CVPR #22

4D-RGPT: Toward Region-level 4D Understanding via Perceptual Distillation

Anonymous CVPR submission

Paper ID 22

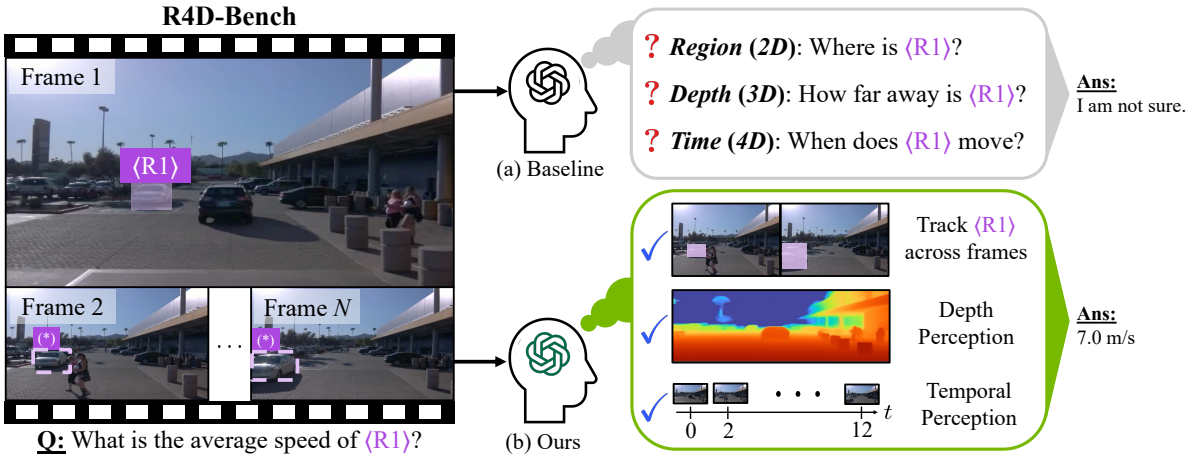


Figure 1. **Overview of Region-level 4D Understanding.** 4D region-level VQA, *e.g.*, our R4D-Bench, requires MLLMs to be able to track regions (2D), perceive depth (3D), and temporal progression (4D). Baseline MLLMs cannot recognize one or more of these aspects and thus fail to answer questions correctly. With our distillation framework, our 4D-RGPT better perceives these aspects and answers accurately. We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

Abstract

Despite advances in Multimodal LLMs (MLLMs), their ability to reason over 3D structures and temporal dynamics remains limited, constrained by weak 4D perception and temporal understanding. Existing 3D and 4D Video Question Answering (VQA) benchmarks also emphasize static scenes and lack region-level prompting. We tackle these issues by introducing: **(a)** *4D-RGPT, a specialized MLLM designed to capture 4D representations from video inputs with enhanced temporal perception;* **(b)** *Perceptual 4D Distillation (P4D), a training framework that transfers 4D representations from a frozen expert model into 4D-RGPT for comprehensive 4D perception; and* **(c)** *R4D-Bench, a benchmark for depth-aware dynamic scenes with region-level prompting, built via a hybrid automated and human-verified pipeline. Our 4D-RGPT achieves notable improvements on both existing 4D VQA benchmarks (+5.3% across 6 benchmarks) and the proposed R4D-Bench benchmark (+4.3%).*

1. Introduction

By integrating visual inputs with Large Language Models (LLMs) [1, 14, 46, 76], Multimodal LLMs (MLLMs) demonstrate remarkable capabilities in complex understanding across vision and language modalities. However, current MLLMs, even proprietary models such as GPT-4o [45], often struggle with highly specialized tasks that require fine-grained spatial[1] and temporal visual understanding.

In this paper, we advance MLLMs for one such challenging task: *Region-level 4D Understanding*. This unique problem combines two critical aspects: (1) **4D understanding**, which demands answering questions regarding depth information, temporal dynamics, or object interactions in 3D space over time; and (2) **region-level understanding**, which requires grounding language queries to specific visual regions for controllable input. Region-level 4D VQA is essential for demanding real-world applications, such as autonomous driving and industrial inspection, where 4D information is critical and user queries must precisely target specific regions rather than rely on ambiguous descriptions.

[1]We use "spatial" in this paper to refer to 3D (*i.e.*, 2D + depth), rather than 2D as in several general video understanding works.

As an example, in Fig. 1, the 4D question "*What is the average speed of* $\langle R1 \rangle$*?*" specifically targets the speed of the car marked by the purple bounding box $\langle R1 \rangle$.

To achieve 4D understanding, previous works mainly rely on conventional Supervised Fine-Tuning (SFT) [23, 43, 75, 84] or Reinforcement Learning (RL) [28, 42, 47, 58, 74] paradigms, optimizing primarily over the final text output using self-curated data. However, due to the difficulty of curating large-scale, well-annotated dynamic video data, these works often struggle with dynamic scenarios. In region-level 4D VQA, having strong 4D understanding is even more critical, as it requires tracking region movement over time. More recently, several works [8, 9, 11, 15, 73, 88, 89] exploit external models to inject 3D knowledge into MLLMs to improve spatial understanding capabilities. However, external 3D knowledge mainly helps understand static videos, without fully achieving 4D understanding. Moreover, these approaches often integrate additional modules into the architecture, introducing additional inference burdens.

To address these challenges, we propose ***4D-RGPT***, a specialized MLLM with effective *4D perception* and thus better 4D understanding capabilities. 4D perception refers to the ability to extract low-level 4D perceptual knowledge, *e.g.*, depth and optical flow. Specifically, 4D-RGPT perceives 4D knowledge via our proposed **P**erceptual **4**D **D**istillation (**P4D**) *training-only* framework. P4D adopts both latent and explicit distillation processes to effectively distill 4D perceptual knowledge from an expert 4D teacher model into the student 4D-RGPT. Notably, unlike previous works, P4D contains only *training-only* modules, incurring no additional inference cost. Finally, we introduce Timestamp Positional Encoding (TPE) to provide explicit temporal cues, enhancing MLLMs' temporal perception capability.

Finally, while various 3D/4D VQA benchmarks have been proposed recently [15, 22, 30, 56, 79, 91], they often lack either region-prompted questions or sufficient 4D understanding challenges. As demonstrated in Fig. 1, this limitation prevents comprehensive evaluation of region-based 4D VQA capabilities, namely, answering questions about specific regions (*e.g.*, $\langle R1 \rangle$) in a 4D context. To bridge this gap, we construct ***R4D-Bench***, a new benchmark containing both static and dynamic scene understanding tasks with region-based 4D questions.

Our experiments show that 4D-RGPT improves over the baseline on both non-region-based 3D/4D benchmarks (+5.3% on average across 6 benchmarks) and our region-based R4D-Bench benchmark (+4.3%), while effectively capturing explicit 4D signals.

Our main contributions are as follows:

- We propose *4D-RGPT* (Sec. 4.1), a specialized MLLM that perceives 4D information for enhanced understanding.
- We propose the *P4D* (Sec. 4.2) training framework to distill 4D perceptual knowledge into 4D-RGPT without introducing additional inference cost.
- We introduce *R4D-Bench* (Sec. 5), a region-based 4D VQA benchmark that requires region-level 4D understanding.

Table 1. **Comparison among 3D / 4D VQA Benchmarks.** Existing benchmarks either lack dynamic video data or region prompts, while our R4D-Bench is the first to provide both at scale. All benchmarks are downloaded from official sources as of August 2025, and the numbers of VQA might differ from the original papers. Static videos contain only camera movement, while dynamic videos contain both camera and object movement. †We only adopt real-world videos from the VLM4D benchmark.

Dataset	Regions	Input Type	FPS	# Visual	# QA
SAT-real [56]	✗	Images	-	196	150
MMSI-Bench [79]	✗	Images	-	2.5k	1.0k
OmniSpatial [22]	✗	Images	-	561	1.5k
VSTI-Bench [15]	✗	Static Video	24	312	6k
STI-Bench [30]	✗	Dynamic Video	$10 \sim 30$	369	2k
VLM4D-real† [91]	✗	Dynamic Video	$12 \sim 24$	600	1k
R4D-Bench (Ours)	✓	Dynamic Video	$10 \sim 30$	780	1.5k

2. Related Work

2.1. Multimodal LLMs (MLLMs)

The success of LLMs [1, 3, 14, 46, 65, 66, 76] has inspired various MLLMs [12, 31, 33, 34, 38, 45, 52, 61] for multimodal understanding or generation. While several works [17, 37, 57, 59, 81, 90] excel at video understanding, they lack specialization in region-level or 3D/4D tasks.

Region-Level MLLMs understand specified regions within visual inputs. Earlier works [6, 7, 24, 39, 48, 51, 63, 69, 87, 92] use bounding box coordinates as text prompts, while others [11, 32, 41, 44, 70, 85] extract Region of Interest (RoI) visual features. Visual markers [5, 25, 72, 77] provide intuitive region indication. However, region-level video understanding remains challenging, especially for dynamic scenes where user queries provide sparse region annotations without temporal tracking (Fig. 1). While recent works [11, 19] address this, they do not fully explore 4D dynamic scenarios. We propose *4D-RGPT* (Sec. 4.1) to interpret 4D spatio-temporal knowledge without 4D annotations during training.

3D/4D MLLMs focus on spatial and temporal understanding. Previous works [8, 11, 15, 21, 29, 47, 56, 60, 75, 88] enhance MLLMs with depth or 3D reconstruction models but require additional modules, introducing inference costs. Others use SFT [23, 43, 75, 84] or RL [28, 42, 47, 58, 74] with text-based supervision, which is insufficient for 4D perception. We propose *P4D* (Sec. 4.2) to enhance 4D perception without modifying the architecture. Prior works [53, 68] distill into vision-only encoders, *e.g.*, ViT, VideoMAE. 3DRS [21] employs distillation for static 3D scenes, while P4D addresses dynamic scenes.

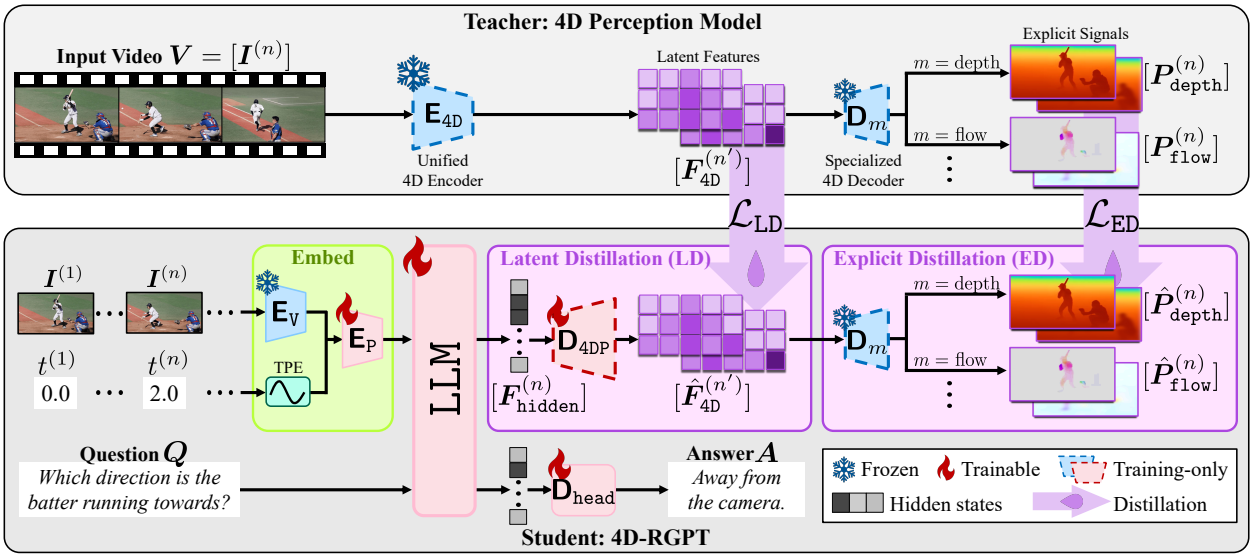


Figure 2. **Perceptual 4D Distillation (P4D) framework for 4D-RGPT.** For each frame $I^{(i)}$ in V, 4D-RGPT extracts 4D representations through training-only modules, *i.e.*, $\mathbf{D}_{4DP}$ and $\mathbf{D}_m$ for $m \in \mathcal{M}$. This includes both latent features, *i.e.*, $\hat{F}_{4D}$, and explicit signals, *e.g.*, depth $\hat{P}_{depth}$ or optical flow maps $\hat{P}_{flow}$. We also incorporate timestamp positional encodings (TPE) to provide temporal cues for 4D-RGPT to be temporally aware. In the P4D framework, the frozen teacher, *i.e.*, 4D perception model, captures 4D expert knowledge from V. It is then distilled to the student 4D-RGPT via two strategies. (a) *Latent Distillation (LD)*: We align the latent $\hat{F}_{4D}$ with the teacher's intermediate 4D embeddings F_{4D}. (b) *Explicit Distillation (ED)*: We align the explicit $\hat{P}_m$ with the teacher's final 4D signals P_m. 4D-RGPT is optimized end-to-end using both SFT loss and the distillation losses, *i.e.*, $\mathcal{L}_{LD}$ and $\mathcal{L}_{ED}$.

