair, we spatially pool the temporal feature difference to obtain a motion-change descriptor:

$$\delta_t = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{F}_{t+1}[n] - \mathbf{F}_t[n]) \in \mathbb{R}^D, \qquad t = 1, \ldots, T-1. \tag{1}$$

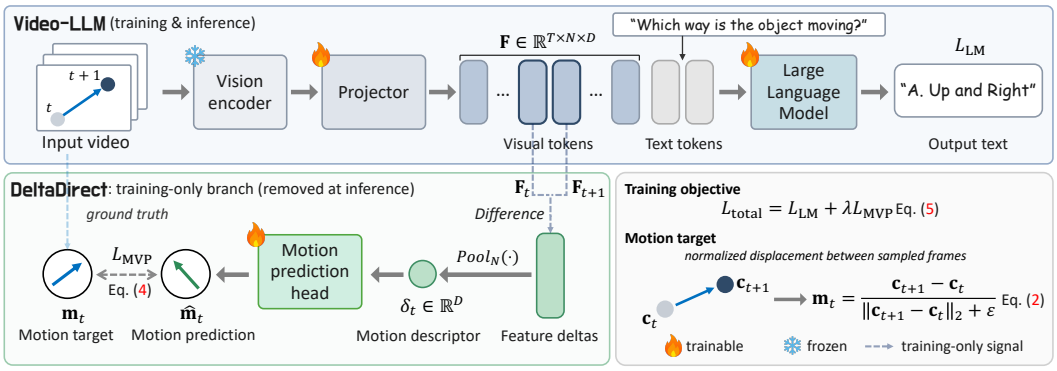


Figure 6: **`DeltaDirect` Overview.** Our diagnosis shows that visual complexity weakens the direction signal and reopens the binding gap. This finding motivates `DeltaDirect`, which makes the projector output carry a stronger signed displacement signal before it enters the LLM. We pool adjacent-frame feature deltas at the projector output into a motion descriptor δ_t, and a lightweight linear head predicts the normalized 2-D motion target $\mathbf{m}_t$ from it. This auxiliary branch is used only during training and removed at inference, introducing no additional inference-time overhead.

Feature differencing suppresses content that remains stable across adjacent frames and focuses the auxiliary branch on motion-induced change. We compute deltas at the projector output because these are the visual tokens directly passed to the LLM. This allows the auxiliary branch to emphasize motion-induced changes in the representation used for language readout, without modifying the vision encoder or introducing additional motion inputs at inference time.

5.2 Training-Only Motion Vector Prediction

Motion vector target. Because MODIRECT-INST is synthetic, the object center $\mathbf{c}_t \in \mathbb{R}^2$ is known at each frame. For each moving frame pair, we define the normalized 2-D motion target

$$\mathbf{m}_t = \frac{\mathbf{c}_{t+1} - \mathbf{c}_t}{\|\mathbf{c}_{t+1} - \mathbf{c}_t\|_2 + \varepsilon} \in \mathbb{R}^2. \tag{2}$$

This target captures the signed image-plane displacement direction while removing speed as a confounding factor. Rather than treating directions as discrete labels for an auxiliary classifier, the normalized vector target preserves the geometry of signed displacement: opposite motions have opposite targets, and different directions remain geometrically related in the same 2-D space.

Motion prediction head. A lightweight linear head predicts the motion vector from each pooled delta:

$$\hat{\mathbf{m}}_t = \mathbf{W}\,\delta_t + \mathbf{b}, \qquad \mathbf{W} \in \mathbb{R}^{2\times D},\ \mathbf{b} \in \mathbb{R}^2. \tag{3}$$

The head introduces only $2D+2$ trainable parameters and is used only during training.

MVP loss. We define the Motion Vector Prediction loss as

$$L_{\text{MVP}} = \frac{1}{T-1} \sum_{t=1}^{T-1} \|\hat{\mathbf{m}}_t - \mathbf{m}_t\|_2^2. \tag{4}$$

This objective does not ask the model to memorize a direction word. Instead, it directly shapes the projector output so that adjacent-frame changes preserve signed displacement cues before entering the LLM. We apply MVP supervision at the projector output by default, and compare it with intermediate-layer and readout-position supervision in Appendix E.

5.3 Training Objective and Inference

The final training objective combines standard next-token prediction with the MVP loss:

$$L_{\text{total}} = L_{\text{LM}} + \lambda_{\text{direct}} L_{\text{MVP}}, \tag{5}$$

where L_{LM} is the next-token cross-entropy on answer tokens and λ_{direct} controls the auxiliary loss weight. We set $\lambda_{\text{direct}} = 1.0$ by default. The MVP branch updates only the projector and motion prediction head; the LLM decoder is trained solely by L_{LM}. During inference, the MVP branch is removed, leaving the original input format, token sequence, and decoding procedure unchanged.

Table 1: **DeltaDirect improves motion direction understanding on both MoDirect-SynBench and MoDirect-RealBench.** We report Top-1 accuracy (%). Rows with ± report the mean and standard deviation over three seeds. Best results are in **bold** and second best are underlined.

Method	MoDirect-SynBench					MoDirect-RealBench				Overall Avg.
	P-Syn	C-Syn	P-Real	C-Real	Avg.	SSv2 [21]	KTH [63]	TOMATO [64]	Avg.	
Random Chance	25.0	25.0	25.0	25.0	25.0	50.0	50.0	19.0	39.7	31.3
Closed-source models										
GPT-4o [26]	46.6	37.8	48.0	40.7	43.3	51.0	52.3	34.5	45.9	44.4
Gemini 2.5 Flash [13]	58.1	43.4	61.7	50.6	53.5	65.8	69.4	25.6	53.6	53.5
Open-source models										
Video-LLaVA-7B [44]	28.1	27.8	25.5	27.2	27.2	49.2	49.5	14.6	37.8	31.7
LLaMA-VID-7B [43]	25.3	25.6	24.7	25.0	25.2	51.4	54.2	16.9	40.8	31.9
LLaVA-OneVision-7B [37]	28.3	23.4	29.9	29.3	27.7	52.2	50.3	26.8	43.1	34.3
Qwen2.5-VL-7B [5]	45.1	32.5	30.8	30.2	34.7	52.6	44.2	25.3	40.7	37.2
Qwen2.5-VL-72B [5]	69.5	51.3	57.4	51.1	57.3	66.8	62.0	34.2	54.3	56.0
Qwen3-VL-4B [4]	66.6	50.2	40.8	41.1	49.7	60.9	62.1	35.0	52.7	51.0
InternVL-2.5-4B [9]	31.9	31.8	30.4	31.4	31.4	66.1	50.7	26.1	47.6	38.3
VideoLLaMA3-7B [85]	56.4	46.6	50.0	48.3	50.3	56.5	52.2	19.9	42.9	47.1
LLaVA-Video-7B [89]	27.6	23.4	26.9	25.8	25.9	52.2	50.3	26.8	43.1	33.3
LLaVA-Video-72B [89]	36.5	28.3	28.1	29.3	30.6	50.3	56.0	27.3	44.5	36.5
Motion-specialized models										
Video-LaVIT-7B [28]	24.1	21.0	23.8	20.2	22.3	40.3	39.0	18.6	32.6	26.7
ReMoRa-7B [80]	25.4	22.7	26.2	25.7	25.0	50.8	50.3	25.6	42.2	32.4
Motion-o-7B [19]	46.4	36.4	33.8	34.3	37.7	63.4	56.3	28.3	49.3	42.7
MotionLLM-7B [8]	24.6	24.4	25.2	23.4	24.4	50.2	50.2	21.0	40.5	31.3
MotionSight-7B [16]	41.8	35.0	32.2	34.2	35.8	62.5	53.5	30.5	48.8	41.4
Ours										
LLaVA-Video-7B w/ MoDirect-Inst	**99.7**±0.2	81.1±0.4	75.3±0.7	61.0±0.5	79.3±0.4	72.0±0.6	67.9±1.6	35.8±0.5	58.6±0.5	70.4±0.5
LLaVA-Video-7B w/ DeltaDirect	99.6±0.1	85.5±0.9	86.1±0.9	72.5±0.9	85.9±0.7	**80.7**±0.9	**74.6**±0.9	**38.1**±0.7	**64.5**±0.5	76.7±0.3
Qwen2-0.5B Full-FT w/ MoDirect-Inst	99.5	97.3	62.5	51.5	77.7	59.0	55.2	20.6	44.9	63.7
Qwen2-0.5B Full-FT w/ DeltaDirect	**99.7**	**98.7**	**91.0**	**80.1**	**92.4**	77.8	69.4	21.7	56.3	**76.9**

6 Experimental Results

We evaluate whether DeltaDirect turns our diagnosis into practical gains. We focus on three questions: **(i)** Does it reduce the diagnosed binding gap? **(ii)** Does DeltaDirect improve synthetic and real-world motion direction understanding without real-world direction supervision? **(iii)** Does it preserve general video understanding?

6.1 Setup

Training. For each backbone, DeltaDirect uses the same training configuration as its MoDirect-Inst baseline (Section 4.1). We only adds the MVP loss to the next-token prediction loss. For full fine-tuning, we jointly optimize the vision encoder, projector, and language model using the same subset of approximately 354K samples from VideoChat2-Instruction [39]. For more details on the training configuration, please refer to Appendix A.3.

Evaluation. We evaluate direction recognition on MoDirect-SynBench and MoDirect-RealBench, and general video understanding on standard and fine-grained video benchmarks to assess whether improvements in direction recognition come at the cost of broader capabilities. For MoDirect-SynBench and MoDirect-RealBench, we uniformly sample 8 frames per video and pose a four-way multiple-choice question about the signed image-plane motion direction. Unless otherwise specified, inference uses LMMS-Eval [87] and Hugging Face [75] implementations. The full benchmark list and evaluation details are provided in Appendix A.1.

6.2 Improving Motion Direction without Sacrificing General Video Understanding

The direction binding gap narrows. In Figure 5 (a), instruction tuning on MoDirect-Inst leaves a widening binding gap (12.1 on Cutout-on-Real) under visual complexity. DeltaDirect reduces this gap to 3.7 points on Cutout-on-Real and narrows it on every OOD domain.

DeltaDirect substantially improves motion direction. Table 1 shows that MoDirect-Inst tuning nearly solves Primitive-on-Syn (99.7%) but transfers poorly to the hardest synthetic OOD domain, Cutout-on-Real (61.0%). DeltaDirect targets this gap, improving Cutout-on-Real to 72.5% and raising MoDirect-SynBench average accuracy from 79.3% to 85.9% over MoDirect-Inst alone. It also transfers to real videos without real-world tuning, improving MoDirect-RealBench Avg. from 58.6% to 64.5%. In full fine-tuning, the same trend

Table 2: **Where to apply supervision.**

Stage	MoDirect		MVBench
	SynBench	RealBench	
None	79.3	58.6	60.8
Vision encoder	80.4	59.3	**60.9**
Pre-projector	**86.1**	62.2	60.2
Post-projector	85.9	**64.5**	60.8
LLM visual tokens	70.0	58.3	57.4
Final readout	27.7	42.5	59.9

Table 3: **What motion signal to supervise.**

Supervision target	MoDirect		MVBench
	SynBench	RealBench	
MoDirect-Inst tuning only	79.3	58.6	**60.8**
Frame order	24.7	40.3	54.5
Concat. feature deltas	80.1	58.0	60.7
Delta equivariance	78.4	58.0	60.6
DeltaDirect	**85.9**	**64.5**	**60.8**

Table 4: **DeltaDirect preserves general video understanding.** We compare DeltaDirect with the base LLaVA-Video-7B on standard and fine-grained video benchmarks. EgoSchema uses the validation set; Vinoground reports the Group score.

Method	Standard video benchmarks					
	MVBench [39]	NExT-QA [76]	PcptTest [58]	EgoSchema [51]	TGIF-QA [27]	Avg.
LLaVA-Video-7B [89]	59.5	**82.7**	64.6	59.4	**80.7**	69.4
LLaVA-Video-7B w/ DeltaDirect	**60.8±0.1**	81.3±1.2	**65.8±0.1**	**60.8±1.1**	80.3±0.2	**69.8±0.3**
Qwen2-0.5B Full-FT w/ MoDirect-Inst	47.4	67.8	48.4	47.2	**82.7**	58.7
Qwen2-0.5B Full-FT w/ DeltaDirect	**48.9**	**69.3**	**49.2**	**47.6**	**82.7**	**59.5**

Method	Fine-grained video benchmarks				
	TempComp. [48]	Vinoground [86]	FAVOR [70]	MotionBench [23]	Avg.
LLaVA-Video-7B [89]	70.5	15.6	47.2	56.0	47.3
LLaVA-Video-7B w/ DeltaDirect	**73.4±0.2**	**15.9±0.6**	**48.3±0.4**	**56.3±1.1**	**48.5±0.3**
Qwen2-0.5B Full-FT w/ MoDirect-Inst	56.3	5.8	**32.0**	37.1	32.8
Qwen2-0.5B Full-FT w/ DeltaDirect	**57.2**	**6.8**	30.1	**39.1**	**33.3**

holds: DeltaDirect raises MoDirect-SynBench performance from 77.7% to 92.4% and MoDirect-RealBench Avg. from 44.9% to 56.3%.