2.2. 3D/4D VQA Benchmarks

Several benchmarks evaluate MLLMs' 3D and 4D understanding. OmniSpatial [22], VSTI-Bench [15], SAT [56], and MMSI-Bench [79] focus on 3D spatial understanding in images. STI-Bench [30] is a pioneering work that introduces 4D VQA on both static and dynamic videos, while VLM4D [91] focuses on semantic understanding in dynamic videos. However, these benchmarks lack region-level prompting or sufficient dynamic video data (Tab. 1). We introduce *R4D-Bench* (Sec. 5) with region-level prompts and diverse 4D understanding tasks.

3. Preliminaries and Notations

We briefly review the background and introduce notation for an MLLM and a 4D perception model.

Multimodal LLMs extend the understanding capabilities of LLMs to visual inputs such as images and videos. The architecture typically consists of: (a) $\mathbf{E}_V$: a vision encoder for input visuals, *e.g.*, images or videos; (b) $\mathbf{E}_P$: a multi-modal projector that aligns the visual and textual features within a shared space; (c) LLM: an auto-regressive model that takes in both features and generates output hidden states or tokens in a step-by-step manner; (d) $\mathbf{D}_{head}$: a linear head layer that maps the hidden states to the final vocabulary space for text generation.

4D Perception Models, *e.g.*, L4P [2], encode a latent feature from input visuals for multiple 4D low-level representations. They consist of a unified encoder $\mathbf{E}_{4D}$ and specialized decoders $\mathbf{D}_m$ for each 4D modality $m \in \mathcal{M}$. Each 4D modality $m \in \mathcal{M}$ describes some per-pixel 4D properties of the input video. For example, m can be either "depth," which describes the per-pixel depth values, or "flow," which describes the per-pixel optical flow between adjacent frames.

We denote the input video as $V = [I^{(n)}]_{n=1:N}$ with each image frame $I^{(n)} \in \mathbb{R}^{H \times W \times 3}$. Here, N is the number of input frames and (H, W) is the spatial size. Given V, we can acquire its 4D latent representation as follows,

$$F_{4D} = \mathbf{E}_{4D}(V) \in \mathbb{R}^{N' \times h' \times w' \times c'}, \quad (1)$$

where N', h', w' are the down-sampled number of frames, height, and width of $\mathbf{E}_{4D}$'s outputs and c' is the number of output channels.

For each m, the decoder $\mathbf{D}_m$ decodes F_{4D} to its corresponding low-level representation, *i.e.*,

$$P_m = \mathbf{D}_m(F_{4D}). \quad (2)$$

We use the following 4D modalities $\mathcal{M}$ in this work: (a) $m =$ depth where $P_{depth}^{(n)} \in \mathbb{R}^{H \times W \times 1}$ describes the per-pixel depth values; (b) $m =$ flow where $P_{flow}^{(n)} \in \mathbb{R}^{H \times W \times 2}$ describes the per-pixel optical flow between adjacent frames;

(c) $m = \texttt{motion}$ where $\boldsymbol{P}^{(n)}_{\texttt{motion}} \in \mathbb{R}^{H\times W\times 1}$ describes whether a pixel is moving or static in 3D space; (d) $m = \texttt{camray}$ where $\boldsymbol{P}^{(n)}_{\texttt{camray}} \in \mathbb{R}^{H\times W\times 6}$ describes the per-pixel Plucker ray maps.

4. Approach

Overview. Given a video $\boldsymbol{V}$ and a question $\boldsymbol{Q}$, an MLLM responds with an answer $\boldsymbol{A}$ autoregressively. To tackle the complex, dynamic scenes presented in 4D VQA benchmarks, we develop an MLLM that can better answer questions by incorporating 4D knowledge from a teacher model and leveraging low-level representations, *e.g.*, depth and flow, over time. To this end, we design ***4D-RGPT*** to capture both *latent* 4D features and *explicit* 4D signals from $\boldsymbol{V}$ with **training-only** modules. These 4D representations enable the model to better perceive 4D knowledge during training, without introducing additional inference cost. Additionally, to accurately capture temporal progression for answering 4D questions, we introduce Timestamp Positional Encoding (TPE) to provide explicit temporal cues to the MLLM.

To circumvent the extreme training cost and instability of training MLLMs from scratch, we introduce our **P**erceptual **4**D **D**istillation (**P4D**) framework to distill 4D knowledge into 4D-RGPT during training. As shown in Fig. 2, we leverage a frozen expert 4D perception model [2] (teacher), to supervise both latent and explicit 4D representations of 4D-RGPT (student). The latent distillation provides intermediate guidance on abstract 4D features, while the explicit distillation ensures accurate extraction of interpretable low-level 4D signals. We describe the 4D-RGPT architecture in Sec. 4.1 and the P4D framework in Sec. 4.2.

4.1. 4D-RGPT

Given an input video $\boldsymbol{V}$ with N sampled frames $[\boldsymbol{I}^{(n)}]_{n=1}^{N}$, and the timestamps $\{t^{(n)}\}_{n=1}^{N}$ of each frame, our 4D-RGPT consists of training-only 4D perception modules that can extract 4D representations for distillation in P4D (Sec. 4.2). Moreover, 4D-RGPT can perceive temporal progression by incorporating timestamp positional encodings into input visual features. In short, we use a 4D perception decoder $\mathbf{D}_{\texttt{4DP}}$ to extract latent 4D features and prediction heads $\mathbf{D}_m$ for $m \in \mathcal{M}$ to extract explicit 4D signals.

Latent 4D Representations. To capture latent 4D representations for P4D, we extract $\hat{\boldsymbol{F}}_{\texttt{4D}}$ from the input video. Through the video encoder $\mathbf{E}_{\texttt{V}}$, multi-modal projector $\mathbf{E}_{\texttt{P}}$, and LLM, each frame $\boldsymbol{I}^{(n)}$ is encoded as hidden state features $\boldsymbol{F}^{(n)}_{\texttt{hidden}} \in \mathbb{R}^{h\times w\times c}$, where $l = hw$ is the number of per-image tokens, (h, w) is the spatial size of visual features, and c is the hidden dimension. We introduce a *training-only* MLP as a 4D perception decoder $\mathbf{D}_{\texttt{4DP}}$ on top of the MLLM to decode latent 4D representations $\hat{\boldsymbol{F}}^{(n)}_{\texttt{4D}}$. Specifically, we first sample and resize (`Rearrange`) the hidden $\boldsymbol{F}^{(n)}_{\texttt{hidden}}$ to match the target shape of (N', h', w') in Eq. 1. Thus, for each down-sampled frame $n' \in [1, N']$, we have

$$\hat{\boldsymbol{F}}^{(n')}_{\texttt{4D}} = \mathbf{D}_{\texttt{4DP}}\left(\texttt{Rearrange}(\boldsymbol{F}^{(n)}_{\texttt{hidden}})\right). \tag{3}$$

Explicit 4D Representations. Although $\hat{\boldsymbol{F}}_{\texttt{4D}}$ can capture rich 4D features, explicit 4D signals, *e.g.*, depth maps, are more interpretable and provide unambiguous supervision. To capture explicit 4D representations for P4D, we extract explicit 4D signals $\hat{\boldsymbol{P}}_m$ given $\hat{\boldsymbol{F}}_{\texttt{4D}}$ via the *training-only* prediction heads $\mathbf{D}_m$ from the frozen 4D perception model. Specifically, for each $m \in \mathcal{M}$, we have

$$\hat{\boldsymbol{P}}_m = \mathbf{D}_m(\hat{\boldsymbol{F}}_{\texttt{4D}}). \tag{4}$$

Timestamp Positional Encoding (TPE). Accurate temporal perception, such as "when" an event occurred and "how long" an action took, is fundamental to 4D VQA. For example, to answer "*What is the average speed of the car?*," even if the MLLM can perceive depth and knows its displacement, it still needs to understand the time duration of the video to compute speed. Incorrect temporal perception can lead to significant errors in acquiring the displacement over the correct time duration, *i.e.*, speed.

We observe that MLLMs struggle with temporal perception when there are no explicit time cues (see the experiments in Sec. 6.3 and Tab. 6). To provide temporal cues, we encode timestamps directly into the MLLM's visual input as positional encodings. That is, for each input frame $\boldsymbol{I}^{(n)}$ from video $\boldsymbol{V}$ that is sampled at time $t^{(n)}$, we add a sinusoidal timestamp positional encoding $\boldsymbol{p}^{(n)} \in \mathbb{R}^{D}$ to the visual features $\mathbf{E}_{\texttt{V}}(\boldsymbol{I}^{(n)})$ before feeding them into the $\mathbf{E}_{\texttt{P}}$, where

$$\boldsymbol{p}^{(n)}[2i] = \sin\left(\frac{t^{(n)}}{T^{\frac{2i}{D}}}\right) \text{ and } \boldsymbol{p}^{(n)}[2i+1] = \cos\left(\frac{t^{(n)}}{T^{\frac{2i}{D}}}\right). \tag{5}$$

Here T is the maximum timescale and i is the index.

4.2. Perceptual 4D Distillation (P4D)

To answer 4D questions, MLLMs must understand not only semantic content but also various aspects of 4D knowledge, such as sub-pixel movements and numeric depth values. For example, to answer "*Is the person moving closer to the camera?*", the MLLM must compare the depth values of the *person* across frames. Recent 3D/4D specialized MLLMs either rely on self-curated training datasets or exploit external models to enhance 3D knowledge. However, both are insufficient for MLLMs to fully achieve 4D understanding. Moreover, introducing external modules results in additional inference costs. Therefore, a mechanism that provides direct supervision on the MLLM's internal 4D perception capabilities without introducing additional modules is desirable.

To this end, we propose our P4D framework. We leverage L4P [2] as the frozen expert 4D perception model (teacher) to

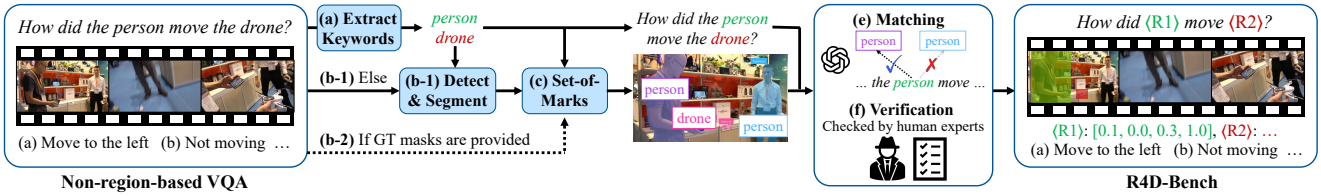


Figure 3. **Curation pipeline of our R4D-Bench.** Given existing non-region 4D VQA benchmarks, we (a) first extract the noun keywords from the question as candidates for objects of interest. (b) Next, if ground truth segmentation masks are provided, we use them for step (d). Otherwise, we use off-the-shelf GroundingDINO [36] and SAM2 [55] to extract segmentation masks for each object of interest. (c) We generate a SoM [77] image for the first frame. (d) We prompt Qwen-2.5VL [52] with the SoM image and the processed question to match the objects referred to in the question with the regions. (e) Finally, the generated matching results are verified by human experts.

transfer its expert representations to our student, 4D-RGPT. We use the same architecture and pre-trained weights as provided in their paper. To ensure comprehensive knowledge transfer, we propose dual-branch distillation: latent distillation and explicit distillation.

Latent Distillation. We start by introducing latent distillation to supervise the MLLM's latent 4D representations, *i.e.*, $\hat{\boldsymbol{F}}_{\texttt{4D}}$, on the latent space. Latent distillation serves as intermediate 4D guidance to the MLLM on the latent space. Specifically, our latent distillation loss $\mathcal{L}_{\texttt{LD}}$ is defined to pull the margin $\Delta_{\texttt{LD}}$ between the latent 4D features from the teacher model $\boldsymbol{F}_{\texttt{4D}}$ and those from the student model $\hat{\boldsymbol{F}}_{\texttt{4D}}$:

$$\mathcal{L}_{\texttt{LD}} = \sum_{n'=1}^{N'} \Delta_{\texttt{LD}}(\boldsymbol{F}_{\texttt{4D}}^{(n')}, \hat{\boldsymbol{F}}_{\texttt{4D}}^{(n')}). \tag{6}$$

Explicit Distillation. On the other hand, we introduce explicit distillation to supervise the MLLM's explicit 4D representations, *i.e.*, $\hat{\boldsymbol{P}}_m$, on the signal space. Explicit distillation provides direct, interpretable supervision to ensure the MLLM captures accurate 4D signals in $\mathcal{M}$. Specifically, our explicit distillation loss $\mathcal{L}_{\texttt{ED}}$ is defined to pull the margin Δ_m between the explicit 4D signals from the teacher model $\boldsymbol{P}_m$ and those from the student model $\hat{\boldsymbol{P}}_m$:

$$\mathcal{L}_{\texttt{ED}} = \sum_{n=1}^{N} \sum_{m \in \mathcal{M}} \lambda_m \Delta_m(\boldsymbol{P}_m^{(n)}, \hat{\boldsymbol{P}}_m^{(n)}), \tag{7}$$

where λ_m describes the loss weights of each m.

Training. We optimize our 4D-RGPT using both SFT and P4D. The overall loss function is a combination of the standard cross-entropy SFT loss $\mathcal{L}_{\texttt{SFT}}$, latent distillation loss $\mathcal{L}_{\texttt{LD}}$, and explicit distillation loss $\mathcal{L}_{\texttt{ED}}$. We train on various 3D / 4D conversation datasets, including RoboFAC [40], SAT [56], VSTI-Bench [15] (the training split), and Wolf [27]. Please refer to the supplementary material for more training details.

5. R4D-Bench

Recently, there has been significant progress in 3D/4D VQA [15, 22, 30, 56, 79, 91]. Several new benchmarks require MLLMs to have depth perception or understand 3D interactions among objects. However, existing benchmarks do not evaluate MLLMs on 4D region-based understanding in complex, real-world scenarios. As shown in Tab. 1, they lack the following critical properties:

- **Lack of Dynamic Scenes**: Most focus on indoor scenes with minimal object interaction or constrained movement, which do not fully capture the complexity of real-world object manipulation and dynamic changes.
- **Lack of Region Prompting**: Region prompts allow controlled and intuitive user queries in VQA. Without this ability, an MLLM's interpretability and usability in practical applications are hindered.

To address these gaps, we introduce ***R4D-Bench*** (see the rightmost example in Fig. 3), a novel benchmark that challenges MLLMs with region-level 4D VQA, where depth and temporal perception are critical.

Task Formulation. Given an input video $\boldsymbol{V} = [\boldsymbol{I}^{(n)}]_{n=1:N}$ of N frames, a region-prompted 4D question $\boldsymbol{Q}$, and a set of region masks $\boldsymbol{M}$ describing the objects of interest in $\boldsymbol{Q}$ in $\boldsymbol{I}^{(1)}$, the task is to respond with the correct or most suitable answer from a set of options.

Benchmark. We curate R4D-Bench based on existing non-region-based 4D VQA benchmarks, *i.e.*, STI-Bench [30] and VLM4D [91]. Our pipeline (Fig. 3) employs a hybrid automated and human-verified process to transform conventional VQ pairs into highly specific region-prompted questions.

The process begins with a non-region-prompted 4D VQA. In the example of Fig. 3, we are given a video of two persons and a drone with the query question "How did the person move the drone?" First, we use Qwen2.5-VL [52] to perform keyword extraction (**Extract**) and identify objects of interest from the query question, *e.g.*, the *person* and the *drone*. While videos from some sources, *e.g.*, DAVIS [50], provide annotations of object masks, other real-world videos lack such detailed annotations. Hence, we leverage state-of-the-art object detection and segmentation models, *i.e.*, GroundingDINO [36] and SAM2 [55], to generate accurate object masks (**Detect & Segment**) for the identified objects of interest. We then apply the segmentation masks with their corresponding keywords onto the video frame to generate

an image with **Set-of-Marks** [77]. This serves as an intermediate and potential portrayal of the region-prompted QA before the final step of checking correctness.

Since the objects of interest can be non-unique (*e.g.*, multiple persons) and segmentation masks can be noisy, ensuring correct association between extracted keywords and found regions is critical. We check correctness with both automated and human-in-the-loop processes. We use Qwen2.5-VL [52] to automatically match the generated region marks to the entities in the question (**Matching**). Finally, human annotators verify and correct any mismatches (**Verification**). We also trim videos to ensure all RoIs are visible in the first frame.

This concludes our region prompting process. The original VQA is transformed into R4D-Bench format, where entities are replaced by region tokens, *e.g.*, "How did $\langle R1 \rangle$ move $\langle R2 \rangle$?" with their corresponding region masks.

Statistics. Our R4D-Bench benchmark consists of 1.4k region-prompted VQAs. Each question is a multiple-choice problem with four to five answer options. The benchmark provides region-prompted challenges to semantic and numerical 4D understanding in both static and dynamic scenes. The static split includes 3 categories: (1) Dimension Measurement; (2) 3D Video Grounding; and (3) Spatial Relation. The dynamic split includes 6 categories: (1) Counting objects; (2) Translational movement; (3) Rotational movement; (4) False Positive detection; (5) Speed & Acceleration estimation; and (6) Displacement & Path Length measurement. We provide more details for each question type in the supplementary material.

6. Experiments

6.1. Experiment Setup

Benchmarks. We evaluate our 4D-RGPT on various 4D VQA benchmarks, including our R4D-Bench and existing ones, *i.e.*, STI-Bench [30], VLM4D-real [91], OmniSpatial [22], MMSI-Bench [79], SAT [56], and VSTI-Bench [15]. Please note that the first four benchmarks are testing-only benchmarks and are disjoint from our training data. Apart from the numerical questions in VSTI-Bench, where we report relative accuracy, we report the multiple-choice accuracy for all other benchmarks.

Comparison Models. We compare our 4D-RGPT with various proprietary MLLMs, *e.g.*, GPT-4o [45], GPT-5 [46], Gemini-2.5-Pro [12]; open-source generalized MLLMs, *e.g.*, Qwen2.5-VL [52]; and recent 3D/4D specialized MLLMs, *e.g.*, SpatialReasoner [42], ViLaSR [74], and SpaceR [47].

Architecture. We select a SOTA open-source generalized MLLM, NVILA-Lite-8B [38], as our MLLM backbone, which uses SigLIP [82] as the $\mathbf{E}_{\mathtt{V}}$ and Qwen2 [62] as the `LLM`. For the 4D perception model $\mathbf{E}_{\mathtt{4D}}$ and $\mathbf{D}_m$, we follow the exact architecture and weights of L4P [2]. We document training setups in the supplementary material.

Table 2. **Evaluation on non-region-level 3D / 4D benchmarks.** We report the average multiple-choice accuracy (↑) on each benchmark. For simplicity, we use the following abbreviations: STI (STI-Bench [30]), V4D (VLM4D-real [91]), MMSI (MMSI-Bench [79]), OS (OmniSpatial [22]), and VSTI (VSTI-Bench [15]).

Methods	STI	V4D	MMSI	OS	SAT	VSTI
Proprietary General MLLMs						
GPT-4o [45]	34.8	60.0	30.3	47.8	57.5	38.2
GPT-5 [46]	39.3	-	40.7	59.9	-	-
Gemini-2.5-Pro [12]	41.4	63.5	36.9	55.4	-	-
Gemini-1.5-Pro [61]	-	-	-	-	64.8	-
Open-source General MLLMs						
InternVL2.5-8B [10]	-	42.4	28.7	-	-	-
Qwen2.5-VL-7B [52]	32.1	43.3	25.9	39.2	-	-
VideoLLaMA3-7B [83]	35.2	46.5	-	-	-	-
LLaVA-Video-7B [86]	-	-	-	-	53.5	-
LLaVA-OneVision-7B [26]	29.0	36.0	24.5	35.7	41.7	-
LLaVA-NeXT-Video-7B [35]	29.9	-	26.8	-	-	40.0
Spatial MLLMs						
VLM-3R-7B [15]	-	-	-	-	-	58.8
LLaVA-Video-7B + SAT [56]	-	-	-	-	63.4	-
ViLaSR-7B [74]	33.4	46.9	30.2	-	-	-
SpatialReasoner-7B [42]	31.0	43.4	22.7	-	-	-
SpaceR-7B [47]	37.0	51.3	28.8	-	47.8	-
NVILA-Lite-8B [38]	33.8	46.5	31.3	37.2	62.0	45.2
4D-RGPT-8B (Ours)	**37.6** +3.8	**52.7** +6.2	**33.3** +2.0	**40.4** +3.2	**64.7** +2.7	**59.1** +13.9

6.2. Main Results

We present the effectiveness of 4D-RGPT in Tab. 2 and Tab. 3, showing improvements over baseline MLLMs.

Non-region-based 4D VQA. In Tab. 2, we evaluate 4D-RGPT on several non-region-level 3D/4D VQA benchmarks, including input modalities of both images and videos. We compare with various state-of-the-art proprietary MLLMs, open-source general MLLMs, and recent 3D/4D MLLMs. 4D-RGPT consistently improves over the baseline NVILA-Lite-8B by a large margin across all benchmarks, especially on VLM4D [91] and VSTI-Bench [15]. Compared to other MLLMs with similar model sizes, 4D-RGPT achieves SOTA performance over open-source MLLMs and competitive performance with GPT-4o [45]. Please note that SpatialReasoner [42], ViLaSR [74], and SpaceR [47] are all trained with RL to further boost accuracy.

R4D-Bench. In Tab. 3, we present quantitative comparisons of our 4D-RGPT on R4D-Bench against other MLLMs. For fair comparison, we use SoM [77] to indicate the regions of interest for all MLLMs. Additionally, for all open-source MLLMs and 4D-RGPT, we use the same number of sampled frames, *i.e.*, 16 frames. We observe that although SpaceR [47] outperforms Qwen2.5-VL [52] in Tab. 2, it falls behind on R4D-Bench, suggesting that SpaceR is highly tuned for non-region VQA and its region understanding is

Table 3. **Evaluation on R4D-Bench.** We report performance on the static split (Sta), the dynamic split (Dyn), and all 9 tasks of R4D-Bench. For simplicity, we abbreviate them as follows: 3D **V**ideo **G**rounding (VG); **D**imension **M**easurement (DM); **S**patial **R**elationship (SR); **R**otational (R); **C**ounting (C); **T**ranslational (T); **F**alse **P**ositive (FP); **S**peed & **A**cceleration (SA); and **D**isplacement & **P**ath Length (DP).

Methods	Avg	Sta	Dyn	VG	DM	SR	R	C	T	FP	SA	DP
Random	23.4	20.0	24.7	20.0	20.0	20.0	25.0	25.0	25.0	25.0	20.0	20.0
GPT-4o [45]	42.8	30.3	47.5	30.7	26.8	43.9	49.1	35.2	51.8	54.1	27.0	10.7
Qwen2.5-VL-7B [52]	40.6	**34.1**	43.1	39.1	25.7	48.8	50.0	38.4	46.6	28.9	45.9	28.6
LLaVA-Video-7B [86]	39.7	26.9	44.6	23.4	28.4	36.6	46.2	30.2	50.4	33.6	48.6	35.7
ViLaSR-7B [74]	39.6	31.5	42.6	34.4	24.6	48.8	46.2	42.8	51.3	3.7	43.2	17.9
SpatialReasoner-7B [42]	38.3	31.2	41.0	35.4	25.7	36.6	43.4	37.1	49.3	11.9	32.4	17.9
SpaceR-7B [47]	37.0	26.2	41.1	30.7	18.0	41.5	47.2	40.3	43.8	25.9	51.4	21.4
NVILA-Lite-8B [38]	37.9	29.1	41.3	33.9	20.2	46.3	41.5	39.6	41.9	40.7	45.9	32.1
4D-RGPT-8B (Ours)	**42.2**	32.9	**45.7**	35.1	26.3	52.2	43.1	40.1	48.7	40.2	50.9	38.9
	+4.3	**+3.8**	**+4.4**	**+1.2**	**+6.1**	**+5.9**	**+1.6**	**+0.5**	**+6.8**	-0.5	**+5.0**	**+6.8**

Table 4. **Alternative strategies for 4D VQA.** We compare P4D with direct SFT (*4D-SFT*) and straightforward designs of incorporating $\boldsymbol{F}_{4D}$ from the 4D perception model, *i.e.*, *4D-Concat* and *4D-PE*. For simplicity, we use the same abbreviations as in Tab. 3 and STI for STI-Bench [30].

Methods	$\boldsymbol{F}_{4D}$	STI	R4D-Bench		
			Avg	Sta	Dyn
Zero-shot	✗	33.8	37.9	29.1	41.3
4D-SFT	✗	34.7	40.1	32.2	43.8
4D-Concat	✓	34.8	39.5	30.6	42.9
4D-PE	✓	31.3	36.0	26.6	39.5
Ours (*P4D*)	✓	**37.6**	**42.2**	**32.9**	**45.7**

Q: At 2.0 sec, what is the instantaneous acceleration of ⟨R1⟩ (m/s^2)?

✗**GPT**: −0.40 ✗**Qwen**: −0.40 ✗**NVILA**: −0.60 ✓**Ours**: −0.70

Q: How many times did ⟨R1⟩ hit ⟨R2⟩ upward?

✗**GPT**: 1 ✗**Qwen**: 0 ✗**NVILA**: 1 ✓**Ours**: 4

Figure 4. **VQA comparison among baseline MLLMs and 4D-RGPT on R4D-Bench.** For the baseline MLLMs, we use GPT-4o-20241120 [45], Qwen-2.5VL-7B-Instruct [52], and NVILA-Lite-8B [38]. We note that the regions labeled with (*) or (*) are not provided in R4D-Bench; they are visualized for readability.

weakened. Overall, 4D-RGPT achieves the best performance among all open-source MLLMs by at least 1.6% on average and 2.6% on the dynamic split.