DeltaDirect preserves general video understanding. Table 4 shows that although DeltaDirect targets motion direction, it preserves aggregate general video performance in Standard [27, 39, 51, 58, 76] video benchmarks and improves Fine-grained [23, 48, 70, 86] video benchmarks. For LLaVA-Video-7B, the Standard average stays on par (69.4 → 69.8) while the Fine-grained average increases from 47.3 to 48.5. In the full fine-tuning setup, DeltaDirect shows the same trend over MoDirect-Inst on both averages (58.7 → 59.5, 32.8 → 33.3). Thus, direction supervision does not trade off broader video capability.

6.3 Ablation Study

Where to apply supervision. In Table 2, we apply MVP supervision at various locations in a Video-LLM. Post-projector supervision performs best on average, while readout supervision performs poorly despite the binding gap being observed there. This supports strengthening motion signals at the vision-language interface, before the LLM transforms them for answer generation. We provide the full results in Table 13.

What motion signal to supervise. In Table 3, we compare DeltaDirect with various supervision targets including a direction-aware delta equivariance objective. DeltaDirect performs best, indicating that signed 2-D motion-vector supervision is most effective. We describe each supervision objective in Appendix E.1.8.

7 Conclusions

In this work, we identify *directional motion blindness*, a systematic failure of Video-LLMs to resolve signed image-plane motion direction. Through controlled diagnosis on MoDirect, we localize the failure to a *direction binding gap*: motion direction remains linearly accessible inside the model, but is not reliably connected to the correct verbal response. Instruction tuning closes this gap on the source domain, but visual complexity reopens it through a magnitude deficit in the shared direction representation. We address this with DeltaDirect, a training-only objective that predicts motion vectors from adjacent-frame projector-feature deltas, strengthening signed displacement cues without changing inference. Trained only with synthetic motion supervision, DeltaDirect improves synthetic and real-world direction understanding while preserving general video understanding. These results suggest that diagnosis-driven supervision can help close perception–language gaps in Video-LLMs.

Acknowledgments and Disclosure of Funding

This work was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (RS-2025-02216217, 50%) and (RS-2026-25616405, 25%). Additionally, it was supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government (MSIT) under grant RS-2024-00353131 (10%) and the Institute of Information & Communications Technology Planning & Evaluation (IITP)-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (IITP-2026-RS-2023-00259004, 10%). This research was also supported by the MSIT (Ministry of Science and ICT), Korea, under the National Program for Excellence in SW (2023-0-00042, 5%) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation) in 2026.

References

[1] Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *NeurIPS*, 2024. 2, 7

[2] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, 2023. 32

[3] Kyungho Bae, Jinhyung Kim, Sihaeng Lee, Soonyoung Lee, Gunhee Lee, and Jinwoo Choi. Mash-vlm: Mitigating action-scene hallucination in video-llms through disentangled spatial-temporal representations. In *CVPR*, 2025. 1, 3

[4] 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*, 2025. 3, 9, 30

[5] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 9, 30

[6] Richard T Born and David C Bradley. Structure and function of visual area mt. *Annu. Rev. Neurosci.*, 28 (1):157–189, 2005. 1

[7] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *CVPR*, 2024. 5

[8] Ling-Hao Chen, Shunlin Lu, Ailing Zeng, Hao Zhang, Benyou Wang, Ruimao Zhang, and Lei Zhang. Motionllm: Understanding human behaviors from human motions and videos. *TPAMI*, 2025. 9

[9] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, 2024. 9, 27, 30

[10] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *NeurIPS*, 2024. 5

[11] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024. 1

[12] Jihoon Chung, Tyler Zhu, Max Gonzalez Saez-Diez, Juan Carlos Niebles, Honglu Zhou, and Olga Russakovsky. Unifying specialized visual encoders for video language models. In *ICML*, 2025. 3

[13] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 1, 3, 9

[14] Daniel Cores, Michael Dorkenwald, Manuel Mucientes, Cees G. M. Snoek, and Yuki M. Asano. Tvbench: Redesigning video-language evaluation. *arXiv:2410.07752*, 2024. 3

[15] Nieves Crasto, Philippe Weinzaepfel, Karteek Alahari, and Cordelia Schmid. MARS: Motion-augmented RGB stream for action recognition. In *CVPR*, 2019. 3

[16] Yipeng Du, Tiehan Fan, Kepan Nan, Rui Xie, Penghao Zhou, Xiang Li, Jian Yang, Zhenheng Yang, and Ying Tai. Motionsight: Boosting fine-grained motion understanding in multimodal LLMs. In *ICLR*, 2026. 3, 9

[17] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *CVPR*, 2025. 1

[18] Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, Zhang Li, Guozhi Tang, Bin Shan, Chunhui Lin, Qi Liu, Binghong Wu, Hao Feng, Hao Liu, Can Huang, Jingqun Tang, Wei Chen, Lianwen Jin, Yuliang Liu, and Xiang Bai. Ocrbench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. In *NeurIPS*, 2025. 27

[19] Bishoy Galoaa, Shayda Moezzi, Xiangyu Bai, and Sarah Ostadabbas. Motion-o: Trajectory-grounded video reasoning. *arXiv preprint arXiv:2603.18856*, 2026. 9

[20] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014. 1

[21] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017. 3, 9, 28, 31, 32, 33, 34, 39

[22] Wes Gurnee and Max Tegmark. Language models represent space and time. In *ICLR*, 2024. 5

[23] Wenyi Hong, Yean Cheng, Zhuoyi Yang, Weihan Wang, Lefan Wang, Xiaotao Gu, Shiyu Huang, Yuxiao Dong, and Jie Tang. Motionbench: Benchmarking and improving fine-grained video motion understanding for vision language models. In *CVPR*, 2025. 1, 3, 10, 18, 23

[24] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022. 6

[25] Ziyuan Huang, Shiwei Zhang, Jianwen Jiang, Mingqian Tang, Rong Jin, and Marcelo H. Ang. Self-supervised motion learning from static images. In *CVPR*, 2021. 3

[26] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 3, 9, 29

[27] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *CVPR*, 2017. 10, 17, 22

[28] Yang Jin, Zhicheng Sun, Kun Xu, Liwei Chen, Hao Jiang, Quzhe Huang, Chengru Song, Yuliang Liu, Di Zhang, Yang Song, Kun Gai, and Yadong Mu. Video-LaVIT: Unified video-language pre-training with decoupled visual-motional tokenization. In *ICML*, 2024. 3, 9

[29] Minji Kim, Taekyung Kim, and Bohyung Han. Map the flow: Revealing hidden pathways of information in videoLLMs. In *ICLR*, 2026. 3, 4, 6, 19

[30] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022. 5

[31] Heeseung Kwon, Manjin Kim, Suha Kwak, and Minsu Cho. MotionSqueeze: Neural motion feature learning for video understanding. In *ECCV*, 2020. 3

[32] Dongho Lee, Jongseo Lee, and Jinwoo Choi. Cast: cross-attention in space and time for video action recognition. In *NeurIPS*, 2023. 3

[33] Jongseo Lee, Kyungho Bae, Kyle Min, Gyeong-Moon Park, and Jinwoo Choi. Essential: Episodic and semantic memory integration for video class-incremental learning. In *ICCV*, 2025. 32

[34] Jongseo Lee, Wooil Lee, Gyeong-Moon Park, Seong Tae Kim, and Jinwoo Choi. Disentangled concepts speak louder than words: Explainable video action recognition. In *NeurIPS*, 2025. 3

[35] Jongseo Lee, Wooil Lee, Gyeong-Moon Park, Seong Tae Kim, and Jinwoo Choi. Pcbear: Pose concept bottleneck for explainable action recognition. In *CVPRW*, 2025.

[36] Jongseo Lee, Joohyun Chang, Dongho Lee, and Jinwoo Choi. CA^{2}ST: Cross-attention in audio, space, and time for holistic video recognition. *TPAMI*, 48(3):2803–2819, 2026. 3

[37] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-onevision: Easy visual task transfer. *TMLR*, 2025. 1, 9, 18, 19, 26, 30, 34

[38] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *NeurIPS*, 2023. 2, 5, 7

[39] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *CVPR*, 2024. 1, 3, 4, 9, 10, 17, 19, 22, 26, 31, 32, 33

[40] Lei Li, Yuanxin Liu, Linli Yao, Peiyuan Zhang, Chenxin An, Lean Wang, Xu Sun, Lingpeng Kong, and Qi Liu. Temporal reasoning transfer from text to video. In *ICLR*, 2025. 1, 3

[41] Shicheng Li, Lei Li, Yi Liu, Shuhuai Ren, Yuanxin Liu, Rundong Gao, Xu Sun, and Lu Hou. Vitatecs: A diagnostic dataset for temporal concept understanding of video-language models. In *ECCV*, 2024. 3

[42] Yan Li, Bin Ji, Xintian Shi, Jianguo Zhang, Bin Kang, and Limin Wang. TEA: Temporal excitation and aggregation for action recognition. In *CVPR*, 2020. 3

[43] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *ECCV*, 2024. 1, 3, 9

[44] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *EMNLP*, 2024. 1, 3, 9, 30

[45] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2, 6

[46] Ruyang Liu, Chen Li, Haoran Tang, Yixiao Ge, Ying Shan, and Ge Li. St-llm: Large language models are effective temporal learners. In *ECCV*, 2024. 3

[47] Ruyang Liu, Shangkun Sun, Haoran Tang, Ge Li, and Wei Gao. Flow4Agent: Long-form video understanding via motion prior from optical flow. In *ICCV*, 2025. 3

[48] Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. Tempcompass: Do video llms really understand videos? In *ACL*, 2024. 1, 3, 10, 17, 22

[49] Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, Xiuyu Li, Haotian Tang, Yunhao Fang, Yukang Chen, Cheng-Yu Hsieh, De-An Huang, An-Chieh Cheng, Jinyi Hu, Sifei Liu, Ranjay Krishna, Pavlo Molchanov, Jan Kautz, Hongxu Yin, Song Han, and Yao Lu. NVILA: Efficient frontier visual language models. In *CVPR*, 2025. 3

[50] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *ACL*, 2024. 3, 26

[51] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *NeurIPS*, 2023. 1, 3, 10, 17, 22

[52] Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *First Conference on Language Modeling*, 2024. 2, 3, 7

[53] Ken Nakayama. Biological image motion processing: a review. *Vision Research*, 25(5):625–660, 1985. doi: 10.1016/0042-6989(85)90171-3. 1

[54] Hong Nguyen, Dung Tran, Hieu Hoang, Phong Nguyen, and Shrikanth Narayanan. MOOSE: Pay attention to temporal dynamics for video understanding via optical flows. *arXiv preprint arXiv:2506.01119*, 2025. 3

[55] nostalgebraist. Interpreting GPT: The logit lens. LessWrong, August 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. Accessed: 2026-02-22. 6, 24

[56] Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. Llms know more than they show: On the intrinsic representation of llm hallucinations. In *ICLR*, 2025. 2, 3, 5

[57] Yoonah Park, Haesung Pyun, and Yohan Jo. Bridging the knowledge-prediction gap in llms on multiple-choice questions. *arXiv preprint arXiv:2509.23782*, 2025. 5

[58] Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. In *NeurIPS*, 2023. 1, 3, 10, 17, 22

[59] Daiqing Qi, Dongliang Guo, Hanzhang Yuan, Handong Zhao, Mengxuan Hu, Lehan Yang, and Sheng Li. Improve temporal reasoning in multimodal large language models via video contrastive decoding. In *NeurIPS*, 2025. 1

[60] Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024. 2, 7

[61] Joshua Robinson, Christopher Michael Rytting, and David Wingate. Leveraging large language models for multiple choice question answering. In *ICLR*, 2023. 4

[62] Mohammadreza Salehi, Jae S Park, Tanush Yadav, Aditya Kusupati, Ranjay Krishna, Yejin Choi, Hannaneh Hajishirzi, and Ali Farhadi. Actionatlas: A videoqa benchmark for domain-specialized action recognition. In *NeurIPS*, 2024. 1

[63] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *ICPR*, 2004. 3, 9, 31, 32, 33, 34, 39

[64] Ziyao Shangguan, Chuhan Li, Yuxuan Ding, Yanan Zheng, Yilun Zhao, Tesca Fitzgerald, and Arman Cohan. Tomato: Assessing visual temporal reasoning capabilities in multimodal foundation models. In *ICLR*, 2025. 1, 3, 9, 22, 31, 32, 33, 34, 39

[65] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014. 3

[66] Jonathan C. Stroud, David A. Ross, Chen Sun, Jia Deng, and Rahul Sukthankar. D3D: Distilled 3D networks for video action recognition. In *WACV*, 2020. 3

[67] Yucheng Sun, Alessandro Stolfo, and Mrinmaya Sachan. Probing for arithmetic errors in language models. In *EMNLP*, 2025. 5

[68] Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Language models linearly represent sentiment. In *ICML*, 2024. 2, 7

[69] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, 2022. 32

[70] Chongjun Tu, Lin Zhang, Pengtao Chen, Peng Ye, Xianfang Zeng, Wei Cheng, Gang YU, and Tao Chen. Favor-bench: A comprehensive benchmark for fine-grained video motion understanding. In *NeurIPS*, 2025. 3, 10, 17, 23

[71] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. TDN: Temporal difference networks for efficient action recognition. In *CVPR*, 2021. 3

[72] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *CVPR*, 2023. 32

[73] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Zun Wang, Yansong Shi, et al. Internvideo2: Scaling foundation models for multimodal video understanding. In *ECCV*, 2024. 1, 3