In Fig. 4, we showcase two cases of 4D-RGPT against other MLLMs on R4D-Bench. In both cases, the regions of interest are constantly moving. Only 4D-RGPT effectively perceives the 4D dynamics and provides the correct answers.

6.3. Ablation Studies

To justify our various designs, we conduct extensive ablation studies and analysis. For most experiments in this subsection, we report results on STI-Bench [30] and the static and dynamic question subsets of R4D-Bench. Without specific notes, we use the same training data, and all other components are kept identical unless specified.

Alternative Strategies. Besides P4D, there are other strategies to utilize 4D conversation data or the latent feature $\boldsymbol{F}_{4D}$ from the 4D perception models to enhance MLLMs' 4D understanding. First, denoted as *4D-SFT*, we apply solely SFT to the entire MLLM without access to $\boldsymbol{F}_{4D}$. Additionally, there are two straightforward ways to leverage $\boldsymbol{F}_{4D}$. Denoted as *4D-Concat*, we directly concatenate $\boldsymbol{F}_{4D}$ with the 2D visual features $\boldsymbol{E}_V(\boldsymbol{V})$. We note that this requires additional training on $\mathbf{E}_P$ as the dimension differs from the original visual features. On the other hand, denoted as *4D-PE*, we project $\boldsymbol{F}_{4D}$ to positional encodings (PE) for the visual features, similar to the spatial PE proposed in SR-3D [11].

As shown in Tab. 4, apart from *4D-PE*, both *4D-SFT* and *4D-Concat* improve over the *Zero-shot* baseline. However, they all fall short compared to P4D. Moreover, *4D-Concat* and *4D-PE* require additional inference costs as they need to compute $\boldsymbol{F}_{4D}$ for each input during inference. In comparison, P4D requires solely training-only 4D perception modules, making 4D-RGPT as efficient as *Zero-shot* during inference.

Perceptual 4D Distillation. To validate the effectiveness of P4D, we experiment with various distillation strategies used in latent distillation ($\mathcal{L}_{LD}$ in Eq. (6)) and explicit distillation ($\mathcal{L}_{ED}$ in Eq. (7)). In Tab. 5, we ablate different combinations of distillation on $\hat{\boldsymbol{F}}_{4D}$ and $\hat{\boldsymbol{P}}_m$.

We first observe that applying $\mathcal{L}_{LD}$ alone (*LD-only*) improves the performance over the *Zero-shot* baseline by 2.3% on R4D-Bench. For $\mathcal{L}_{ED}$, adding more $m \in \mathcal{M}$ incrementally improves the performance steadily, with $m = \texttt{depth}$ and $m = \texttt{flow}$ being the most effective ones (see *LD+D* and *LD+D+F*). While $\mathcal{L}_{ED}$ alone (*ED-only*) also improves the performance on R4D-Bench by 1.9%, combining both (*LD+ED*) achieves the best average performance, showing

Table 5. **Analysis of 4D modalities in P4D.** We ablate the effectiveness of different combinations of distillation in latent distillation (LD) on $\hat{\boldsymbol{F}}_{4D}$ and explicit distillation (ED) on $\hat{\boldsymbol{P}}_m$. For simplicity, we use the same abbreviations as Tab. 4 and **D**epth (D), **F**low (F), **M**otion (M), and **C**amray (C) for each $m \in \mathcal{M}$.

Methods	$\hat{\boldsymbol{F}}_{4D}$	$\hat{\boldsymbol{P}}_m$				STI	R4D-Bench		
		D	F	M	C		Avg	Sta	Dyn
Zero-shot	✗	✗	✗	✗	✗	33.8	37.9	29.1	41.3
LD-Only	✓	✗	✗	✗	✗	34.2	40.2	32.0	43.3
LD+D	✓	✓	✗	✗	✗	33.4	40.8	32.5	44.0
LD+D+F	✓	✓	✓	✗	✗	36.2	41.9	**33.1**	45.3
LD+D+F+M	✓	✓	✓	✓	✗	36.5	42.0	**33.1**	45.4
ED-Only	✗	✓	✓	✓	✓	35.4	39.8	31.5	42.9
Ours (*LD+ED*)	✓	✓	✓	✓	✓	**37.6**	**42.2**	32.9	**45.7**

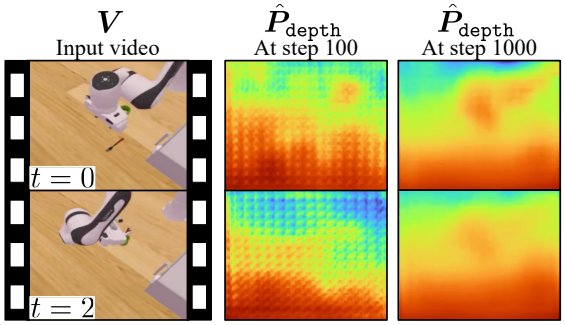


Figure 5. **Predicted depth maps at different training steps.** We visualize the progress of $\hat{\boldsymbol{P}}_{\texttt{depth}}$ throughout training.

Table 6. **Ablation studies on explicit temporal cues.** We experiment without and with different choices of explicit time cues. For simplicity, we use the same abbreviations as Tab. 4.

Methods	Time cues	STI	R4D-Bench		
			Avg	Sta	Dyn
Zero-shot	✗	33.8	37.9	29.1	41.3
P4D	✗	34.8	41.0	31.8	44.5
P4D+mark	marks	35.1	41.1	31.5	44.7
P4D+prompt	prompts	36.1	41.5	32.1	45.0
Ours (*P4D+TPE*)	TPE	**37.6**	**42.2**	**32.9**	**45.7**

the complementary benefits of both LD and ED.

4D Perception Visualization. In Fig. 5, we visualize the progress of how 4D-RGPT learns to extract 4D signals through P4D. We show a video from our training set [40] with extracted $\hat{\boldsymbol{P}}_{\texttt{depth}}$ at various steps. $\hat{\boldsymbol{P}}_{\texttt{depth}}$ is barely meaningful at first but gradually captures the 3D structure of the scene as training proceeds. This indicates that P4D successfully distills 4D perception capabilities into 4D-RGPT.

Timestamp Positional Encoding (TPE). MLLMs often struggle with temporal perception when no explicit time cues are provided. We conduct a controlled toy experiment to validate this observation by curating a simple benchmark with VQAs that require temporal perception, such as "*How many seconds have passed in the input video?*" We observe that NVILA-Lite-8B [38] is naively guessing the answers, resulting in accuracy close to random guessing. This problem is further exacerbated by the inconsistency among multiple sources of data with different frame rates. We detail the toy experiment in the supplementary material.

Without introducing additional modules, we test two simple solutions to provide explicit temporal cues to MLLMs. First, denoted as *P4D+mark*, we add explicit time marks similar to SoM [77] on each $\boldsymbol{I}^{(n)}$, such as burned-in text showing the timestamp, *e.g.*, "$t^{(n)}$ s" Second, denoted as *P4D+prompt*, we add explicit time information in $\boldsymbol{Q}$, such as "*The following video frames are sampled from a video 19 seconds long and recorded at 30 frames per second.*"

Both *P4D+mark* and *P4D+prompt*, as shown in Tab. 6, can improve 4D VQA performance. However, they require additional data preprocessing, distract MLLMs from the main visual and textual content, and do not generalize well to region-level settings, *i.e.*, R4D-Bench. Our *P4D+TPE* consistently improves performance across both benchmarks, as shown in the last row of Tab. 6.

Table 7. **Ablation studies on different training designs in 4D-RGPT.** We ablate different training designs on whether each module is trainable and whether to use LoRA [20]. For simplicity, we use the same abbreviations as Tab. 4.

Methods	Trainable			STI	R4D-Bench		
	$\mathbf{E}_{\texttt{V}}$	$\mathbf{E}_{\texttt{P}}$	LLM		Avg	Sta	Dyn
Zero-shot	✗	✗	✗	33.8	37.9	29.1	41.3
Tune-All	✓	✓	✓	34.7	38.8	30.1	42.1
Tune-V	✓	✗	✗	32.3	35.8	27.3	39.0
Tune-P	✗	✓	✗	34.3	38.6	29.8	42.0
Tune-LLM	✗	✗	✓	35.4	40.5	32.2	43.7
Tune-LLM-LoRA	✗	✗	LoRA	37.0	41.1	**33.0**	44.2
Tune-P+LLM-LoRA	✗	✓	LoRA	36.5	41.4	32.8	44.7
Ours (*Tune-P+LLM*)	✗	✓	✓	**37.6**	**42.2**	32.9	**45.7**

Architecture Design. In Tab. 7, we ablate different designs on whether $\mathbf{E}_{\texttt{V}}$, $\mathbf{E}_{\texttt{P}}$, or LLM is trainable or frozen. Our *Tune-P+LLM* achieves the best performance by tuning both $\mathbf{E}_{\texttt{P}}$ and LLM, while keeping $\mathbf{E}_{\texttt{V}}$ frozen. This is likely because $\mathbf{E}_{\texttt{P}}$ requires finetuning for TPE and P4D works best on LLM.

7. Conclusion

We show that existing MLLMs struggle with region-level 4D VQA due to not fully perceiving 4D information. Without incurring additional inference cost, our 4D-RGPT effectively improves MLLMs' 4D perception by learning from a 4D perception model via a novel distillation framework, P4D. Additionally, we introduce a proper benchmark, R4D-Bench, for this domain, contributing to region-level 4D VQA. Extensive experiments confirm the effectiveness of our approach on both non-region-level and region-level 4D VQA.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 2

[2] Abhishek Badki, Hang Su, Bowen Wen, and Orazio Gallo. L4P: Low-level 4D vision perception unified. *arXiv preprint arXiv:2502.13078*, 2025. 3, 4, 6

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 2

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proc. CVPR*, 2020. 13

[5] Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. ViP-LLaVA: Making large multimodal models understand arbitrary visual prompts. In *Proc. CVPR*, 2024. 2

[6] Gongwei Chen, Leyang Shen, Rui Shao, Xiang Deng, and Liqiang Nie. LION: Empowering multimodal large language model with dual-level visual knowledge. In *Proc. CVPR*, 2024. 2

[7] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023. 2

[8] Pingyi Chen, Yujing Lou, Shen Cao, Jinhui Guo, Lubin Fan, Yue Wu, Lin Yang, Lizhuang Ma, and Jieping Ye. SD-VLM: Spatial measuring and understanding with depth-encoded vision-language models. In *Proc. NeurIPS*, 2025. 2

[9] Yiming Chen, Zekun Qi, Wenyao Zhang, Xin Jin, Li Zhang, and Peidong Liu. Reasoning in space via grounding in the world. *arXiv preprint arXiv:2510.13800*, 2025. 2

[10] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024. 6

[11] An-Chieh Cheng, Yang Fu, Yukang Chen, Zhijian Liu, Xiaolong Li, Subhashree Radhakrishnan, Song Han, Yao Lu, Jan Kautz, Pavlo Molchanov, et al. 3d aware region prompted vision language model. *arXiv preprint arXiv:2509.13317*, 2025. 2, 7

[12] 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. 2, 6

[13] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. CVPR*, 2017. 13

[14] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 1, 2, 13

[15] Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Huaizhi Qu, Dilin Wang, Zhicheng Yan, et al. VLM-3R: Vision-language models augmented with instruction-aligned 3d reconstruction. *arXiv preprint arXiv:2505.20279*, 2025. 2, 3, 5, 6, 13, 17, 18, 19

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. AISTATS*, 2010. 13

[17] Yanbin Hao, Diansong Zhou, Zhicai Wang, Chong-Wah Ngo, and Meng Wang. PosMLP-Video: Spatial and temporal relative position encoding for efficient video recognition. *IJCV*, 2024. 2

[18] D Hendrycks. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. 13

[19] Miran Heo, Min-Hung Chen, De-An Huang, Sifei Liu, Subhashree Radhakrishnan, Seon Joo Kim, Yu-Chiang Frank Wang, and Ryo Hachiuma. Omni-RGPT: Unifying image and video region-level understanding via token marks. In *Proc. CVPR*, 2025. 2

[20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-rank adaptation of large language models. In *Proc. ICLR*, 2022. 8

[21] Xiaohu Huang, Jingjing Wu, Qunyi Xie, and Kai Han. Mllms need 3d-aware representation supervision for scene understanding. In *Proc. NeurIPS*, 2025. 2

[22] Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. OmniSpatial: Towards comprehensive spatial reasoning benchmark for vision language models. *arXiv preprint arXiv:2506.03135*, 2025. 2, 3, 5, 6, 17

[23] Dohwan Ko, Sihyeon Kim, Yumin Suh, Minseo Yoon, Manmohan Chandraker, Hyunwoo J Kim, et al. ST-VLM: Kinematic instruction tuning for spatio-temporal reasoning in vision-language models. *arXiv preprint arXiv:2503.19355*, 2025. 2

[24] Byung-Kwan Lee, Beomchan Park, Chae Won Kim, and Yong Man Ro. CoLLaVO: Crayon large language and vision mOdel. In *Proc. ACL*, 2024. 2

[25] Xuanyu Lei, Zonghan Yang, Xinrui Chen, Peng Li, and Yang Liu. Scaffolding coordinates to promote vision-language coordination in large multi-modal models. In *Proc. ACL*, 2025. 2

[26] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *TMLR*, 2025. 6

[27] Boyi Li, Ligeng Zhu, Ran Tian, Shuhan Tan, Yuxiao Chen, Yao Lu, Yin Cui, Sushant Veer, Max Ehrlich, Jonah Philion, et al. Wolf: Dense video captioning with a world summarization framework. *TMLR*, 2025. 5, 13, 14, 18, 19

[28] Hongxing Li, Dingming Li, Zixuan Wang, Yuchen Yan, Hang Wu, Wenqi Zhang, Yongliang Shen, Weiming Lu, Jun Xiao, and Yueting Zhuang. SpatialLadder: Progressive training for

spatial reasoning in vision-language models. *arXiv preprint arXiv:2510.08531*, 2025. 2

[29] Pengteng Li, Pinhao Song, Wuyang Li, Weiyu Guo, Huizai Yao, Yijie Xu, Dugang Liu, and Hui Xiong. See&trek: Training-free spatial prompting for multimodal large language model. In *Proc. NeurIPS*, 2025. 2

[30] Yun Li, Yiming Zhang, Tao Lin, XiangRui Liu, Wenxiao Cai, Zheng Liu, and Bo Zhao. STI-Bench: Are MLLMs ready for precise spatial-temporal world understanding? In *Proc. ICCV*, 2025. 2, 3, 5, 6, 7, 15, 16, 17

[31] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. VILA: On pre-training for visual language models. In *Proc. CVPR*, 2024. 2

[32] Weifeng Lin, Xinyu Wei, Ruichuan An, Peng Gao, Bocheng Zou, Yulin Luo, Siyuan Huang, Shanghang Zhang, and Hongsheng Li. Draw-and-understand: Leveraging visual prompts to enable mllms to comprehend what you want. In *Proc. ICLR*, 2025. 2

[33] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Proc. NeurIPS*, 2023. 2

[34] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proc. CVPR*, 2024. 2

[35] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, ocr, and world knowledge, 2024. 6

[36] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *Proc. ECCV*, 2024. 5, 14

[37] Zikang Liu, Longteng Guo, Yepeng Tang, Tongtian Yue, Junxian Cai, Kai Ma, Qingbin Liu, Xi Chen, and Jing Liu. Vrope: Rotary position embedding for video large language models. In *Proc. EMNLP*, 2025. 2

[38] Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, et al. NVILA: Efficient frontier visual language models. In *Proc. CVPR*, 2025. 2, 6, 7, 8, 13

[39] Jinghui Lu, Haiyang Yu, Yanjie Wang, Yongjie Ye, Jingqun Tang, Ziwei Yang, Binghong Wu, Qi Liu, Hao Feng, Han Wang, et al. A bounding box is worth one token-interleaving layout and text in a large language model for document understanding. In *ACL Findings*, 2025. 2

[40] Weifeng Lu, Minghao Ye, Zewei Ye, Ruihan Tao, Shuo Yang, and Bo Zhao. RoboFAC: A comprehensive framework for robotic failure analysis and correction. *arXiv preprint arXiv:2505.12224*, 2025. 5, 8, 13, 14, 18, 19

[41] Chuofan Ma, Yi Jiang, Jiannan Wu, Zehuan Yuan, and Xiaojuan Qi. Groma: Localized visual tokenization for grounding multimodal large language models. In *Proc. ECCV*, 2024. 2

[42] Wufei Ma, Yu-Cheng Chou, Qihao Liu, Xingrui Wang, Celso M de Melo, Jianwen Xie, and Alan Yuille. SpatialReasoner: Towards explicit and generalizable 3d spatial reasoning. In *Proc. NeurIPS*, 2025. 2, 6, 7

[43] Wufei Ma, Luoxin Ye, Celso M de Melo, Alan Yuille, and Jieneng Chen. Spatialllm: A compound 3d-informed design towards spatially-intelligent large multimodal models. In *Proc. CVPR*, 2025. 2

[44] Yunze Man, De-An Huang, Guilin Liu, Shiwei Sheng, Shilong Liu, Liang-Yan Gui, Jan Kautz, Yu-Xiong Wang, and Zhiding Yu. ARGUS: Vision-centric reasoning with grounded chain-of-thought. In *Proc. CVPR*, 2025. 2

[45] OpenAI. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 1, 2, 6, 7, 20, 21

[46] OpenAI. Gpt-5. https://openai.com/chatgpt, 2025. Large language model. 1, 2, 6

[47] Kun Ouyang, Yuanxin Liu, Haoning Wu, Yi Liu, Hao Zhou, Jie Zhou, Fandong Meng, and Xu Sun. Spacer: Reinforcing mllms in video spatial reasoning. *arXiv preprint arXiv:2504.01805*, 2025. 2, 6, 7

[48] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, Qixiang Ye, and Furu Wei. Grounding multimodal large language models to the world. In *Proc. ICLR*, 2024. 2

[49] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proc. CVPR*, 2016. 14

[50] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *arXiv:1704.00675*, 2017. 5, 14

[51] Shraman Pramanick, Guangxing Han, Rui Hou, Sayan Nag, Ser-Nam Lim, Nicolas Ballas, Qifan Wang, Rama Chellappa, and Amjad Almahairi. Jack of all tasks master of many: Designing general-purpose coarse-to-fine vision-language model. In *Proc. CVPR*, 2024. 2

[52] Alibaba Group Qwen Team. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. 2, 5, 6, 7, 14, 16

[53] Gorjan Radevski, Dusan Grujicic, Matthew Blaschko, Marie-Francine Moens, and Tinne Tuytelaars. Multimodal distillation for egocentric action recognition. In *Proc. ICCV*, 2023. 2

[54] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proc. ICCV*, 2021. 13

[55] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. In *Proc. ICLR*, 2025. 5, 14

[56] Arijit Ray, Jiafei Duan, Reuben Tan, Dina Bashkirova, Rose Hendrix, Kiana Ehsani, Aniruddha Kembhavi, Bryan A Plummer, Ranjay Krishna, Kuo-Hao Zeng, et al. SAT: Spatial aptitude training for multimodal language models. In *Proc. COLM*, 2025. 2, 3, 5, 6, 14, 19

[57] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. Timechat: A time-sensitive multimodal large language model for long video understanding. In *Proc. CVPR*, 2024. 2

[58] Yifan Shen, Yuanzhe Liu, Jingyuan Zhu, Xu Cao, Xiaofeng Zhang, Yixiao He, Wenming Ye, James Matthew Rehg, and

Ismini Lourentzou. Fine-grained preference optimization improves spatial reasoning in vlms. In *Proc. NeurIPS*, 2025. 2

[59] Yumeng Shi, Quanyu Long, Yin Wu, and Wenya Wang. Causality matters: How temporal information emerges in video language models. *arXiv preprint arXiv:2508.11576*, 2025. 2

[60] Peiwen Sun, Shiqiang Lang, Dongming Wu, Yi Ding, Kaituo Feng, Huadai Liu, Zhen Ye, Rui Liu, Yun-Hui Liu, Jianan Wang, et al. Spacevista: All-scale visual spatial reasoning from mm to km. *arXiv preprint arXiv:2510.09606*, 2025. 2

[61] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 2, 6

[62] Qwen Team et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024. 6

[63] Yunjie Tian, Tianren Ma, Lingxi Xie, Jihao Qiu, Xi Tang, Yuan Zhang, Jianbin Jiao, Qi Tian, and Qixiang Ye. ChatterBox: Multi-round multimodal referring and grounding. In *Proc. AAAI*, 2025. 2

[64] Maxim Tkachenko, Mikhail Malyuk, Andrey Holmanyuk, and Nikolai Liubimov. Label Studio: Data labeling software, 2020-2025. Open source software available from https://github.com/HumanSignal/label-studio. 15

[65] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 2

[66] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2

[67] 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 *Proc. CVPR*, 2023. 13

[68] Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Lu Yuan, and Yu-Gang Jiang. Masked video distillation: Rethinking masked feature modeling for self-supervised video representation learning. In *Proc. CVPR*, 2023. 2

[69] Weiyun Wang, Yiming Ren, Haowen Luo, Tiantong Li, Chenxiang Yan, Zhe Chen, Wenhai Wang, Qingyun Li, Lewei Lu, Xizhou Zhu, et al. The All-Seeing project v2: Towards general relation comprehension of the open world. In *Proc. ECCV*, 2024. 2

[70] Weiyun Wang, Min Shi, Qingyun Li, Wenhai Wang, Zhenhang Huang, Linjie Xing, Zhe Chen, Hao Li, Xizhou Zhu, Zhiguo Cao, et al. The all-seeing project: Towards panoptic visual recognition and understanding of the open world. In *Proc. ICLR*, 2024. 2

[71] 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. 13

[72] Sangmin Woo, Kang Zhou, Yun Zhou, Shuai Wang, Sheng Guan, Haibo Ding, and Lin Lee Cheong. Black-box visual prompt engineering for mitigating object hallucination in large vision language models. In *Proc. NAACL*, 2025. 2

[73] Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. In *Proc. NeurIPS*, 2025. 2

[74] Junfei Wu, Jian Guan, Kaituo Feng, Qiang Liu, Shu Wu, Liang Wang, Wei Wu, and Tieniu Tan. Reinforcing spatial reasoning in vision-language models with interwoven thinking and visual drawing. In *Proc. NeurIPS*, 2025. 2, 6, 7

[75] Runsen Xu, Weiyao Wang, Hao Tang, Xingyu Chen, Xiaodong Wang, Fu-Jen Chu, Dahua Lin, Matt Feiszli, and Kevin J Liang. Multi-SpatialMLLM: Multi-frame spatial understanding with multi-modal large language models. *arXiv preprint arXiv:2505.17015*, 2025. 2

[76] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. 1, 2

[77] Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*, 2023. 2, 5, 6, 8, 15

[78] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proc. CVPR*, 2025. 13

[79] Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, et al. MMSI-Bench: A benchmark for multi-image spatial intelligence. *arXiv preprint arXiv:2505.23764*, 2025. 2, 3, 5, 6, 17

[80] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3d indoor scenes. In *Proc. ICCV*, 2023. 13

[81] Xiangyu Zeng, Kunchang Li, Chenting Wang, Xinhao Li, Tianxiang Jiang, Ziang Yan, Songze Li, Yansong Shi, Zhengrong Yue, Yi Wang, et al. Timesuite: Improving mllms for long video understanding via grounded tuning. In *Proc. ICLR*, 2025. 2

[82] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proc. ICCV*, 2023. 6, 13

[83] 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. 6

[84] Jiahui Zhang, Yurui Chen, Yanpeng Zhou, Yueming Xu, Ze Huang, Jilin Mei, Junhui Chen, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, et al. From flatland to space: Teaching vision-language models to perceive and reason in 3d. In *Proc. NeurIPS*, 2025. 2

[85] Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Yu Liu, Kai Chen, and Ping Luo. GPT4RoI: Instruction tuning large language model on region-of-interest. In *Proc. ECCV Workshop*, 2024. 2

[86] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024. 6, 7

[87] Liang Zhao, En Yu, Zheng Ge, Jinrong Yang, Haoran Wei, Hongyu Zhou, Jianjian Sun, Yuang Peng, Runpei Dong, Chunrui Han, and Xiangyu Zhang. ChatSpot: Bootstrapping multimodal llms via precise referring instruction tuning. In *Proc. IJCAI*, 2024. 2

[88] Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Learning from videos for 3d world: Enhancing mllms with 3d vision geometry priors. In *Proc. NeurIPS*, 2025. 2

[89] Hanyu Zhou and Gim Hee Lee. LLaVA-4D: Embedding spatiotemporal prompt into lmms for 4d scene understanding. *arXiv preprint arXiv:2505.12253*, 2025. 2

[90] Honglu Zhou, Xiangyu Peng, Shrikant Kendre, Michael S Ryoo, Silvio Savarese, Caiming Xiong, and Juan Carlos Niebles. Strefer: Empowering video llms with space-time referring and reasoning via synthetic instruction data. In *Proc. ICCV*, 2025. 2

[91] Shijie Zhou, Alexander Vilesov, Xuehai He, Ziyu Wan, Shuwang Zhang, Aditya Nagachandra, Di Chang, Dongdong Chen, Eric Xin Wang, and Achuta Kadambi. VLM4D: Towards spatiotemporal awareness in vision language models. In *Proc. ICCV*, 2025. 2, 3, 5, 6, 15, 17

[92] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *Proc. ICLR*, 2024. 2

4D-RGPT: Toward Region-level 4D Understanding via Perceptual Distillation

Supplementary Material

The appendix is organized as follows:

- In Sec. A1, we provide implementation and training details for P4D and 4D-RGPT, including model architecture, training data, computational resources, and loss functions.
- In Sec. A2, we provide the detailed design of R4D-Bench, including the nine question categories and dataset curation process.
- In Sec. A3, we provide additional experimental results, including results with other NVILA variants, analysis of temporal perception capabilities, training data mixture, more qualitative results, and visualizations.