[74] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022. 5

[75] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019. 9, 17

[76] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *CVPR*, 2021. 10, 17, 22

[77] Zihui Xue, Mi Luo, and Kristen Grauman. Seeing the arrow of time in large multimodal models. In *NeurIPS*, 2025. 3

[78] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024. 18, 19, 31

[79] Haosen Yang, Deng Huang, Bin Wen, Jiannan Wu, Hongxun Yao, Yi Jiang, Xiatian Zhu, and Zehuan Yuan. Self-supervised video representation learning with motion-aware masked autoencoders. *arXiv preprint arXiv:2210.04154*, 2022. 3

[80] Daichi Yashima, Shuhei Kurita, Yusuke Oda, and Komei Sugiura. ReMoRa: Multimodal large language model based on refined motion representation for long-video understanding. In *CVPR*, 2026. 9

[81] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mPLUG-Owl3: Towards long image-sequence understanding in multi-modal large language models. In *ICLR*, 2025. 30, 34

[82] Baiqiao Yin, Qineng Wang, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, et al. Spatial mental modeling from limited views. In *ICLR*, 2026. 5

[83] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *CVPR*, 2023. 18, 19, 31

[84] Yu-Wei Zhan, Xin Wang, Hong Chen, Tongtong Feng, Wei Feng, Ren Wang, Guangyao Li, Qing Li, and Wenwu Zhu. PhyVLLM: Physics-guided video language model with motion-appearance disentanglement. *arXiv preprint arXiv:2512.04532*, 2025. 3

[85] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025. 3, 9, 30

[86] Jianrui Zhang, Mu Cai, and Yong Jae Lee. Vinoground: Scrutinizing lmms over dense temporal reasoning with short videos. *arXiv preprint arXiv:2410.02763*, 2024. 3, 10, 17, 23

[87] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, et al. Lmms-eval: Reality check on the evaluation of large multimodal models. In *NAACL*, 2025. 9, 17

[88] Yuanhan Zhang, Bo Li, haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. Llava-next: A strong zero-shot video understanding model, 2024. URL https://llava-vl.github.io/blog/2024-04-30-llava-next-video/. 1, 30

[89] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun MA, Ziwei Liu, and Chunyuan Li. LLaVA-video: Video instruction tuning with synthetic data. *TMLR*, 2025. 3, 4, 6, 9, 10, 18, 26, 29, 30, 34

[90] Zijia Zhao, Yuqi Huo, Tongtian Yue, Longteng Guo, Haoyu Lu, Bingning Wang, Weipeng Chen, and Jing Liu. Efficient motion-aware video mllm. In *CVPR*, 2025. 3

[91] Zijia Zhao, Haoyu Lu, Yuqi Huo, Yifan Du, Tongtian Yue, Longteng Guo, Bingning Wang, Jing Liu, et al. Needle in a video haystack: A scalable synthetic evaluator for video mllms. In *ICLR*, 2025. 1

[92] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 40(6):1452–1464, 2017. 2, 6

[93] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024. 32

[94] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, et al. Mlvu: Benchmarking multi-task long video understanding. In *CVPR*, 2025. 1

[95] Shijie Zhou, Alexander Vilesov, Xuehai He, Ziyu Wan, Shuwang Zhang, Aditya Nagachandra, Di Chang, Dongdong Chen, Xin Eric Wang, and Achuta Kadambi. Vlm4d: Towards spatiotemporal awareness in vision language models. In *ICCV*, 2025. 3

[96] Yi Zhu, Zhenzhong Lan, Shawn Newsam, and Alexander G. Hauptmann. Hidden two-stream convolutional networks for action recognition. In *ACCV*, 2018. 3

Appendix

Table of Contents

A Implementation Details 17

A.1 Evaluation . . . 17

A.2 Model Card . . . 18

A.3 Training Details: LoRA Fine-tuning . . . 18

A.4 Training Details: Full Fine-tuning with `DeltaDirect` . . . 19

B Dataset and Benchmarks 20

B.1 MoDirect . . . 20

B.2 General Video Benchmarks . . . 22

C Analysis Details 24

C.1 Linear Probing Details . . . 24

C.2 Logit Lens Details . . . 24

D Additional Analysis 26

D.1 Video LLMs Instruction-tuning Dataset Analysis . . . 26

D.2 Input-side Scaffolds: Design and Full Results . . . 27

D.3 Motion Direction Binding Gap Beyond MCQ Format . . . 29

D.4 Motion Direction Binding Gap Across Video-LLMs . . . 29

E Additional Experimental Results 31

E.1 Ablation Study of `DeltaDirect` . . . 31

E.2 `DeltaDirect` Across Video-LLM Backbones . . . 34

E.3 Additional Analysis of `DeltaDirect` . . . 34

F Case Study 36

G Limitations 37

H Broader Impacts 37

A Implementation Details

This section describes the implementation details shared across the experiments in this paper. To facilitate reproducibility, we will publicly release the complete training and evaluation code, model checkpoints, and the full MoDirect upon publication.

Unless otherwise stated, all experiments use the following settings. Training is conducted on eight NVIDIA A6000 GPUs with 48 GB of memory each, and evaluation on eight NVIDIA RTX 4090 GPUs. We use `lmms-eval` [87] for all supported benchmarks. For benchmarks not supported by `lmms-eval`, we use the official evaluation scripts where available; otherwise, we implement the evaluation protocols directly. We uniformly sample $T=8$ frames per video during both training and inference.

A.1 Evaluation

In this section, we provide the full evaluation details. Unless otherwise specified, all inference runs are conducted using LMMS-Eval [87] and Hugging Face [75] implementations. We provide the sources and model cards of all evaluated models in Appendix A.2. We will publicly release MoDirect-SynBench, MoDirect-RealBench, and all trained model weights to support reproducibility.

A.1.1 MoDirect

We evaluate both MoDirect-SynBench and MoDirect-RealBench using the LMMS-Eval framework. For each video, we uniformly sample 8 frames and ask a four-way multiple-choice question about the signed image-plane motion direction. The model is allowed to generate up to 16 new tokens. We parse the generated response by first extracting an answer option, and then compare it with the ground-truth option to compute accuracy. All evaluations are run on eight NVIDIA RTX 4090 GPUs with a per-device batch size of 1.

A.1.2 Standard Video Benchmarks

MVBench. MVBench [39] evaluates general video understanding across diverse temporal reasoning tasks.

NExT-QA. NExT-QA [76] evaluates causal and temporal reasoning in video question answering. We sample 16 frames per video and report the resulting accuracy.

Perception Test. Perception Test [58] evaluates multimodal video understanding across perception-oriented reasoning tasks.

EgoSchema. EgoSchema [51] evaluates long-form egocentric video understanding through multiple-choice question answering. We use the publicly released subset, sample 32 frames per video, and report accuracy.

TGIF-QA. TGIF-QA [27] evaluates short-video question answering with emphasis on actions, transitions, and temporal dynamics.

A.1.3 Fine-grained Video Benchmarks

TempCompass. TempCompass [48] evaluates fine-grained temporal reasoning in videos, including event order, duration, and temporal relations. We exclude the captioning subtask.

VinoGround. VinoGround [86] evaluates compositional video-language understanding by testing whether models can distinguish subtle visual and temporal differences. We sample 16 frames per video and report only the group score, the strictest of the three official metrics (text, video, and group score).

FAVOR-Bench. FAVOR-Bench [70] evaluates fine-grained video understanding under temporally sensitive question-answering settings.

MotionBench. MotionBench [23] evaluates motion-centric video understanding with emphasis on fine-grained motion perception and reasoning.

A.2 Model Card

In this section, we provide the Hugging Face checkpoints for all evaluated models. All open-source models use greedy decoding with a maximum of 1024 new tokens. Following prior multiple-choice direction evaluation settings, we use a maximum of 16 new tokens because the answer is determined from the generated option token.

Table 5: Hugging Face checkpoints and GitHub repositories for all evaluated models.

Model	HF Checkpoint / GitHub
GPT	https://platform.openai.com/docs/models
Gemini	https://ai.google.dev/
Qwen2.5-VL-7B	Qwen/Qwen2.5-VL-7B-Instruct
Qwen2.5-VL-72B	Qwen/Qwen2.5-VL-72B-Instruct
Qwen3-VL-4B	Qwen/Qwen3-VL-4B-Instruct
VideoLLaMA3-2B	DAMO-NLP-SG/VideoLLaMA3-2B
VideoLLaMA3-7B	DAMO-NLP-SG/VideoLLaMA3-7B
Video-LLaVA-7B	LanguageBind/Video-LLaVA-7B
LLaVA-OneVision-7B	lmms-lab/llava-onevision-qwen2-7b-ov
LLaVA-Video-7B	lmms-lab/LLaVA-Video-7B-Qwen2
LLaVA-Video-72B	lmms-lab/LLaVA-Video-72B-Qwen2
LLaMA-VID-7B	YanweiLi/llama-vid-7b-full-224-video-fps-1
InternVL-2.5-4B	OpenGVLab/InternVL2_5-4B
mPLUG-Owl3-7B	https://github.com/x-plug/mplug-owl
Video-LaVIT-7B	rain1011/Video-LaVIT-v1
ReMoRa-7B	naisekizero/ReMoRa
Motion-o-7B	bishoygaloaa/motion-o
MotionLLM-7B	https://github.com/IDEA-Research/MotionLLM
MotionSight-7B	https://github.com/NJU-PCALab/MotionSight
DeltaDirect (LLaVA-Video-7B)	https://github.com/KHU-VLL/DeltaDirect

A.3 Training Details: LoRA Fine-tuning

We perform LoRA fine-tuning with the official LLaVA-NeXT repository[1] for LLaVA-Video-7B [89] and LLaVA-OneVision-7B [37]. LoRA fine-tuning takes approximately 7 hours.

A.3.1 DeltaDirect

We provide detailed training configurations for the DeltaDirect variant and the associated LoRA setup. We use LLaVA-Video-Qwen2 as the backbone model. This backbone consists of a Qwen2-7B [78] large language model, a SigLIP [83] vision encoder with patch size 14 and input resolution 384×384, and a two-layer GELU-based multimodal projector. The vision encoder is kept frozen throughout training.

Auxiliary Motion Direction Objective. DeltaDirect introduces an auxiliary supervision signal for motion direction. Specifically, we compute temporal differences between consecutive frame features, apply mean pooling, and feed the resulting representation into a linear prediction head. The auxiliary direction loss is jointly optimized with the main objective using a loss weight of $\lambda_{\text{direct}} = 1.0$.

Optimization Details. Training is performed for 1 epoch using AdamW with a learning rate of 1×10^{-5}, cosine learning-rate decay, a warmup ratio of 0.03, and zero weight decay. We enable mixed precision training with bf16 and tf32, as well as gradient checkpointing. The effective batch size is 144, obtained from 8 GPUs, a per-device batch size of 9, and gradient accumulation over 2 steps. Training uses DeepSpeed ZeRO-2.

[1] https://github.com/LLaVA-VL/LLaVA-NeXT

Fine-tuning and LoRA Setup. We fine-tune the multimodal projector and the large language model while keeping the vision encoder frozen. The projector and large language model use separate learning rates of 2×10^{-5} and 1×10^{-5}, respectively. LoRA is applied only to the large language model, with rank $r = 64$, scaling factor $\alpha = 128$, and dropout 0.05. The multimodal projector is fully fine-tuned.

Video Processing. Videos are processed using 8 frames with forced sampling. We apply bilinear spatial pooling with stride 2 to reduce the spatial resolution of visual features.

A.4 Training Details: Full Fine-tuning with `DeltaDirect`

Setup. We jointly optimize the vision encoder, projector, and language model of a Qwen2-0.5B [78] backbone paired with a SigLIP [83] vision encoder and a two-layer GELU projector, initialized from the Stage-0 LLaVA-OneVision-0.5B [37] checkpoint (∼893M trainable parameters). Full fine-tuning takes approximately 52 hours. For the training data, we subsample VideoChat2-Instruction (VideoChat2-IT) [39] following the data configuration of Kim et al. [29][2], resulting in approximately 354K training samples.

Objective. We combine the standard language-modeling loss with the auxiliary motion-direction loss of `DeltaDirect`. Since not all training samples contain motion-direction vector annotations, we apply the auxiliary loss only to annotated samples and mask it out otherwise. The final objective is

$$\mathscr{L} = \mathscr{L}_{\text{LM}} + \lambda_{\text{direct}}\, m\, \mathscr{L}_{\text{direct}},$$

where $m = 1$ for samples with motion-direction vector annotations and $m = 0$ otherwise.

Hyperparameters. We train for 2 epochs with AdamW at learning rate 2×10^{-5} for the language model and projector, and 2×10^{-6} for the vision encoder. We use a cosine schedule with warmup ratio 0.03, no weight decay, and an effective batch size of 96 (batch size 2 $\times$ gradient accumulation 6 across 8 GPUs). Each video is force-sampled to T=8 frames with spatial average pooling at stride 2. For more details, please refer to Table 6.

Table 6: Hyperparameters for full fine-tuning.