A1. Additional Details

A1.1. Model Architecture

MLLM. As mentioned in Sec. 6.1, we use NVILA-Lite-8B [38] as our base MLLM in the main experiments. NVILA is a unified open-sourced MLLM family that tackles both image and video understanding.

Considering the tradeoff between performance and inference efficiency, there are two groups of NVILA variants, *e.g.*, NVILA (Base) and NVILA-Lite, where the latter is more efficient. For example, NVILA-Lite uses a 3×3 downsampling kernel in $\mathbf{E}_{\mathtt{P}}$ while NVILA (Base) uses 2×2. We select NVILA-Lite as our base MLLM due to its competitive performance and higher efficiency.

For all NVILA variants, we use their open-sourced weights from `HuggingFace` [71]. Specifically, we use the following checkpoints:

- `Efficient-Large-Model/NVILA-Lite-8B`;
- `Efficient-Large-Model/NVILA-Lite-15B`;

For the vision encoder (tower) $\mathbf{E}_{\mathtt{V}}$, they use SigLIP [82], specifically `siglip-so400m-patch14-384`. For the multi-modal projector $\mathbf{E}_{\mathtt{P}}$, they use a 2-layer MLP with a hidden dimension of 4,608.

4D Perception Model. As mentioned in Sec. 6.1, we use L4P [38] as our 4D perception model. A 40-layer ViT-based video encoder from VideoMAEv2 [67] is adopted for $\mathbf{E}_{\mathtt{4D}}$, and DPT [54] is adopted for each $\mathbf{D}_m$ where $m \in \mathcal{M}$. Each $\mathbf{D}_m$ has the same architecture but different output channels depending on the target modality. As mentioned in Sec. 3, the output channels are $1, 2, 1, 6$ for the `depth`, `flow`, `motion`, `camray`, respectively. L4P has 1,337M parameters and takes approximately 300ms to process a 16-frame video on an A100 GPU. Since L4P is only required during training, its 4D signals can be pre-computed and stored offline, adding no inference overhead to 4D-RGPT.

Figure A1. **An example from VSTI-Bench [15] training data**. The corresponding conversation is as follows: (1) *User*: "These are frames of a video. Approximately how far (in meters) did the camera move between frame 14 and frame 20 of 32? Please answer the question using a single word or phrase."; (2) *GPT*: "1.6".

4D-RGPT. In 4D-RGPT, we design a lightweight 4D perception decoder $\mathbf{D}_{\mathtt{4DP}}$ to efficiently extract 4D perceptual latent from `LLM`'s hidden states. It is a 3-layer MLP with a hidden dimension of 2,560. We use GELU [18] as the activation function between each layer. For initialization, we use Xavier initialization [16] for all weights and zeros for all biases. Additionally, 4D-RGPT employs Temporal Positional Encoding (TPE) to enhance the temporal understanding of the model. For TPE (Eq. (5)), we use $T = 10,000$.

A1.2. Data Mixture

We provide more details about the training data mixture used in our training.

VSTI-Bench [15] is a new dataset built upon VSI-Bench [78]. While VSI-Bench focuses on the spatial understanding of static 3D scenes, VSTI-Bench further investigates the spatial-temporal understanding of how spatial relations evolve over time. We use only the training set of VSTI-Bench and do not use the VSI-Bench. The videos are sourced from ScanNet [13] and ScanNet++ [80]. The training set contains roughly 1.2k unique videos and 130k QA pairs. A training sample is shown in Fig. A1.

Wolf [27] is a large-scale video captioning dataset with high-quality captions generated by VLMs. Wolf provides detailed captions across three domains: autonomous driving, general scenes, and robotics. We use the NuScenes [4] portion of Wolf, *i.e.*, the autonomous driving domain. We use Llama-3.1-70B-Instruct [14] with the template-based text prompts to generate question-answer pairs based on these captions, creating conversational data suitable for 4D VQA training. The training set contains roughly 5k unique videos and 15k QA pairs. A training sample is shown in Fig. A2.

RoboFAC [40] is a large-scale dataset for semantic understanding of robotic arm videos, including a training split with simulated robotic arm videos involving various actions. We adopt it into our training data mixture due to its stable

CVPR #22

Figure A2. **An example from Wolf [27] training data**. The corresponding conversation is as follows: (1) *User*: "What traffic participants are around the ego-centric vehicle in the video? Please count and list all of them."; (2) *GPT*: "1 car is near the ego vehicle's expected path. 1 traffic cone is in the ego vehicle's lane and potentially blocking the ego vehicle. 4 barriers are in the ego vehicle's lane and potentially blocking the ego vehicle.".

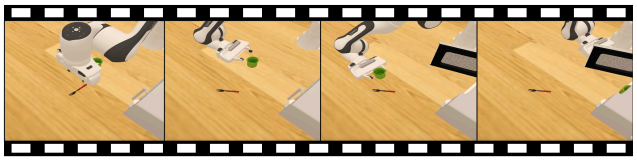


Figure A3. **An example from RoboFAC [40] training data**. The corresponding conversation is as follows: (1) *User*: "From the video, what task is the robot engaged in?"; (2) *GPT*: "Put the fork in the cup and put them in the microwave.".

(a) First frame. (b) Second frame.

Figure A4. **An example from SAT [56] training data.** The corresponding conversation is as follows: (1) *User*: "Were any of the objects in the initial frame that you can still see in the second frame moved from their original positions? Options: ['green tapered square potted houseplant was moved right and towards the camera in the first frame', 'green tapered square potted houseplant was moved left and away from the camera in the first frame']"; (2) *GPT*: "green tapered square potted houseplant was moved right and towards the camera in the first frame.".

camera views with limited background variations but rich robotic arm movements. It contains roughly 10k unique videos and 65k conversations. A training sample is shown in Fig. A3.

SAT [56] is an image-based VQA dataset. Though it is image-based, we consider it helpful for 4D VQA training due to its relevance on dynamic scene understanding across images. The training set contains roughly 190k unique simulated images and 170k QA pairs. A training sample is shown in Fig. A4.

A1.3. Training Details

Our training starts from the pre-trained NVILA weights with an initial learning rate of $1e{-5}$. We use a cosine learning rate scheduler with a warmup ratio of 0.03. We train on a multi-node cluster comprising 8 nodes. Each node has NVIDIA A100-SXM4-80GB GPUs and an AMD EPYC 7J13 64-Core Processor CPU. The total batch size is 1,024. We train for 5 epochs over approximately 12 hours.

Losses. As mentioned in Sec. 4.2, we train our model with both SFT loss $\mathcal{L}_{\texttt{SFT}}$ and P4D loss, *i.e.*, latent distillation loss $\mathcal{L}_{\texttt{LD}}$ and explicit distillation loss $\mathcal{L}_{\texttt{ED}}$. Specifically, our total loss is

$$\mathcal{L} = \mathcal{L}_{\texttt{SFT}} + \alpha\mathcal{L}_{\texttt{LD}} + \beta\mathcal{L}_{\texttt{ED}}, \tag{A8}$$

where α and β are hyperparameters to balance the three loss terms. We set $\alpha = 0.5$ and $\beta = 0.1$.

In Eq. 6, we set $\Delta_{\texttt{LD}}$ to be the Smooth-L1 distance function. In Eq. 7, we set each Δ_m to be the Smooth-L1 distance function and λ_m to be $1.0, 0.1, 0.05, 0.05$ for $m \in \{\texttt{depth}, \texttt{flow}, \texttt{motion}, \texttt{camray}\}$, respectively.

A2. R4D-Bench

We provide more details about R4D-Bench, including the 9 question categories (Sec. A2.2) and dataset curation process (Sec. A2.1).

A2.1. Dataset Curation

To construct R4D-Bench, we develop a hybrid automated and human-in-the-loop process that converts existing non-region-based 4D VQA benchmarks into region-based format. Recall Sec. 5 and Fig. 3, our curation process consists of the following stages.

(a) Keyord Extraction. Given a question $\boldsymbol{Q}$ and the first frame $\boldsymbol{I}^{(1)}$ of a video, we first identify the objects mentioned in $\boldsymbol{Q}$. We employ Qwen2.5-VL-32B-Instruct [52] to parse the question and extract object references. The model is given the following system prompt.

> **Task:** You will receive (1) an RGB image (the first frame of a video) and (2) a natural-language question about objects in the image.
> **Instructions:** Identify the object(s) mentioned in the question and wrap them with angle brackets <>. Do not change any other part of the text. If no object matches, return the original question.
> **Example:**
> *Input:* "What is the teacher right hand holding?"
> *Output:* "What is the `<teacher>` right hand holding?"

(b) Detect & Segment. If the segmentation masks of the identified objects are annotated in the original source, *e.g.*, DAVIS [49, 50], we skip this stage. Otherwise, we extract the 2D bounding boxes and segmentation masks for each identified object using a combination of GroundingDINO [36] and SAM2 [55]. Specifically, we use GrondingDINO (`IDEA-Research/grounding-dino-base` from

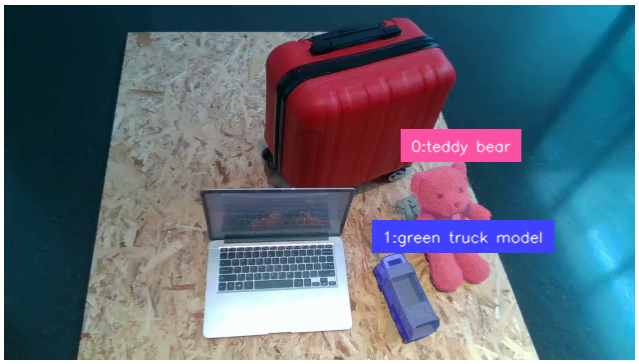


Figure A5. **An example of SoM visual input in R4D-Bench.** We apply SoM [77] on $\boldsymbol{I}^{(1)}$ to generate intermediate region-based visual inputs. The corresponding input $\boldsymbol{Q}$ is "*At 9.00 sec, what is the positional relationship of the green truck model relative to the teddy bear?*"

`HuggingFace`) to detect objects based on the extracted object classes from (a). We set both detection and text thresholds to 0.25. The detected bounding boxes are then refined using SAM2 (`sam2.1_hiera_large`) to obtain refined segmentation masks.

(c) Set of Marks. We leverage Set-of-Mark (SoM) [77] to generate a intermediate region-based visual, serving as a bridge to convert non-region-based inputs into our final region-based format. We overlay numbered markers on the detected objects in $\boldsymbol{I}^{(1)}$, creating an annotated image where each object is labeled with a unique ID and its class name, *e.g.*, "0:cat", "1:table". An example image is shown in Fig. A5.

(d) Matching. We feed the annotated image from (c) and $\boldsymbol{Q}$ into Qwen2.5-VL-32B-Instruct with the following prompt to match the objects in $\boldsymbol{Q}$ to the marked regions.

Task: You will receive (1) an RGB image with labeled objects (a frame from a video) and (2) a natural-language question.
Instructions:

- Identify which labeled objects the question refers to
- Replace object mentions with tokens: `<obj_1>`, `<obj_2>`, etc.
- If no objects match, return the original question with empty `obj_classes`

Output Format: End your answer with "`### Final Answer:`" followed by JSON:

```
{
  "question": "...",
  "obj_classes": ["id:class_name", ...]
}
```

Examples:
Q: "What is the color of the car?"
(car labeled as `1:car`)
A:

```
{
  "question": "What is the color of
               <obj_1>?",
  "obj_classes": ["1:car"]
}
```

Q: "What is the color of the cars?"
(two cars: `1:car`, `2:car`)
A:

```
{
  "question": "What is the color of
               <obj_1> and <obj_2>?",
  "obj_classes": ["1:car", "2:car"]
}
```

Q: "What is the color of the car?"
(no car labeled)
A:

```
{
  "question": "What is the color of
               the car?",
  "obj_classes": []
}
```

(e) Verification. We manually verify all converted questions to ensure quality. We use Label Studio [64] to build a simple interface where human annotators can review each QA pair along with the video and the detected regions. Questions where the grounding fails, *i.e.*, no objects detected or object misalignment, are fixed by annotators. If a question cannot be fixed, it is filtered out. We trim down the input video if the object appears later in the video instead of the first frame. We exclude VQA sample where the object of interest in $\boldsymbol{Q}$ is too ambiguous to ground clearly for our human annotators. Overall, samples requiring correction or removal account for more than 50% of the candidate samples from the automated pipeline, underscoring the necessity of this human verification step. The final R4D-Bench contains 1,419 region-based QA pairs.

A2.2. Question Categories

R4D-Bench contains 9 question categories covering both static and dynamic aspects of 4D understanding. Of the 9 categories, 4 of them are sourced from VLM4D [91] and the other 5 are sourced from STI-Bench [30]. For each category, we provide its defintiion below. We also attach several video examples in the supplementary folder under `r4d_examples/` .

For the **T**ranslational (T), **R**otational (R), **C**ounting (C), and **F**alse **P**ositive (FP) questions, we follow the definitions in VLM4D [91]. We downloaded the dataset from their official source on HuggingFace, *i.e.*, `shijiezhou/VLM4D` .

However, as of the time of writing, they do not provide the list of QA pairs for each category. Therefore, we leverage Qwen2.5-VL-32B-Instruct [52] and human annotators to classify each QA pair into the 4 categories. Of the region-based QA pairs in R4D-Bench obtained from VLM4D, the distribution across different categories is as follows:

- Translational: 61.3%
- Rotational: 10.2%
- Counting: 15.4%
- False Positive: 13.1%

In comparison, the official VLM4D benchmark has the following distribution:

- Translational: 55%
- Rotational: 19%
- Counting: 17%
- False Positive: 9%

Our categorization results are largely consistent with the official distribution with slight difference.

For the 3D **V**ideo **G**rounding (VG), **S**patial **R**elationship (SR), **D**imension **M**easurement (DM), **D**isplacement & **P**ath Length (DP), and **S**peed & **A**cceleration (SA) questions, we follow the definition of STI-Bench [30]. We downloaded the dataset from their official source on HuggingFace, *i.e.*, `MINT-SJTU/STI-Bench`. We note that the original STI-Bench contains two additional categories, *i.e.*, *Ego-centric Orientation* and *Trajectory Description*, where these questions focuses on the ego-centric 4D understanding from the viewpoint itself. Since R4D-Bench focuses on region-based 4D VQA, where another region of interest needs to be provided, these questions are not applicable and removed from R4D-Bench.

The followings are the detailed explanations for each category:

Translational (T) questions target the MLLM's capabilities to understand the linear movement of objects. They usually involve the following movement-related diretion, such as left, right, north, south, away, towards, etc. We provide several examples of R4D-Bench translational questions in Fig. A6.

Rotational (R) questions, on the other hand, care about the rotational movement of objects. They usually involve the following movement-related words, such as rotate, spin, twist, turn, etc. We provide several examples of R4D-Bench rotational questions in Fig. A7.

Counting (C) questions focusing on the MLLM's ability to accurately count the number of objects or occurrences of actions. We provide several examples of R4D-Bench counting questions in Fig. A8.

False Positive (FP) questions are designed to trick the MLLM. The questions will intentionally describe events that do not actually occur within the video, *e.g.*, asking about movements when no object is moving. We note that the original VLM4D false positive questions also ask about objects

Figure A6. **Translational questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

that do not exist in the video. Due to the nature of region-based 4D VQA in R4D-Bench, we do not include these types of questions since the regions cannot refer to non-existent objects. We provide several examples of R4D-Bench false positive questions in Fig. A9.

3D Video Grounding (VG) questions ask MLLMs to retrive the 3D bounding box of objects. The options are formatted as JSON with "dimension (size)" $\in \mathbb{R}^3$, "central point (coordinate)" $\in \mathbb{R}^3$ and "orientation" $\in \mathbb{R}^3$, (*i.e.*, yawn, pitch, and roll) or "camera heading" $\in \mathbb{R}^1$. We provide an example in Fig. A10. As shown in the example, the MLLM needs to be fairly precise to answer these questions correctly, as the differences between options can be quite small.

Spatial Relationship (SR) questions assess the 3D spatial relationship between selected objects or the camera. The options usually involve relative positioning terms, such as left, right, front, back, up, down, etc. We provide an example of R4D-Bench spatial relation questions in Fig. A11.

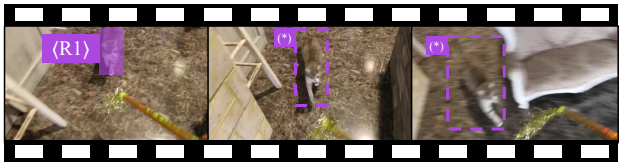
Dimension Measurement (DM) questions care about the physical measurements of objects, such as size and distance. They usually require MLLMs to understand and perceive depth information in order to predict the numerical values. We provide an example of R4D-Bench dimension measurement questions in Fig. A12.

Q: *Is ⟨R1⟩ spinning clockwise or counter-clockwise?*

✗ no dancers ✗ counter-clockwise ✓ clockwise ✗ no spinning

Q: *Is the ⟨R1⟩ spinning clockwise or counter-clockwise?*

✗ clockwise ✗ no cars ✗ not moving ✓ counter-clockwise



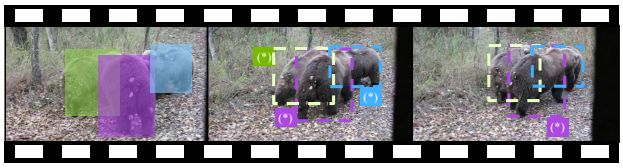
Q: *Is ⟨R1⟩ turning to the left or right from its own perspective?*

✓ left ✗ right ✗ not moving ✗ not sure

Figure A7. **Rotational questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

Q: How many times did ⟨R1⟩ dribble the ball with his left hand?

✗ 4 ✗ 8 ✗ 1 ✓ 0



Q: How many times did ⟨R1⟩, ⟨R2⟩, ⟨R3⟩ are facing away from the camera?

✗ 1 ✗ 0 ✗ 3 ✓ 2

Q: How many spoonfuls of ⟨R1⟩ did the person pour to the right?

✗ 3 ✓ 2 ✗ 4 ✗ 1

Figure A8. **Counting questions in R4D-Bench.** We note that the regions labeled with (*), (*), or (*) are not provided in R4D-Bench; they are visualized for readability.

Displacement & Path Length (DP) questions measures the travel distance of objects. They often involve MLLMs to track motion across selected frames. We provide an example of R4D-Bench displacement and path length questions in Fig. A13.

Speed & Acceleration (SA) questions estimate the motion dynamics of objects. The MLLM needs to consider both the displacement and time intervals to answer them correctly. We provide an example of R4D-Bench speed and acceleration questions in Fig. A14.

A3. Additional Results

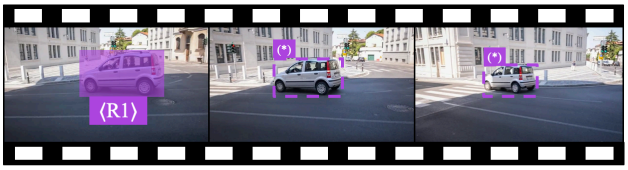
More NVILA variants. In Tab. A1 and Tab. A2, we provide additional results using NVILA-Lite-15B as the base MLLM on non-region-based 4D VQA and R4D-Bench, respectively. We observe consistent performance improvements across various benchmarks.

Temporal Perception. As discussed in Sec. 4.1 and Sec. 6.3, we observe that MLLMs tend to struggle with temporal perception. To demonstrate such a deficiency, we conduct a toy experiment. As shown in Fig. A15, we curate *TimeBench*, a simple set of VQA questions that require temporal perception of input frames, such as "*How many seconds have passed in the input video?*". All videos are acquired from the STI-Bench [30] and VLM4D [91]. We note that these two benchmarks have 4 different frame rates, ranging from 10 to 30, as shown in Tab. 1. This makes it even more challenging for MLLMs to infer time duration. To avoid ambiguity in answers, we provide 4 extra options for each question, ranging from 0.25× to 4× of the actual time duration.

Zero-shot and *P4D* in Tab. A3 show that without cues, MLLMs struggle to know how much time has passed in the input frames. The baselines are naively guessing the answers, resulting in an accuracy close to random guessing (20%). This problem is further exaggerated by the inconsistency that

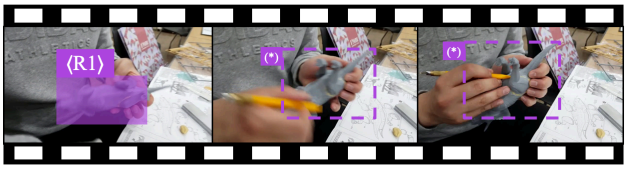
Table A1. **Evaluation on non-region-level 3D / 4D benchmarks.** We report the average multiple-choice accuracy (↑) on each benchmark. For simplicity, we use the following abbreviations: STI (STI-Bench [30]), V4D (VLM4D-real [91]), MMSI (MMSI-Bench [79]), OS (OmniSpatial [22]), and VSTI (VSTI-Bench [15]).

Methods	STI	V4D	MMSI	OS	SAT	VSTI
NVILA-Lite-8B	33.8	46.5	31.3	37.2	62.0	45.2
4D-RGPT-8B (Ours)	37.6 +3.8	52.7 +6.2	33.3 +2.0	40.4 +3.2	64.7 +2.7	59.1 +13.9
NVILA-Lite-15B	34.2	45.1	29.5	41.0	62.7	42.4
4D-RGPT-15B (Ours)	38.1 +3.9	53.7 +8.6	31.7 +2.2	42.7 +1.7	65.3 +2.6	58.6 +16.2



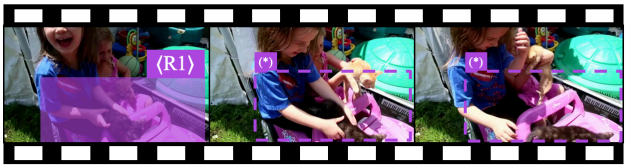
Q: *Is ⟨R1⟩ spinning clockwise or counter-clockwise?*

✓ not spinning ✗ clockwise ✗ counter-clockwise ✗ no cars



Q: *What direction is ⟨R1⟩ moving towards?*

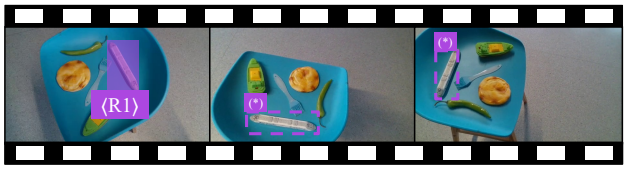
✓ staying in place ✗ left ✗ right ✗ towards



Q: *What direction is ⟨R1⟩ moving toward?*

✓ not moving ✗ left ✗ uphill ✗ right

Figure A9. **False positive questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.



Q: *At 7.00 sec, identify the correct 3D bounding box localization for ⟨R1⟩ from a single frame. (unit: cm, °).*

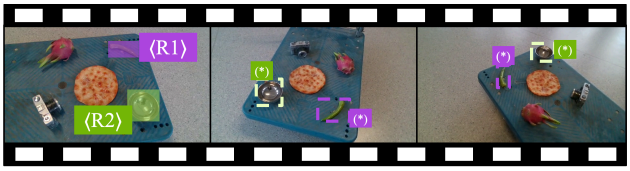
✓
```
{
  dimensions: [23.62, 3.51, 2.79],
  central_point: [5.88, 9.73, 51.40],
  orientation:
  {
      yaw: 167.42,
      pitch: 15.93,
      roll: 59.99
  }
}
```

✗
```
{
  dimensions: [22.87, 3.12, 2.79],
  central_point: [5.88, 9.73, 51.15],
  orientation:
  {
      yaw: 167.42,
      pitch: 12.18,
      roll: 63.74
  }
}
```

Figure A10. **3D video grounding questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability. For simplicity, we only show 1 correct option and 1 wrong option here, but there are 5 options for each 3D video grounding question in R4D-Bench.

different sources of training data and evaluation benchmarks have different frame rates.

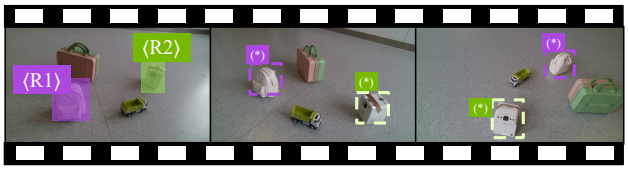
We observe that both *P4D+mark* and *P4D+prompt* can greatly improve the performance on *TimeBench*, which is



Q: *At 7.00 sec, what is the positional relationship of the ⟨R1⟩ relative to the ⟨R2⟩?*

✗ left ✓ right ✗ front ✗ back ✗ up

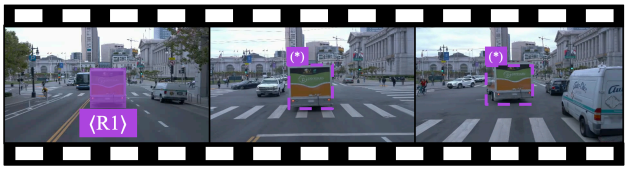
Figure A11. **Spatial relation questions in R4D-Bench.** The question asks about the spatial relationship at 7 seconds, which corresponds to the middle frame out of the three frames shown. We note that the regions labeled with (*) or (*) are not provided in R4D-Bench; they are visualized for readability.



Q: *At 0.00 sec, what is the most likely minimum relative distance between ⟨R1⟩ and ⟨R2⟩ (unit: cm)?*

✗ 51.52 ✗ 59.00 ✓ 54.63 ✗ 47.78 ✗ 64.12

Figure A12. **Dimension measurement questions in R4D-Bench.** We note that the regions labeled with (*) or (*) are not provided in R4D-Bench; they are visualized for readability.