Hyperparameter	**Value**
Epochs	2
Total steps	9,438
LR (LLM, projector)	2×10^{-5}
LR (vision tower)	2×10^{-6}
LR schedule / warmup ratio	cosine / 0.03
Weight decay	0
Optimizer	AdamW (fused)
Per-device batch size	2
Gradient accumulation	6
GPUs	8
Effective batch size	96
Precision	bf16 + TF32
Gradient checkpointing	on
DeepSpeed	ZeRO Stage 2
Max sequence length	32,768
Frames T	16
Training samples	453,054
Synthetic Shape-Plain	98,399
VideoChat2-IT [39]	354,655

[2] https://github.com/byminji/map-the-flow/blob/main/docs/TRAIN.md

B Dataset and Benchmarks

B.1 MoDirect

In the main text, we introduced MoDirect as a dataset family for motion direction instruction tuning and evaluation. In this section, we provide detailed descriptions of its three components: MoDirect-Inst for instruction tuning, MoDirect-SynBench as a controlled synthetic benchmark for analysis and evaluation, and MoDirect-RealBench for real-world evaluation.

B.1.1 MoDirect-Inst

MoDirect-Inst is a synthetic instruction-tuning dataset constructed from the Primitive-on-Syn (P-Syn) domain, which serves as the source domain for training. The dataset contains 100,000 video–question–answer (QA) pairs with diverse motion-related question formats. Each video consists of 8 frames at a resolution of 384×384, depicting a single object with diverse appearance, motion patterns, and procedural background textures. Examples are shown in Figure 9.

QA composition and answer formats. The dataset contains both multiple-choice and open-ended QA formats, encouraging the model to learn both discrete decision boundaries and natural language grounding of motion concepts. The QA formats include direction prediction, motion description, trajectory understanding, object appearance recognition, location grounding, motion existence prediction, and rotation reasoning. A detailed breakdown of QA categories and their frequencies is provided in Table 7 (a). Direction-related QA pairs consist of both open-ended responses and multiple-choice questions. Specifically, we include both 5-way MCQ settings (4 cardinal directions + stationary) and 9-way MCQ settings (8 directions + stationary) to support both coarse and fine-grained motion understanding. Trajectory MCQs additionally require the model to reason about compound motion patterns, such as sequential direction changes in zigzag or roundtrip trajectories. The dataset also includes appearance-related questions about object and background properties, location grounding questions based on 3×3 spatial regions, and motion existence questions that determine whether an object is moving or stationary. Rotation-related QA pairs are generated only for circular-motion videos, where the model predicts clockwise (CW) or counterclockwise (CCW) motion directions using either multiple-choice or open-ended formats. Some QA categories naturally appear less frequently because they are only valid for specific motion types. For example, rotation QA pairs are generated only for circular-motion videos, while trajectory MCQs are constructed only from multi-stage motion patterns such as zigzag and roundtrip motions.

Dataset statistics. Table 7 further summarizes the distributions of motion patterns, motion directions, and procedural background textures. The dataset includes diverse motion patterns, including perturbed linear motion, zigzag trajectories, circular motion, roundtrip motion, and static scenes. Motion directions are approximately uniformly distributed across all 8 directions, reducing directional bias during instruction tuning. To increase visual diversity and reduce low-level shortcut learning, the dataset incorporates diverse procedural background textures, including gradients, checkerboards, stripes, noise-based textures, paper-like patterns, and vignette-style backgrounds. The final background distribution is not perfectly uniform because all generated samples are filtered using minimum foreground–background contrast constraints to ensure object visibility.

B.1.2 MoDirect-SynBench

MoDirect-SynBench is a controlled synthetic benchmark designed to isolate motion direction understanding. It follows a 2×2 factorial design over foreground type and background type: Primitive-on-Syn (P-Syn), Cutout-on-Syn (C-Syn), Primitive-on-Real (P-Real), and Cutout-on-Real (C-Real).

Dataset size and balance. Each domain contains 6,000 video samples, with 1,500 samples per direction (left, right, up, down). This balanced design ensures that performance differences cannot be attributed to class imbalance.

Video generation. Each video contains a single object moving with constant velocity along a linear trajectory. The starting position is sampled uniformly within a valid region, and the velocity is chosen such that (i) the object remains within the frame for the entire sequence and (ii) the total displacement exceeds a minimum threshold. This avoids degenerate cases such as negligible motion or early exit from the frame. The example is shown in Figure 10.

Table 7: MoDirect-Inst statistics.

(a) QA types

QA type	Description	Cnt
direction MCQ (9-way)	8 directions + stationary, 9 choices	17,973
direction MCQ (5-way)	4 directions + stationary, 5 choices	17,903
direction open	open-ended direction answer	13,953
description	motion description	9,885
appearance	object/background appearance description	10,393
location MCQ	3×3 region, 9 choices	6,461
location open	open-ended location answer	6,386
move or not MCQ	yes/no multiple choice	5,128
move or not open	yes/no open-ended answer	5,168
trajectory MCQ	compound direction, 4 choices	4,039
rotation MCQ	CW/CCW multiple choice	1,294
rotation open	CW/CCW open-ended answer	1,417
Total	–	100,000

(b) Motion

Type	Cnt
perturbed linear	29,942
zigzag	20,126
circular	19,830
roundtrip	14,972
static	15,130

(c) Direction

Dir	Cnt
right	8,268
left	8,087
up	8,122
down	8,176
top-right	8,068
top-left	8,067
bottom-right	8,083
bottom-left	8,169

(d) Background

Texture	Cnt
gradient	23,528
stripes	9,840
solid	9,701
checker	9,690
noise	9,407
gaussian noise	9,407
blobs	9,096
speckle	8,953
paper	5,917
vignette	4,406

Task formulation. We adopt a four-way MCQ format, where the answer options correspond to four signed directions. The order of answer options is randomized for each sample, so that the correct answer letter is not a fixed alias of the motion direction. This design directly tests whether the model can bind the perceived direction to the prompt-specific answer option.

Question format. Each video is paired with a direction question: *"From the viewer's perspective, in which direction is the object moving in this video?"* We use a 4-way multiple-choice setting with candidates {Left, Right, Up, Down}. Candidate order is randomly shuffled to eliminate answer-position bias.

B.1.3 MoDirect-RealBench

MoDirect-RealBench is a real-world motion direction benchmark constructed by curating samples from existing video datasets, including Something-Something-V2 (SSv2), KTH, and TOMATO. For SSv2 and KTH, we reformulate each video into a motion direction multiple-choice question that asks: *"From the viewer's perspective, in which direction is the something or person moving in this video?"*

Something-Something-V2. We select all samples from four action categories that explicitly encode left–right motion: *"Pulling something from left to right," "Pulling something from right to left," "Pushing something from left to right,"* and *"Pushing something from right to left."* Each sample is reformulated into a binary multiple-choice question, resulting in a chance performance of 50%. This filtering results in 722 samples. We include all available samples from the selected categories, without additional curation or filtering. The example is shown in Figure 11 (a).

KTH. We use samples from the *walking*, *jogging*, and *running* classes. Motion direction (left vs. right) is determined from human skeleton trajectories by measuring the dominant horizontal displacement of key joints. This results in 899 samples. We include all videos from the selected action classes, without manual selection. Each sample is cast as a binary MCQ (left vs. right), yielding a chance accuracy of 50%. The example is shown in Figure 11 (b).

TOMATO. We include 403 samples from the TOMATO [64] benchmark, which evaluates visual temporal reasoning across three demonstration categories: *human-centric*, *real-world object-centric*, and *simulated* scenarios. Human-centric videos involve human body movements or interactions, such as hand gestures or full-body actions. Object-centric videos focus on the temporal dynamics of real-world objects in natural scenes. Simulated videos depict simplified synthetic environments with abstract moving entities designed to isolate temporal reasoning from appearance complexity. We use the original annotations and convert them into a unified multiple-choice format. Each sample is cast as an MCQ with its original answer options; since the number of options varies across questions, the expected chance accuracy is 19.0%. Examples are shown in Figure 11 (c).

Evaluation format. All datasets are converted into a multiple-choice format to ensure consistent evaluation. This allows direct comparison with synthetic benchmarks while testing real-world generalization.

B.2 General Video Benchmarks

B.2.1 Standard

MVBench. MVBench [39] is a multi-modal video understanding benchmark for evaluating temporal comprehension in Multi-modal Large Language Models (MLLMs). It consists of 20 video tasks constructed via a static-to-dynamic transformation and evaluates models using automatically generated multiple-choice questions based on video annotations.

NExT-QA. NExT-QA [76] is a video question answering benchmark designed to evaluate deeper video understanding beyond description, focusing on causal reasoning, temporal action reasoning, and scene comprehension. It provides both multiple-choice and open-ended QA tasks and reveals that existing models remain weak in causal and temporal reasoning despite strong performance on shallow scene description.

Perception Test. Perception Test [58] is a multimodal video benchmark designed to evaluate perception and reasoning abilities of pre-trained multimodal models across video, audio, and text modalities. It focuses on skills such as memory, abstraction, physics, and semantics, and provides densely annotated real-world videos with multiple-choice and grounded video question answering for comprehensive evaluation.

EgoSchema. EgoSchema [51] is a long-form video question answering benchmark designed to evaluate long-term video understanding in vision–language models. It contains over 5,000 human-curated multiple-choice questions derived from Ego4D videos, requiring reasoning over approximately three-minute-long clips and demonstrating the difficulty of long temporal reasoning for current models. In this work, we use the subset of EgoSchema included in our evaluation suite and report multiple-choice accuracy following the standard evaluation protocol.

TGIF-QA. TGIF-QA [27] is a video question answering benchmark that extends visual question answering from images to videos, focusing on spatio-temporal reasoning. It introduces three video-specific QA tasks and provides a large-scale dataset designed to evaluate models' ability to reason over temporal dynamics in videos.

B.2.2 Fine-grained

TempCompass. TempCompass [48] is a benchmark designed to evaluate temporal perception abilities of Video LLMs across diverse temporal aspects and task formats. It introduces videos that differ only in specific temporal properties and provides multiple evaluation tasks to measure nuanced temporal understanding beyond single-frame bias.

VinoGround. VinoGround [86] is a benchmark for evaluating temporal reasoning in short videos using temporal counterfactual video–caption pairs. It consists of 1,000 natural video-caption examples designed to test models' ability to distinguish temporal differences between actions and object transformations.

FAVOR-Bench. FAVOR-Bench [70] is a benchmark for evaluating fine-grained motion understanding in Multimodal Large Language Models (MLLMs). It provides manually annotated videos with both multiple-choice and open-ended tasks to assess detailed temporal motion comprehension.

MotionBench. MotionBench [23] is a benchmark designed to evaluate fine-grained motion comprehension in video understanding models. It assesses motion-level perception through diverse motion-oriented question types collected from varied real-world video sources.

C Analysis Details

C.1 Linear Probing Details

This appendix specifies the linear probing protocol used in Section 3, Section 4, and the cross-backbone direction binding gap analysis in Appendix D.4. The goal of this protocol is to measure whether motion-direction information is linearly accessible at different stages of the Video-LLM pipeline, rather than to introduce an additional high-capacity classifier. We therefore use the same lightweight probe design and optimization setting across all stages and backbones.

Probe protocol. For each stage, we freeze the underlying Video-LLM and train a single linear classifier on top of the extracted features. The probe is optimized with cross-entropy loss using standardized inputs, where the mean and standard deviation are computed only from the training split to avoid test-set leakage. Because the probe contains no hidden layers, its performance reflects the linear decodability of the underlying representation rather than the expressive power of the probe itself. Unless otherwise specified, we use this probing protocol for all linear-probing results reported in the main paper and appendix. We use the same hyperparameters for all stages and all backbones, summarized in Table 8.

Table 8: Linear probing hyperparameters used throughout the paper.

Hyperparameter	Value
Probe architecture	Single linear layer + softmax
Loss	Cross-entropy
Optimizer	AdamW
Learning rate	10^{-3}
Weight decay	10^{-2}
Batch size	64
Epochs	50
Train–test split	70/30 stratified

Pipeline stages. We probe four stages along the Video-LLM pipeline. First, we probe the visual encoder output $\mathbf{V} \in \mathbb{R}^{T\times M\times D_v}$. Second, we probe the projector output $\mathbf{F} \in \mathbb{R}^{T\times N\times D}$, which maps visual features into the language-model embedding space. Third, we probe the LLM visual-token hidden state $\mathbf{z}_t^\ell \in \mathbb{R}^D$ at frame t and layer ℓ. Following Section 3.1, $\mathbf{z}_t^\ell$ denotes the spatially pooled hidden state over the visual-token positions corresponding to frame t. Finally, we probe the readout token hidden state $\mathbf{h}^\ell \in \mathbb{R}^D$ at layer ℓ. The readout token is the last token in the prefill and is directly used by the LM head for next-token prediction. In our MCQ template, this position immediately precedes the answer character, making it the relevant representation for analyzing answer-option binding.

C.2 Logit Lens Details

While linear probing verifies whether motion direction is linearly decodable from an internal representation, it does not test whether that representation is already aligned with the LM head for answer selection. To measure this alignment, we use the logit lens at the readout token position.

Formulation. The logit lens [55] feeds an intermediate hidden state directly to the frozen LM head, without any additional training, to inspect what the model would predict if decoding stopped at that layer. For layer $\ell \in \{1,\dots,L\}$, we compute the logit-lens letter prediction at the readout token position as

$$\hat{y}^\ell_{\text{lens}} = \arg\max_{c\in\{A,B,C,D\}} \Big(\mathbf{W}_u \cdot \text{Norm}(\mathbf{h}^\ell)\Big)_{t_c}, \tag{6}$$

where $\mathbf{h}^\ell \in \mathbb{R}^D$ is the readout token hidden state at layer ℓ, $\text{Norm}(\cdot)$ is the final pre-unembedding normalization layer of the LLM, $\mathbf{W}_u \in \mathbb{R}^{V\times D}$ is the frozen LM head, and t_c is the vocabulary index of answer-letter token c.

Protocol. Unless otherwise specified, we use the same prompt format, video preprocessing, answer-token candidates, and held-out evaluation split as in the corresponding MCQ evaluation. We apply the frozen final normalization layer and LM head to each intermediate readout-token hidden state and evaluate whether the highest-scoring answer-letter token matches the ground-truth option. No additional parameters are trained for this analysis.

Interpretation. Unlike a learned linear probe, the logit lens evaluates whether the model's own LM head can already read out the correct answer from a given layer. Thus, a gap between linear-probe accuracy and logit-lens accuracy indicates that motion-direction information is present in the representation but is not yet aligned with the answer-selection space.

D Additional Analysis

D.1 Video LLMs Instruction-tuning Dataset Analysis

This section provides details of the instruction-data analysis summarized in Section 3.2. We describe the direction supervision estimation pipeline, its validation via human annotation, and the resulting dataset statistics.

Table 9: **Motion direction supervision is scarce in video instruction-tuning data.** Proportion of QA pairs identified by two stages of the Direction Supervision Estimation pipeline. **Keyword Matched** reports the number of QA pairs that contain explicit directional keywords and the corresponding total number of QA pairs in each dataset. **Semantically Direction-dependent QA** reports the subset of keyword-matched QA pairs that require motion direction understanding after semantic filtering, where the total corresponds to the size of the keyword-matched subset. **Final Ratio** denotes the proportion of semantically direction-dependent QA pairs relative to the total number of QA pairs in the dataset.

Dataset	Keyword Matched		Semantically Direction-dependent QA		Final Ratio
	Matched	Total	Directional QA	Matched	
VideoChat2-IT [39]	143,759	957,999	18,134	143,759	1.89%
VideoChatGPT100K [50]	23,156	99,814	989	23,156	0.99%
LLaVA-OneVision (video) [37]	47,240	487,632	3,766	47,240	0.77%
LLaVA-Video-178K [89]	802,700	5,699,294	51,776	802,700	0.91%

D.1.1 Motion Direction Supervision Estimation Pipeline

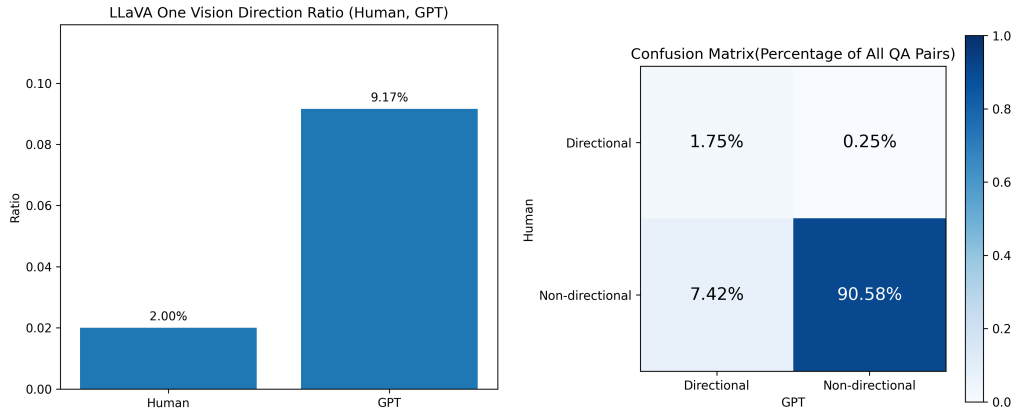
Keyword pre-filtering for motion direction. The instruction data contains a diverse set of QA pairs. To efficiently identify candidate samples, we first apply a lightweight pre-filtering step based on keyword matching. We define a keyword set $\mathscr{K} = \{\text{left, right, up, down}\}$, which captures the primary horizontal and vertical directions. A QA pair is retained if any keyword in $\mathscr{K}$ appears as a case-insensitive whole-word match in either the question or the ground-truth answer. This step intentionally retains a broad set of candidate QA pairs, including cases where directional terms appear without requiring motion direction reasoning. For example, words such as "right" may refer to correctness rather than spatial direction, and phrases like "close up" may include directional terms without implying object motion. These cases cannot be reliably disambiguated by keyword matching alone and are further filtered in the subsequent semantic classification stage.

Semantic filtering for motion direction. We apply a semantic filtering step to determine whether a QA pair truly requires motion direction understanding. We formulate this as a binary classification problem, where each QA pair is labeled as *directional* or *non-directional*. The classifier takes both the question and the ground-truth answer as input, and predicts whether motion direction reasoning is required to answer the question correctly. We implement this classifier using a GPT-4o model via API. This step filters out semantically irrelevant cases that pass the keyword pre-filtering, yielding a refined subset of direction-dependent QA pairs.

D.1.2 Reliability of Semantic Filtering

Human verification. Since GPT-based semantic filtering may cause classification errors, we validate its reliability against human annotations. We randomly sample 5,000 QA pairs from the LLaVA-OneVision instruction dataset [37] and label each pair as *directional* or *non-directional*. Annotators follow the same labeling criteria used for the GPT-based classifier and are not exposed to its predictions. The resulting annotations are used to analyze agreement and error patterns between GPT and human judgments.

Error analysis. Using these annotations, we analyze the agreement between GPT predictions and human labels, as shown in Figure 7. Figure 7 (a) shows that GPT overestimates the proportion of direction-dependent QA pairs. Figure 7 (b) further shows that this discrepancy is primarily driven by false positives. Specifically, the classifier incorrectly labels non-directional QA pairs as directional, while missing very few true directional cases. This results in high recall (87.5%) but low



(a) (b)

Figure 7: **LLM-based semantic classification tends to over-predict direction.** (a) Comparison of overall direction ratios. While human annotations show that only a small fraction of QA pairs require direction understanding, the LLM predicts direction at a substantially higher rate, further supporting the over-prediction tendency. (b) Confusion matrix comparing LLM predictions with human annotations. The model more often labels non-direction QA pairs as direction than it misses true direction cases, indicating a tendency to over-predict direction.

precision (19.1%), indicating that the classifier rarely misses truly directional QA pairs while substantially over-predicting directional ones. Since the analysis is based on sampled data, we account for sampling variability by computing a 95% confidence interval ($9.17\% \pm 2.8\%$). Even under this uncertainty, the estimated proportion remains substantially higher than the human-annotated rate, indicating consistent overestimation. Because the classifier tends to include non-directional cases rather than exclude true directional ones, we interpret this estimate as an upper bound on the true amount of motion direction supervision.

D.1.3 Results

We quantify the presence of motion direction supervision in video instruction-tuning datasets using the proposed two-stage pipeline. As shown in Table 9, keyword-based pre-filtering retains a substantial portion of QA pairs (9.7–23.2%), but this proportion drops sharply after semantic filtering. Across datasets, only 0.77–1.89% of QA pairs require motion direction understanding, remaining consistently around 1% despite differences in dataset size and composition. Given that GPT-based semantic filtering tends to overestimate due to false positives (Appendix D.1.2), these values should be interpreted as upper bounds. Even under this conservative interpretation, motion direction supervision remains limited across all datasets.

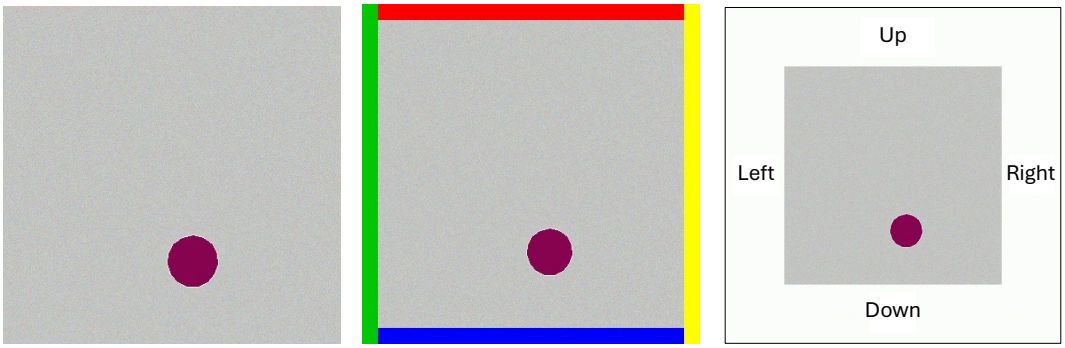
D.2 Input-side Scaffolds: Design and Full Results

In this section, we provide a detailed description of the prompting design and full results underlying this finding. We study input-side scaffolds from two perspectives. On the visual side, we augment video frames with explicit directional cues. On the text side, we modify the prompt to guide step-by-step reasoning about object motion. We consider multiple visual and text scaffolds and evaluate all visual–text combinations across synthetic and real-world benchmarks. We describe the visual and text prompting strategies and then report the full results for all scaffold configurations.

D.2.1 Visual Prompting

Visual prompting introduces explicit directional cues into the video frames, providing additional spatial signals that can guide direction inference [9, 18]. We consider three visual conditions: *(i) Plain* uses the original video without any additional visual cues. *(ii) Color edge* assign distinct colors to the four image borders, allowing the model to relate the object's position to the corresponding boundary cues. *(iii) Text edge* explicitly annotate each border with directional words (e.g., "Up",

"Down", "Left", "Right"), providing a more explicit association between position and direction. Examples of all conditions are shown in Figure 8.



(a) Plain (b) Color edge (c) Text edge

Figure 8: **Visual prompting examples.** We compare the original input with two visual cue variants designed to make directional information more explicit. (a) *Plain* shows the unmodified video frame without any additional visual cue. (b) *Color edge* marks the four image borders with distinct colors. (c) *Text edge* annotates the borders with directional words.

D.2.2 Text Prompting

Text prompting modifies the input question to elicit structured reasoning about motion direction. Rather than determining direction from a single query, these prompts guide the model to explicitly reason over object positions and their temporal changes.

We consider three variants: *(i) Default prompting* asks the direction question directly without additional guidance. *(ii) Temporal prompting* decomposes the task into intermediate steps, asking the model to describe the object and its positions at the beginning and end of the video before determining direction. *(iii) Grid-based prompting* follows the same step-by-step structure as temporal prompting, but additionally introduces an explicit coordinate system over the image, requiring the model to estimate start and end positions within the grid and determine direction from coordinate change. Examples of all conditions are shown in Figure 12, Figure 13, Figure 14.

D.2.3 Full Results

Table 10: **Prompting does not reliably recover motion direction reasoning.** Accuracy under combinations of visual and text prompting.

Visual prompting	Text prompting	P-Syn	SSv2 [21]
Random chance		25.0	50.0
Plain	Default	27.6	52.2
	Temporal	29.0	**57.1**
	Grid	28.2	53.6
Color edge	Default	25.5	48.6
	Temporal	29.4	52.9
	Grid	23.0	49.2
Text edge	Default	32.0	49.9
	Temporal	**34.7**	54.2
	Grid	29.1	50.1

Table 10 reports accuracy under all scaffold combinations on Primitive-on-Syn and SSv2 [21]. On Primitive-on-Syn, the best-performing configuration, *Temporal + Text edge*, reaches 34.7%, only a modest improvement over the raw baseline (27.6%).

On SSv2, the best result is achieved by *Temporal prompting* at 57.1%, but this gain is not consistently preserved across other combinations. Visual prompting alone (e.g., Color edge: 48.6%, Text edge:

49.9%) generally underperforms the raw baseline (52.2%), and combining visual and text prompts does not yield reliable improvements.

Overall, the results show that neither visual cues nor structured text prompting consistently recovers motion direction reasoning, and in many cases, additional prompting introduces noise that degrades performance.

D.3 Motion Direction Binding Gap Beyond MCQ Format

In this section, we verify that the direction binding gap is not an artifact of the letter-MCQ format. We evaluate vanilla LLaVA-Video [89] on Primitive-on-Syn under two additional answer formats that preserve the direction question but change the required output: *direction-word MCQ*, where the four options are the direction words {left, right, up, down} rather than the letters A/B/C/D, and *open-ended generation*, where the model generates a free-text answer that we score with GPT-4o [26]. For each format, we measure the final-readout direction probe accuracy and the corresponding task accuracy.

Table 11: **The direction binding gap persists beyond the MCQ format.** Final-readout direction probe accuracy and task accuracy on vanilla LLaVA-Video, evaluated on Primitive-on-Syn under three answer formats: letter MCQ (default), direction-word MCQ, and open-ended generation. The final-readout direction probe stays above 90% in all three formats, while task accuracy remains low, leaving a binding gap of more than 60 pp in every format. Open-ended generation is scored with GPT-4o.

Answer format	Probe (%)	Acc. (%)	Gap (pp)
Letter MCQ	95.3	27.6	67.7
Direction-word MCQ	92.3	27.8	64.5
Open-ended generation	93.3	19.7	73.6

The binding gap persists across answer formats. Table 11 reports the results. The final-readout direction probe stays above 90% in all three formats, indicating that motion direction is linearly accessible regardless of the required output format. Yet the task accuracy remains low across the board, ranging from 19.7% for open-ended generation to 27.8% for direction-word MCQ. The direction binding gap therefore exceeds 60 pp in every format we tested.

The binding gap is not an MCQ artifact. The persistence of the gap under both direction-word MCQ and open-ended generation rules out the letter-MCQ format as the source of directional motion blindness. The bottleneck lies in converting the linearly accessible direction signal into the correct verbal answer, regardless of whether that answer is a letter, a word, or a free-text response.

D.4 Motion Direction Binding Gap Across Video-LLMs

Setup. We test whether the direction binding gap observed in LLaVA-Video (Section 3.4) generalizes to other Video-LLMs. For each model, we measure two quantities on the Primitive-on-Syn domain of MoDirect-SynBench: (i) direction probing accuracy at the final readout token, following Appendix C.1, and (ii) MCQ accuracy from greedy generation, parsed as the first emitted letter among `{A, B, C, D}`. We define the binding gap as the difference between the two, following Section 3.4. We evaluate every model on 6,000 balanced samples (1,500 per direction, four directions).

Models. We evaluate nine Video-LLMs spanning different vision encoders, projectors, LLM families (Qwen2 / Qwen2.5 / Qwen3 / Vicuna), and parameter scales (2B-7B). We list the full model set with citations in Table 12.

The binding gap is consistent across Video-LLMs. Table 12 reports direction probing and MCQ accuracy on the nine evaluated Video-LLMs. Direction probing is high in every model (75.3 %-97.8 %), confirming that motion direction is linearly decodable from the readout token across architectures. MCQ accuracy, however, stays near the 25 % chance baseline in seven of the nine models; only Qwen2.5-VL (45.1 %) and Qwen3-VL (66.6 %) achieve meaningful direction MCQ out of the

Table 12: **The direction binding gap is universal across Video-LLMs.** Direction probing accuracy and MCQ accuracy at the final readout state, evaluated on Primitive-on-Syn in MoDirectSynBench (6,000 balanced samples). Direction is linearly decodable in every backbone, yet MCQ accuracy stays near the 25 % chance baseline in seven of the nine models. Even Qwen3-VL, the strongest MCQ baseline, retains a 31.2 pp gap.

Method	Direction Probe (%)	MCQ Acc. (%)	Gap (pp)
LLaVA-Video-7B (Qwen2) [89]	95.3	27.6	67.7
mPLUG-Owl3-7B [81]	90.5	25.6	64.9
LLaVA-OneVision-7B (Qwen2) [37]	90.8	28.3	62.5
VideoLLaMA3-2B (Qwen2) [85]	93.6	31.2	62.4
InternVL2.5-4B (Qwen2.5) [9]	91.1	31.9	59.2
Qwen2.5-VL-7B [5]	95.8	45.1	50.7
LLaVA-NeXT-Video-7B (Vicuna) [88]	75.3	25.8	49.5
Video-LLaVA-7B (Vicuna) [44]	77.1	28.1	49.0
Qwen3-VL-4B [4]	**97.8**	**66.6**	**31.2**

box. The resulting binding gap exceeds 30 percentage points in every Video-LLM we evaluate. Even Qwen3-VL — the most MCQ-aligned vanilla model — exhibits a 31.2 pp gap (probe 97.8 % vs. MCQ 66.6 %): the binding gap persists even when downstream MCQ capability is high. The gap is therefore not specific to a particular vision encoder, projector, or LLM family, but reflects a shared failure mode across the Video-LLMs we evaluate.

E Additional Experimental Results

E.1 Ablation Study of `DeltaDirect`

This section provides ablation studies for `DeltaDirect`. Our goal is to verify that the improvement does not come from additional training alone, but from the proposed motion direction supervision and its feature construction. We therefore compare alternative variants that modify the auxiliary objective, the motion feature used for supervision, and the training configuration while keeping the remaining components fixed.

E.1.1 Setup

We vary one design axis at a time around an anchor configuration and report the impact on direction performance. The anchor configuration follows Section 5: LLaVA-Video (Qwen2 [78] LLM, SigLIP [83] vision encoder), $T = 8$ frames, LoRA rank $r = 64$ with $\alpha = 128$ on the LLM, a fully fine-tuned projector, $\lambda_{\text{MVP}} = 1.0$, mean spatial pooling, a linear motion head, the subtraction temporal operator, and MSE loss against unit-vector targets.

Evaluation suite. We report seven direction tasks and one general-video task. The seven direction tasks are: 4-way directional MCQ on the four MoDirect-SynBench domains (chance 25%, 1,500 samples per direction), and the three MoDirect-RealBench sub-tasks KTH, SSv2, and TOMATO. For compactness, we abbreviate the four MoDirect-SynBench domains as P-Syn, C-Syn, P-Real, and C-Real, where P/C denote Primitive/Cutout foregrounds and Syn/Real denote synthetic/real backgrounds. *Dir avg.* denotes the unweighted mean over these seven tasks. MVBench (mean over 20 sub-tasks) monitors general video understanding.

E.1.2 Supervision Location

Table 13: **Ablation: location of MVP supervision (full breakdown).** During MoDirect-Inst instruction tuning, we apply MVP supervision at different points along the Video-LLM pipeline and report Top-1 accuracy (%) on the four MoDirect-SynBench domains (P-Syn, C-Syn, P-Real, C-Real; chance 25%) and three MoDirect-RealBench sub-tasks (SSv2, KTH, TOMATO). Avg. Direction is the unweighted mean over MoDirect-SynBench and MoDirect-RealBench; MVBench [39] monitors general video understanding. ℓ_v and ℓ denote the layer index of the vision encoder [83] ($L_v = 27$) and LLM [78] ($L = 28$), respectively. Best results per column are in **bold**.

Stage	Feat.	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
		P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
None	–	99.7	81.1	75.3	61.0	72.0	67.9	35.8	70.4	60.8
Vision encoder	$\mathbf{V}^{\ell_v=7}$	**99.8**	81.6	76.6	62.3	72.6	66.7	35.5	70.7	60.6
	$\mathbf{V}^{\ell_v=14}$	**99.8**	82.3	76.7	62.7	73.4	69.2	35.2	71.3	**60.9**
	$\mathbf{V}^{\ell_v=21}$	99.7	81.6	76.3	62.1	73.1	67.3	35.5	70.8	60.7
Pre-projector	$\mathbf{V}^{L_v}$	**99.8**	**86.2**	**86.9**	71.4	78.1	71.6	37.0	75.9	60.2
Post-projector (Ours)	$\mathbf{F}$	99.6	85.5	86.1	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	60.8
LLM visual tokens	$\mathbf{z}^{\ell=7}$	27.2	27.1	27.6	26.9	50.6	42.0	20.6	31.7	56.6
	$\mathbf{z}^{\ell=14}$	83.8	72.9	63.1	60.0	73.8	68.6	32.5	65.0	57.4
	$\mathbf{z}^{\ell=21}$	79.0	62.1	63.6	51.5	66.9	71.4	26.6	60.2	57.5
Final readout	$\mathbf{h}^{L}$	28.5	26.7	30.1	25.4	56.8	50.2	20.5	34.0	59.9
Two-tap	$\mathbf{F},\mathbf{z}^{L}$	63.6	38.0	53.3	41.3	60.8	55.1	27.3	48.5	60.3
	$\mathbf{F},\mathbf{z}^{\ell=4}$	45.2	32.2	38.3	32.8	54.8	54.7	24.1	40.3	57.6

Post-projector supervision yields the largest gain. We compare supervision at the projector output (anchor) against four alternatives: (i) vision-encoder intermediate layers (SigLIP blocks 7, 14, 21), (ii) the pre-projector vision-encoder output, (iii) LLM hidden states (layers 7, 14, 21) and the final readout token, and (iv) two-tap combinations of pre- and post-LLM supervision. Table 13 shows that supervision is effective only before the LLM, and that the projector output is the best point: `DeltaDirect` improves Dir avg. over the no-auxiliary baseline by +6.3 pp (70.4 → 76.7) while leaving MVBench unchanged (60.8). Pre-projector supervision is the runner-up (+5.5 pp), but it slightly degrades MVBench (60.2 vs. 60.8).

Vision-encoder taps leave the downstream representation almost unchanged (+0.3 to +0.9 pp). SigLIP already encodes motion direction densely, and supervising it adds gradient that does not propagate to the LLM-side readout. LLM-side taps fail in a different way: early-layer supervision (L7) collapses MCQ formatting and pushes MoDirect-SynBench accuracies to chance, while mid- and late-layer supervision (L14, L21) and readout supervision underperform even the no-auxiliary baseline. Two-tap combinations inherit this degradation. Direct supervision of post-LLM hidden states interferes with the autoregressive readout that produces the answer letter.

E.1.3 Direction Loss

Table 14: **Ablation: direction loss.** We compare MSE on $(\cos\theta, \sin\theta)$ (anchor) with a cosine loss. The cosine loss is 4.3 pp lower on Avg. Direction and comparable on MVBench.

Loss	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
	P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
MSE (anchor)	99.6	**85.5**	**86.1**	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	**60.8**
Cosine	**99.7**	84.0	78.2	64.5	75.9	69.3	35.5	72.4	60.4

MSE on the unit-vector target works best. MSE is widely used as a regression objective in video understanding tasks [2, 33, 69, 72, 93]. We compare the anchor loss, MSE on $(\cos\theta, \sin\theta)$, against a cosine loss $1 - \cos(\hat{\mathbf{m}}_t, \mathbf{m}_t)$. As shown in Table 14, MSE outperforms the cosine loss by 4.3 pp on Avg. Direction (76.7 vs. 72.4) and is slightly better on MVBench (60.8 vs. 60.4). Cosine loss is invariant to the magnitude of the prediction, whereas MSE on a unit-norm target additionally pulls the predicted vector toward unit length, which provides a tighter supervision signal. We therefore use MSE as the default direction loss.

E.1.4 Frame-Pair Operator

Table 15: **Ablation: frame-pair operator.** We compare subtraction with a learned concatenation operator. Subtraction outperforms concatenation on every direction task, while MVBench is comparable.

Operator	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
	P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
Subtraction (anchor)	**99.6**	**85.5**	**86.1**	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	**60.8**
Concatenation	99.5	79.4	76.4	60.6	74.9	68.9	36.2	70.8	60.6

The subtraction prior is useful. We compare the anchor operator, subtraction ($\mathbf{F}_{t+1} - \mathbf{F}_t$), with a learned concatenation operator (Linear($[\mathbf{F}_t; \mathbf{F}_{t+1}]$)). As shown in Table 15, concatenation underperforms subtraction by 5.9 pp on Avg. Direction (70.8 vs. 76.7), while MVBench is comparable (60.6 vs. 60.8). The linear layer over the concatenated features can in principle represent subtraction, so this gap suggests that the model does not recover the frame difference on its own under the limited supervision of MoDirect-Inst. We therefore adopt subtraction as the default operator, which also adds no parameters.

E.1.5 Loss Weight

Table 16: **Ablation: loss weight λ_{MVP}.** The total objective is $L = L_{\text{LM}} + \lambda_{\text{MVP}} \cdot L_{\text{MVP}}$. Avg. Direction increases monotonically with λ_{MVP} in the explored range, while MVBench is comparable.

λ_{MVP}	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
	P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
0.1	**99.8**	82.5	76.3	62.2	72.8	68.6	35.5	71.1	60.7
0.5	**99.8**	84.5	82.4	67.4	76.6	67.3	36.5	73.5	60.4
1.0 (anchor)	99.6	**85.5**	**86.1**	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	**60.8**

A larger λ_{MVP} improves direction performance without hurting MVBench. The total training objective is $L = L_{\text{LM}} + \lambda_{\text{MVP}} \cdot L_{\text{MVP}}$, and we sweep $\lambda_{\text{MVP}} \in \{0.1, 0.5, 1.0\}$. As shown in Table 16, Avg. Direction increases monotonically with λ_{MVP} ($71.1 \rightarrow 73.5 \rightarrow 76.7$). MVBench varies by at most 0.4 pp across the sweep (60.4–60.8), suggesting that λ_{MVP} strengthens the direction-specific signal without degrading general video understanding. We use $\lambda_{\text{MVP}} = 1.0$ as the default. Since Avg. Direction does not saturate within this range, larger values may yield further gains, which we leave for future work.

E.1.6 LoRA Rank

Table 17: **Ablation: LoRA rank.** LoRA rank r with $\alpha = 2r$ on the LLM; the projector is fully fine-tuned. Direction performance is monotone in rank; MVBench is essentially flat across the range.

Rank r	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
	P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
16	98.0	70.8	73.5	57.5	67.3	58.1	34.0	65.6	60.2
32	99.3	80.1	80.7	63.4	72.2	64.4	34.5	70.7	60.5
64 (anchor)	**99.6**	**85.5**	**86.1**	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	**60.8**

Direction performance is monotone in rank; $r = 64$ is on the budget knee. We apply LoRA with rank r and $\alpha = 2r$ to the LLM, with $r \in \{16, 32, 64\}$, while the projector is fully fine-tuned in all settings. $r = 16$ already learns the source domain task (Primitive-on-Syn 98.0%) but degrades on harder splits and on real-world directional tasks. MVBench is essentially flat across the range (60.2-60.8), so the cost of higher rank is borne entirely by the direction signal. We adopt $r = 64$ as the anchor.

E.1.7 Number of Frames

Table 18: **Ablation: frame count.** Frames per video T at fixed effective batch size (144). $T = 8$ is the optimal frame budget for both baseline and `DeltaDirect`; $T = 16$ and $T = 32$ both underperform. The relative gain of `DeltaDirect` over baseline is preserved at $T = 32$. *Avg. Direction* is the unweighted mean over the seven direction tasks.

Method	T	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction	MVBench [39]
		P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]		
Baseline	8	**99.7**	81.1	75.3	61.0	72.0	67.9	35.8	70.4	**60.8**
	32	96.6	71.1	63.3	51.2	67.3	59.8	27.8	62.4	60.1
`DeltaDirect`	**8 (anchor)**	99.6	**85.5**	**86.1**	**72.5**	**80.7**	**74.6**	**38.1**	**76.7**	**60.8**
	16	96.9	79.8	82.1	68.2	75.9	66.3	31.0	71.5	59.6
	32	96.4	77.6	82.3	68.1	76.5	69.6	28.5	71.3	59.4

Eight input frames are sufficient for learning motion direction. We compare $T \in \{8, 16, 32\}$ frames per video at fixed effective batch size (144). $T = 8$ (anchor) is best on Dir avg. (76.7); $T = 16$ drops to 71.5 and $T = 32$ to 71.3. The relative gain of `DeltaDirect` over baseline is preserved at $T = 32$ (+8.9 pp; baseline drops by 8.0 pp from $T = 8$ to $T = 32$). The synthetic clips contain a single coherent motion that 8 frames already capture, so extra frames add gradient noise (smaller per-step batch, more padding) without adding new content.

E.1.8 Supervision Target

In Table 3 of the main paper, we compare `DeltaDirect` against three alternative auxiliary objectives.

Frame order. We attach a small classification head to LLM hidden states at layer 16 that predicts each frame's temporal position via cross-entropy with target $[0, 1, \ldots, T-1]$. This post-LLM objective pressures the language model to maintain temporally discriminative representations, without requiring any direction labels.

Concatenating feature deltas. We compute per-pair frame deltas $\Delta_t = F_{t+1} - F_t$ from the projector output, average them over the spatial dimension, and concatenate the resulting $T-1$ motion tokens to the LLM input sequence. No auxiliary loss or learnable parameter is introduced; the LLM must learn to exploit the appended motion tokens through the standard language modeling objective alone.

Delta equivariance. We collect all spatially averaged frame deltas across the batch from the projector output and compute their pairwise cosine similarity matrix. An MSE loss aligns this representation similarity matrix with the corresponding direction ground-truth similarity matrix. This regularizer constrains relative structure rather than regressing direction directly, and introduces no additional learnable parameters.

Results. `DeltaDirect` outperforms all alternatives, and the direction-agnostic objectives provide limited gains, confirming that explicit direction supervision on the signed 2-D motion vector is necessary.

E.2 `DeltaDirect` Across Video-LLM Backbones

To verify that `DeltaDirect` generalizes beyond our main backbone, we apply it to additional Video-LLMs with minimal architecture-specific changes. We keep the loss formulation, head design, and training recipe identical across backbones, adjusting only the layer from which the head reads visual features.

Table 19: **Ablation: backbone robustness.** `DeltaDirect` replicated across three Video-LLM backbones. `DeltaDirect` consistently improves Avg. Direction across most backbones.

Backbone	Variant	MoDirect-SynBench				MoDirect-RealBench			Avg. Direction
		P-Syn	C-Syn	P-Real	C-Real	SSv2 [21]	KTH [63]	TOMATO [64]	
LLaVA-Video-7B [89]	Vanilla	27.6	23.4	26.9	25.8	52.2	50.3	26.8	33.3
	MoDirect-Inst	99.7±0.2	81.1±0.4	75.3±0.7	61.0±0.5	72.0±0.6	67.9±1.6	35.8±0.5	70.4±0.5
	`DeltaDirect`	99.6±0.1	85.5±0.9	86.1±0.9	72.5±0.9	80.7±0.9	74.6±0.9	38.1±0.7	76.7±0.3
LLaVA-OneVision-7B [37]	Vanilla	28.3	23.4	29.9	29.3	52.2	50.3	26.8	34.3
	MoDirect-Inst	99.8	88.2	82.2	76.9	65.0	51.6	31.3	70.7
	`DeltaDirect`	99.6	89.6	83.2	78.3	66.6	51.5	32.0	71.5
mPLUG-Owl3-7B [81]	Vanilla	25.6	24.2	25.8	25.4	43.4	50.3	21.3	30.9
	MoDirect-Inst	100.0	99.9	93.6	89.5	94.4	91.3	37.2	86.6
	`DeltaDirect`	100.0	99.2	92.6	92.8	93.2	87.1	31.5	85.2

`DeltaDirect` improves direction accuracy across all tested backbones. We replicate `DeltaDirect` on three Video-LLM backbones: LLaVA-Video-7B, LLaVA-OneVision-7B, and mPLUG-Owl3-7B. `DeltaDirect` improves Avg. Direction over the corresponding instruction-tuning baseline on every backbone, with consistent gains on LLaVA-Video (+6.3 pp) and LLaVA-OneVision (+0.8 pp). The smaller gain on LLaVA-OneVision reflects a stronger baseline (Avg. Direction 70.7%), already close to that of LLaVA-Video. We adopt LLaVA-Video-7B as the primary backbone for the strongest baseline and the cleanest signal for ablation analysis throughout the paper.

E.3 Additional Analysis of `DeltaDirect`

The previous subsection shows that `DeltaDirect` substantially improves motion direction accuracy. We now ask *why*, tracing the effect from the projector output through the LLM using the diagnostic framework from Section 3 and Section 4.

Domain-invariant direction signals emerge at the projector output. To locate where the improvement originates, we probe the projector output $\mathbf{F}$. Specifically, we train a four-way direction probe on Primitive-on-Syn and evaluate it on the OOD domains of MoDirect-SynBench (Table 20). Instruction tuning alone remains close to the vanilla model, with an OOD average of 42.2% and only 27.5% on Cutout-on-Real. With `DeltaDirect`, the OOD average increases to 68.6%, and Cutout-on-Real accuracy rises to 55.4%.

A small projector shift amplifies the OOD direction signal. The LLaVA-Video-7B projector is a two-layer MLP that maps SigLIP vision tokens into the LLM embedding space. Each row of a linear layer's weight matrix is one output channel, so we compare the MoDirect-Inst and `DeltaDirect` projectors via row-wise cosine similarity and relative magnitude difference across

Table 20: **`DeltaDirect` improves cross-domain direction probing at the projector output.** We train a direction probe on Primitive-on-Syn projector outputs and evaluate out of domain. OOD Avg. is the macro average over C-Syn, P-Real, and C-Real.

Model	OOD Avg.	C-Real
LLaVA-Video-7B	42.8	28.5
LLaVA-Video-7B w/ MODIRECT-INST	42.2	27.5
LLaVA-Video-7B w/ `DeltaDirect`	**68.6**	**55.4**

Table 21: **`DeltaDirect` selectively amplifies the direction concept vector magnitude on OOD domains.** Direction concept-vector magnitude $\|\mathbf{v}_d^{\ell=21}\|$ at layer 21 under MODIRECT-INST and `DeltaDirect`. The relative change $\Delta(\%)$ grows from near-zero on the source domain Primitive-on-Syn to $+20.4\%$ on the most OOD Cutout-on-Real.

Domain	MODIRECT-INST	`DeltaDirect`	$\Delta(\%)$
Primitive-on-Syn (Source domain)	28.92	28.29	-2.2
Cutout-on-Syn	20.49	20.85	$+1.8$
Primitive-on-Real	18.47	21.37	$+15.7$
Cutout-on-Real (Hardest OOD)	14.15	17.03	$+20.4$

both layers. `DeltaDirect` preserves the direction of every projector row and rescales only its magnitude. Row-wise cosine similarity with the MODIRECT-INST projector stays above 0.99, while the relative magnitude difference is approximately 4%. This magnitude-only shift selectively amplifies the motion-direction signal on OOD domains. Table 21 shows that the direction concept vector magnitude at layer 21 rises by $+20.4\%$ on the most OOD Cutout-on-Real, while the source domain Primitive-on-Syn stays nearly unchanged (-2.2%). This is consistent with the magnitude deficit diagnosed in Section 4.4.

F Case Study

F.1 Direction-aware Motion Understanding

We present additional qualitative examples demonstrating how direction supervision improves motion-aware video understanding under open-ended generation settings. Unlike multiple-choice evaluation, open-ended video description requires the model to jointly capture object identity, motion dynamics, and temporal relationships without explicit answer candidates.

Figure 15 shows an example from Something-Something-V2. Given the prompt "Describe the video in detail," the baseline LLaVA-Video produces a partially wrong description, incorrectly interpreting the manipulated object as a "pen" or "marker" and describing a nonexistent "ink trail" on the paper. Although the model recognizes that motion occurs, it describes the movement only vaguely as a "sweeping motion" without explicitly grounding the motion direction or trajectory. In contrast, `DeltaDirect` generates a more spatially and temporally grounded description by explicitly identifying the leftward movement of the object before the final placement event on the paper. This example suggests that direction supervision improves not only directional prediction itself, but also motion grounding in free-form video description.

Figure 16 presents an additional example from TOMATO under the prompt "Share a detailed breakdown of the video's storyline and visuals." While the baseline model summarizes the activity only coarsely as a sequence of generic hand gestures followed by a final arm extension, `DeltaDirect` produces a more fine-grained motion description by explicitly identifying intermediate directional transitions, including leftward and rightward hand movements before the final extension. Rather than compressing the action into a single high-level summary, the generated response decomposes the motion into sequential motion events that more closely align with the observed video dynamics.

Overall, these examples suggest that direction supervision encourages more precise motion-aware descriptions, improving the grounding of motion trajectories and directional transitions in open-ended video understanding.

F.2 Failure Cases in Open-ended Generation

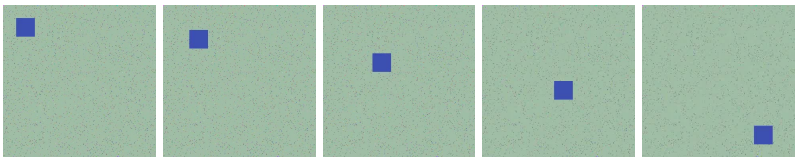
Although direction supervision often improves motion-aware generation, it can sometimes encourage the model to produce overly specific motion descriptions even when the visual evidence is ambiguous. Figure 17 shows an example from YouCook2 under the prompt "How does the hand interact with the sandwich in the video?" The baseline LLaVA-Video generates a simple but visually grounded response describing the placement of cheese slices on the sandwich. In contrast, `DeltaDirect` additionally introduces the phrase "moving in a clockwise direction," even though the observed hand movement does not clearly exhibit a clockwise trajectory. This example suggests that explicit motion supervision may increase the tendency to generate detailed motion descriptions, occasionally leading to unsupported or inaccurate motion interpretations in open-ended video understanding settings.

G Limitations

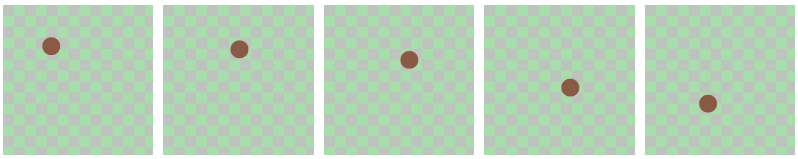
This work focuses on signed 2-D image-plane motion direction, a basic but limited form of motion understanding. Our benchmarks and supervision do not cover depth motion, acceleration, non-rigid deformation, multi-object interaction, or long-horizon event dynamics. Moreover, `DeltaDirect` uses synthetic videos where the 2-D motion vector is analytically available; extending the same supervision to arbitrary real videos may require pseudo-labels from tracking, optical flow, or human annotation. Our controlled synthetic benchmark assumes a single dominant moving object and a clear viewer-centric direction label, whereas real videos may contain camera motion, occlusion, or multiple objects moving in different directions. Finally, our main analysis uses LoRA post-finetuning as the primary training regime. We include from-scratch full fine-tuning results to verify that the benefit of `DeltaDirect` is not limited to adapter-based training, but a broader full-training study across larger backbones and training recipes remains computationally expensive and is left for future work. `DeltaDirect` adds no inference-time computation because the MVP head is removed after training, but it incurs a small training-time overhead for the auxiliary MVP loss. Our diagnostic analyses also require additional probe training, which is used only for analysis and not for deployment.

H Broader Impacts

This work identifies *directional motion blindness* in Video-LLMs: motion direction is linearly decodable from internal representations, yet the model fails to reliably bind it to the correct verbal response. Our diagnosis reveals that this failure stems from a direction binding gap rather than from missing visual perception. We show that `DeltaDirect` improves motion direction understanding on both synthetic and real-world videos, which may benefit applications that require fine-grained motion understanding, e.g., robotics, autonomous navigation, and assistive technologies. Our results suggest that improving Video-LLMs may require diagnosis-driven supervision beyond scaling models or datasets. However, `DeltaDirect` relies on synthetic motion supervision with simplified dynamics and controlled trajectories, which may not fully capture the complexity of real-world motion. Improved motion understanding capabilities could also be applied to surveillance or tracking systems. We encourage future work on responsible deployment and evaluation of such technologies.



(a) MoDirect-Inst Perturbed Linear motion example



(b) MoDirect-Inst Roundtrip motion example

Figure 9: Examples of MoDirect-Inst.



(a) Primitive-on-Syn



(b) Cutout-on-Syn

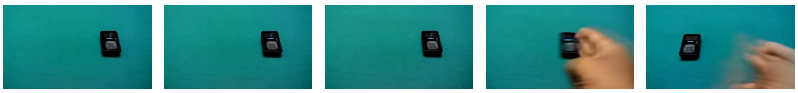


(c) Primitive-on-Real



(d) Cutout-on-Real

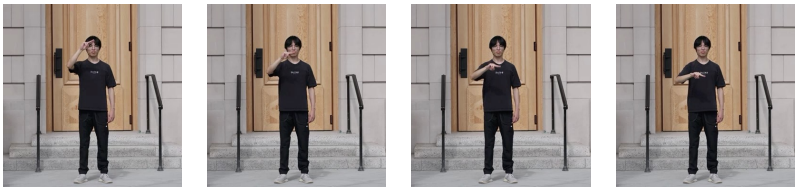
Figure 10: Examples of MoDirect-SynBench.



(a) Something-Something-V2 [21]



(b) KTH [63]



(c) TOMATO [64]

Figure 11: Examples of MoDirect-RealBench.

Default

Visual Prompting: **Plain**

[Question] From the viewer's perspective, in which direction is the object moving in this video?

A. Right B. Left

Visual Prompting: **Color Edge**

[Question] From the viewer's perspective, which colored edge does the object move toward?

A. Green B. Yellow

Visual Prompting: **Text Edge**

[Question] From the viewer's perspective, which text label does the object move toward?

A. Right B. Left

Figure 12: **Default Prompting Example.**

Temporal

Visual Prompting: **Plain**

[Task] Watch the video carefully and identify the direction of movement of the main object.

Let's think **step by step**.

First, describe what object is moving in the video.

Second, describe where the object is located at the beginning of the video.

Third, describe where the object is located at the end of the video.

Finally, based on the position change, select the correct answer.

End your response with: Therefore, the answer is <your choice>.

[Question] From the viewer's perspective, in which direction is the object moving in this video?

A. Left B. Right

Visual Prompting: **Color Edge**

[Task] Watch the video carefully. The video has colored borders on each edge: Red on top, Blue on bottom, Green on left, Yellow on right. Identify which colored border the main object is moving toward.

Let's think **step by step**.

Step 1: Describe which colored border the object is closest to at the beginning of the video.

Step 2: Describe which colored border the object is closest to at the end of the video.

Step 3: Based on the position change, determine which colored border the object is moving toward and select the correct answer.

End your response with: Therefore, the answer is <your choice>.

[Question] From the viewer's perspective, which colored edge does the object move toward?

A. Yellow B. Green

Visual Prompting: **Text Edge**

[Task] Watch the video carefully. The video has directional text labels on each edge: 'Up' on top, 'Down' on bottom, 'Left' on the left, 'Right' on the right. Identify which text label the main object is moving toward.

Let's think **step by step**.

Step 1: Describe which text label the object is closest to at the beginning of the video.

Step 2: Describe which text label the object is closest to at the end of the video.

Step 3: Based on the position change, determine which text label the object is moving toward and select the correct answer.

End your response with: Therefore, the answer is <your choice>.

[Question] From the viewer's perspective, which text label does the object move toward?

A. Left B. Right

Figure 13: **Temporal Prompting Example.**

Grid Prompt

Visual Prompting: **Plain**

[Task] Watch the video carefully and identify the direction of movement of the main object.
[Grid] 8x8, [0,0]=bottom-left, [7,7]=top-right. right=+x, left=-x, up=+y, down=-y.
Let's think step by step.
Step 1: Estimate the object's start positions on the grid.
Step 2: Estimate the object's end positions on the grid.
Step 3: Based on coordinate change, determine direction.
End your response with: Therefore, the answer is <your choice>.
[Question] From the viewer's perspective, in which direction is the object moving in this video?
A. Left B. Right

Visual Prompting: **Color Edge**

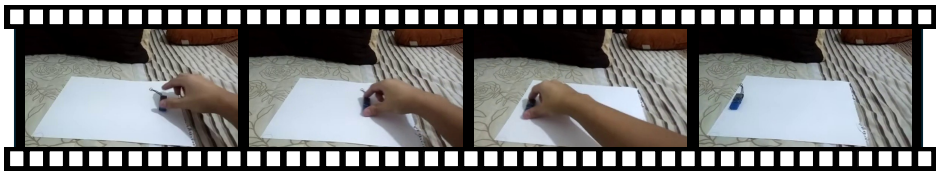
[Task] Watch the video carefully and identify the direction of movement of the main object.
[Grid] 8x8, [0,0]=bottom-left, [7,7]=top-right. right=+x, left=-x, up=+y, down=-y.
Let's think step by step.
Step 1: Estimate the object's start positions on the grid.
Step 2: Estimate the object's end positions on the grid.
Step 3: Based on coordinate change, determine direction.
End your response with: Therefore, the answer is <your choice>.
[Question] From the viewer's perspective, which colored edge does the object move toward?
A. Yellow B. Green

Visual Prompting: **Text Edge**

[Task] Watch the video carefully and identify the direction of movement of the main object.
[Grid] 8x8, [0,0]=bottom-left, [7,7]=top-right. right=+x, left=-x, up=+y, down=-y.
Let's think step by step.
Step 1: Estimate the object's start positions on the grid.
Step 2: Estimate the object's end positions on the grid.
Step 3: Based on coordinate change, determine direction.
End your response with: Therefore, the answer is <your choice>.
[Question] From the viewer's perspective, which text label does the object move toward?
A. Left B. Right

Figure 14: **Grid Prompting Example.**

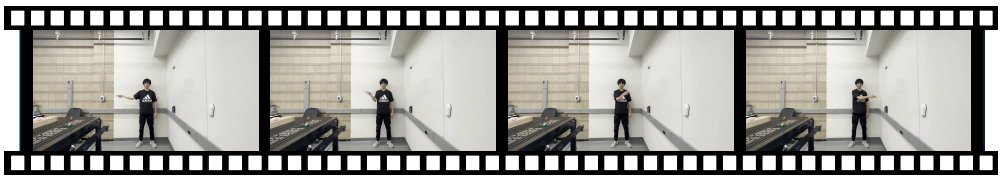
Something-Something v2



User Prompt	Describe the video in detail.
LLaVA-Video	The video opens with a close-up of a hand holding a small, dark-colored object, which appears to be a pen or marker. The hand is positioned over a blank white sheet of paper placed on a bed with a striped pattern and floral designs. The hand moves the object in a sweeping motion across the paper, leaving a trail of ink behind. The background includes a pillow with a brown cover and another pillow with an orange cover. The video concludes with the hand moving away from the paper, leaving the ink trail visible on the paper.
DeltaDirect	The video features a close-up view of a hand holding a blue USB flash drive. The background consists of a bed with a striped and floral-patterned bedspread. The hand moves the USB flash drive to the left around the paper, then places it on the paper. The video concludes with the USB flash drive resting on the paper.

Figure 15: **Qualitative comparison on Something-Something-V2 with an open-ended description prompt.** Compared to the baseline LLaVA-Video, `DeltaDirect` generates a more grounded description of both the object and its leftward motion. Blue text highlights motion- and direction-related expressions.

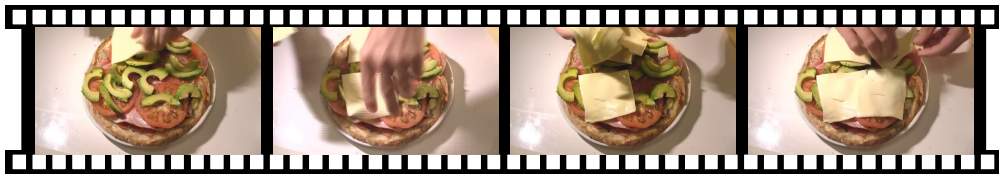
TOMATO



User Prompt	Share a detailed breakdown of the video's storyline and visuals.
LLaVA-Video	The video features a person standing in an indoor setting with a plain white wall and a gray floor. The individual is dressed in a black Adidas t-shirt, black cargo pants, and white sneakers. Throughout the video, the person performs a series of hand gestures, starting with a raised right hand, followed by a waving motion, then a pointing gesture, and finally extending their right arm horizontally to the side. The lighting in the room is even, and there are no other objects or people visible in the frames.
DeltaDirect	The video features a person standing in an indoor setting with a plain background. The individual is dressed in a black T-shirt with a white Adidas logo, black cargo pants, and white sneakers. Throughout the video, the person performs a sequence of hand movements. Initially, the right hand is raised with the palm facing outward. As the video progresses, the hand moves left, then right, and finally extends to the side with the palm facing forward.

Figure 16: **Qualitative comparison on TOMATO with an open-ended description prompt.** Compared to the baseline LLaVA-Video, `DeltaDirect` provides a more precise description of the person's hand motion, explicitly capturing its directional changes from left to right. Blue text highlights motion- and direction-related expressions.

YouCook2



User Prompt	How does the hand interact with the sandwich in the video?
LLaVA-Video	The hand is seen adding more slices of cheese to the sandwich, ensuring an even distribution.
DeltaDirect	The hand is seen adding cheese slices to the sandwich, moving in a clockwise direction.

Figure 17: **Qualitative comparison on YouCook2 under an open-ended video understanding prompt.** Compared to the baseline LLaVA-Video, `DeltaDirect` tends to generate explicit directional motion descriptions, which can sometimes be unsupported by the actual video content. Red text highlights potentially incorrect expressions.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly and accurately reflect the paper's core contributions and scope. Specifically, the paper claims (i) the identification of *directional motion blindness* as a systematic failure mode in Video-LLMs, (ii) a diagnosis that localizes this failure to a *direction binding gap*, and (iii) a solution consisting of the MoDirect dataset and the `DeltaDirect` objective. These claims are consistently supported by the analysis sections and the experimental evaluation sections of the paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have mentioned limitations in Appendix G

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper does not include any theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper provides sufficient details to reproduce the main experimental results, including dataset construction, model architecture, training configurations, analysis procedures, and evaluation protocols. Additional implementation details are provided in Appendix A, analysis protocols are described in Appendix C, instruction-data analysis details are provided in Appendix D.1, and additional experimental setups and analyses are included in Appendix E.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: The MoDirect dataset and codebase are publicly available at https://huggingface.co/datasets/KHUjongseo/Modirect-family and https://github.com/KHU-VLL/DeltaDirect. The paper and appendix additionally describe the dataset construction, generation procedures, training configurations, analysis protocols, and evaluation settings in Appendix A, Appendix B.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The paper specifies the training and evaluation settings in sufficient detail to understand the reported results. The main text describes the model architecture, training procedures, and evaluation protocols, while detailed implementation settings, hyperparameters, optimizer configurations, and fine-tuning setups are provided in Appendix A. Dataset construction, benchmark configurations, and data splits are also clearly defined in Appendix B.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: For our fine-tuned models (LLaVA-Video-7B w/ MoDirect-Inst and w/ DeltaDirect), we report the mean $\pm$ 1-sigma standard deviation (sample std) over three training runs with different random seeds, which vary model initialization of the newly

added parameters and training data order in Table 1. Baseline models are evaluated off-the-shelf with greedy decoding, which is deterministic. The Qwen2-0.5B full fine-tuning results are from a single run due to compute constraints.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide detailed information about the compute resources used for the experiments, such as GPU types, memory, or training time in Appendix A.4 and Appendix A.3.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in this paper fully conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The broader impacts are discussed in Appendix H

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Our experiments use third-party datasets under their original licenses or terms of use and do not relicense those data. Specifically, our synthetic data are constructed using object cutouts derived from MS COCO and scene backgrounds from Places365. For MS COCO, the annotations are released under CC BY 4.0, while the copyrights of the

underlying images remain with the original image owners and associated source-image licenses. For Places365, the official documentation states that the copyrights of all images belong to the image owners. For the real-video benchmark, we use curated subsets from Something-Something-V2, KTH, and TOMATO. Something-Something-V2 is distributed under a research-use data license agreement, the KTH Action Dataset is publicly available for academic research, and the TOMATO dataset card specifies a CC BY-SA 4.0 license. In training, we additionally use publicly released VideoChat2-IT annotations. Its documentation states that only the annotation files are directly distributed and that the original videos must be obtained from the provided source links. The dataset card further states that video copyrights belong to the original creators or hosting platforms and that additional source-dataset agreements may apply.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: We introduce a new dataset (MoDirect) and benchmark, and describe their construction, structure, and usage in detail in the paper and supplementary material. We will release the dataset and code with documentation, usage instructions, and license information upon acceptance.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: This paper does not involve research with human participants, and therefore IRB approvals or equivalent institutional review procedures are not applicable.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: The paper uses existing open-source Video-LLMs as experimental backbones for analysis and evaluation. In addition, GPT-based semantic filtering is used in the instruction-data analysis pipeline to identify direction-dependent QA pairs (Appendix D.1). The roles of these models, together with the associated analysis procedures and experimental settings, are described in the main paper and supplementary material.

    Guidelines:

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