Q: *From 0.00 sec to 12.80 sec, What is the most likely displacement (straight-line distance) of the ⟨R1⟩ between two frames?*

✗ 36.72 m ✗ 15.50 m ✓ 27.50 m ✗ 21.50 m ✗ 30.50 m

Figure A13. **Displacement & path length questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

expected since they provide explicit temporal cues. However, they require additional data preprocessing and distract MLLMs from the main visual and textual content. This toy experiment inspires us to develop methods that can provide temporal cues without modifying the input data, *i.e.*, our TPE.

Training Data Mixture. We conduct an ablation study on the training data mixture for 4D-RGPT. We incrementally add different datasets to analyze their contributions. In Tab. A4, we observe that compared to the *Zero-shot* baseline, adding the training data from VSTI-Bench [15], Wolf [27], or RoboFAC [40] improves the performance on

Q: *At 3.00 sec, What is the most appropriate instantaneous speed of* ⟨R1⟩ *over the specified time interval?*

✗ 3.74 m/s ✗ 0.75 m/s ✓ 0.00 m/s ✗ 14.97 m/s ✗ 7.48 m/s

Figure A14. **Speed & acceleration questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

Table A2. **Evaluation on R4D-Bench.** We report performance on the static split (**Sta**), the dynamic split (**Dyn**), and all 9 tasks of R4D-Bench. For simplicity, we abbreviate them as follows: 3D **V**ideo **G**rounding (VG); **D**imension **M**easurement (DM); **S**patial **R**elationship (SR); **R**otational (R); **C**ounting (C); **T**ranslational (T); **F**alse **P**ositive (FP); **S**peed & **A**cceleration (SA); and **D**isplacement & **P**ath Length (DP).

Methods	Avg	Sta	Dyn	VG	DM	SR	R	C	T	FP	SA	DP
NVILA-Lite-8B	37.9	29.1	41.3	33.9	20.2	46.3	41.5	39.6	41.9	40.7	45.9	32.1
4D-RGPT-8B (Ours)	42.2	32.9	45.7	35.1	26.3	52.2	43.1	40.1	48.7	40.2	50.9	38.9
	+4.3	+3.8	+4.4	+1.2	+6.1	+5.9	+1.6	+0.5	+6.8	-0.5	+5.0	+6.8
NVILA-Lite-15B	39.7	31.7	42.7	36.5	26.8	31.7	50.9	34.0	46.4	34.8	37.8	21.4
4D-RGPT-15B (Ours)	43.0	35.8	45.7	38.5	32.2	39.0	50.0	38.4	49.6	36.3	45.9	28.6
	+3.3	+4.1	+3.10	+2.0	+5.4	+7.3	-0.9	+4.4	+3.2	+1.5	+7.9	+7.2

Q: *How much time has passed in the video?*

(a) 39.40 s (2.00×) (b) 9.85 s (0.50×) **(c) 19.70 s** ✓

(d) 59.10 s (4.00×) (e) 4.92 s (0.25×)

Figure A15. **TimeBench VQA.** We curate a toy benchmark to evaluate MLLMs' temporal perception. We note that the "($M\times$)" indicates the multiplier between the wrong option and the correct one. They are not provided in the actual question but are shown here for clarity.

Table A3. **Ablation studies on explicit temporal cues.** We experiment without and with different choices of explicit time cues. For simplicity, we use the same abbreviations as Tab. 4.

Methods	Time cues	TimeBench	STI	R4D
Zero-shot	✗	22.7	33.8	37.9
P4D	✗	30.1	34.8	41.0
P4D+mark	marks	95.3	35.1	41.1
P4D+prompt	prompts	**98.0**	**36.1**	**41.5**

both non-region-based (STI-Bench) and region-based 4D VQA (R4D-Bench). Though SAT [56] is an image-based VQA dataset, adding it also brings moderate performance gains, *i.e.*, $+0.6\%$ on STI-Bench and $+0.4\%$ on R4D-Bench.

Table A4. **Incremental training data mixture.** We incrementally add different datasets to analyze their contributions to 4D-RGPT. For simplicity, we use the same abbreviations as Tab. 4 and the following for each dataset: **V**STI-Bench [15] (V); **W**olf [27] (W); **R**oboFAC [40] (R); and **S**AT [56] (S).

Methods	V	W	R	S	STI	R4D-Bench		
						Avg	Sta	Dyn
Zero-shot	✗	✗	✗	✗	33.8	37.9	29.1	41.3
V	✓	✗	✗	✗	35.4	39.4	30.0	42.9
V+W	✓	✓	✗	✗	36.0	40.6	31.0	44.2
V+W+R	✓	✓	✓	✗	37.0	41.8	32.2	45.4
V+W+R+S (Ours)	✓	✓	✓	✓	**37.6**	**42.2**	**32.9**	**45.7**

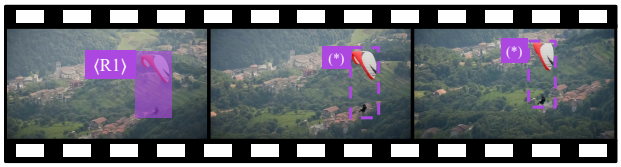
More Qualitative Results. Following the format in Fig. 4, we provide additional qualitative results on R4D-Bench in Fig. A16, Fig. A17, Fig. A18, and Fig. A19.

More $\hat{P}_m$ Visualizations. In Fig. A20, we provide additional visualizations of the 4D-RGPT explicit signals $\hat{P}_m$ at different training steps. In earlier steps, we observe inaccurate predictions with grid-like structures. We hypothesize that this is due to the tokenization process in hidden states of the LLM transformer, *i.e.*, $F_{\texttt{hidden}}$. However, as training proceeds, the grid-like structures gradually diminish, leading to smoother and more reasonable predictions. We demonstrate that our 4D-RGPT can effectively learn to extract explicit 4D perceptual signals through the training of P4D.

Limitations. 4D-RGPT can still produce suboptimal results, particularly in questions requiring precise numerical estimation, *e.g.*, exact speed or displacement values, as illustrated in several failure cases in Fig. A16–A19. We attribute this to the lack of step-by-step reasoning during training.

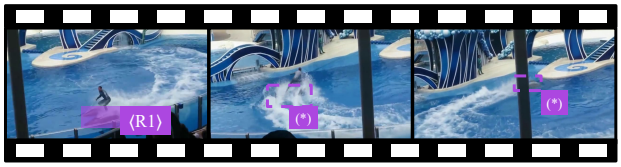
Q: *Are ⟨R1⟩ picking up or putting down the ⟨R2⟩?*

✓ **Ours**: picking up ✗ **GPT**: putting down



Q: *Is the ⟨R1⟩ moving upwards or downwards?*

✓ **Ours**: downwards ✓ **GPT**: downwards



Q: *Are ⟨R1⟩ turning clockwise or counter-clockwise?*

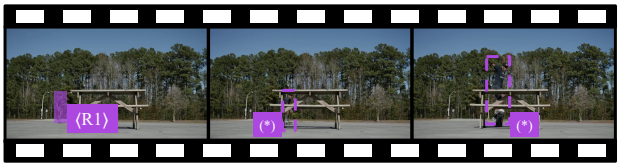
✓ **Ours**: clockwise ✗ **GPT**: counter-clockwise

Q: *Is ⟨R1⟩ turning clockwise or counter-clockwise?*

✗ **Ours**: counter-clockwise ✓ **GPT**: clockwise

Q: *How many ⟨R1⟩ are standing still?*

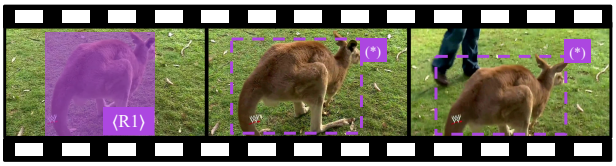
✓ **Ours**: 5 ✗ **GPT**: 3



Q: *How many times does the ⟨R1⟩ jump towards the camera?*

✓ **Ours**: 1 ✓ **GPT**: 1

Figure A16. **More VQA comparison between GPT-4o [45] and 4D-RGPT (Ours) on R4D-Bench.** We provide 2 examples for each of the following categories: Translational, Rotational, and Counting.

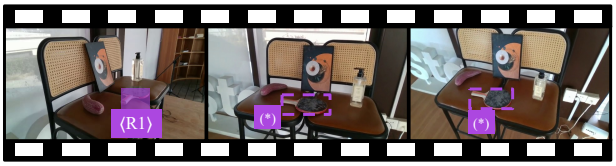


Q: *What direction is ⟨R1⟩ moving towards?*

✓ **Ours**: not moving ✓ **GPT**: not moving

Q: *How many scoops of ⟨R1⟩ does he move left?*

✗ **Ours**: 1 ✗ **GPT**: 2 ✓ **Ans**: 0



Q: *At 27.00 sec, given a single frame, determine the 3D bounding box of ⟨R1⟩. Identify the correct dimensions, central point, and orientation including yaw, pitch, and roll. (unit: cm, °)*

✓ **Ours & GPT**:

```
{
    dimensions: [25.62, 2.38, 15.33],
    central_point: [12.91, 2.77, 90.59],
    orientation: {
        yaw: 117.10,
        pitch: 42.61,
        roll: 114.41
    }
}
```

Q: *At 18.52 sec, what is the 3D bounding box in camera coordinates of the ⟨R1⟩ from a single randomly selected frame? (unit: m, m/s, m/s^2, °)*

✓ **Ours**:

```
{
    C_lwh:
    [0.86, 1.26, 0.74],
    C_central_point:
    [3.55, 1.33, 2.75],
    C_heading:
    27.51
}
```

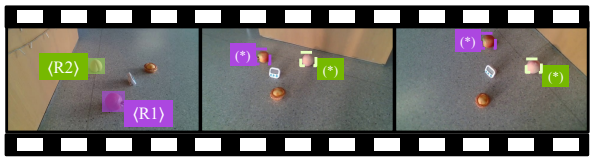
✗ **GPT**:

```
{
    C_lwh:
    [0.86, 1.26, 0.74],
    C_central_point:
    [3.74, 1.39, 2.81],
    C_heading:
    27.20
}
```

Figure A17. **More VQA comparison between GPT-4o [45] and 4D-RGPT (Ours) on R4D-Bench.** We provide 2 examples for each of the following categories: False Positive and 3D Video Grounding.

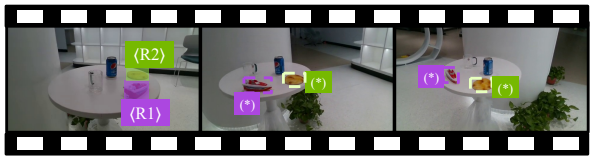
Q: *From 0.00 sec to 1.06 sec, what is the most appropriate height of ⟨R1⟩ ? (unit: m, m/s, m/s², °)*

✓ **Ours**: 1.86 m ✓ **GPT**: 1.86



Q: *At 0.00 sec, What is the most likely minimum relative distance between ⟨R1⟩ and ⟨R2⟩ in a given frame? (unit: cm, °)*

✓ **Ours**: 18.54 cm ✗ **GPT**: 16.71 cm

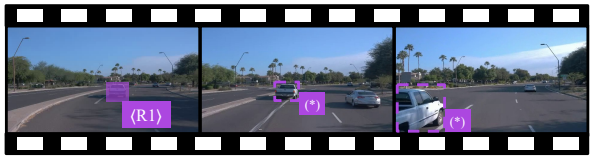


Q: *At 6.00 sec, What is the positional relationship of ⟨R1⟩ relative to ⟨R2⟩ from the observer's perspective?*

✓ **Ours**: left ✓ **GPT**: left

Q: *At 3.00 sec, What is the positional relationship of the ⟨R2⟩ relative to ⟨R1⟩?*

✓ **Ours**: left ✓ **GPT**: left



Q: *At 6.00 sec, what is the most appropriate average or instantaneous speed of ⟨R1⟩?*

✓ **Ours**: 4.60 m/s ✓ **GPT**: 4.60 m/s

Q: *At 14.00 sec, what is the most appropriate average or instantaneous speed of ⟨R1⟩?*

✓ **Ours**: 0.00 m/s ✗ **GPT**: 0.20 m/s

Figure A18. **More VQA comparison between GPT-4o [45] and 4D-RGPT (Ours) on R4D-Bench.** We provide 2 examples for each of the following categories: Spatial Relation, Dimension Measurement, and Speed & Acceleration.

Q: *At 0.28 sec, What is the most appropriate trajectory length (total distance traveled) of ⟨R1⟩ between two frames? (unit: m, m/s, m/s², °)*

✓ **Ours**: 0.0 m ✗ **GPT**: 0.2 m

Q: *From 0.00 sec to 9.50 sec, what is the most likely displacement (straight-line distance) of the camera or object between two frames for ⟨R1⟩?*

✗ **Ours**: 7.53 m ✗ **GPT**: 10.06 m ✓ **Ans**: 8.50 m

Figure A19. **More VQA comparison between GPT-4o [45] and 4D-RGPT (Ours) on R4D-Bench.** We provide 2 examples for each of the following categories: Displacement & Path Length.

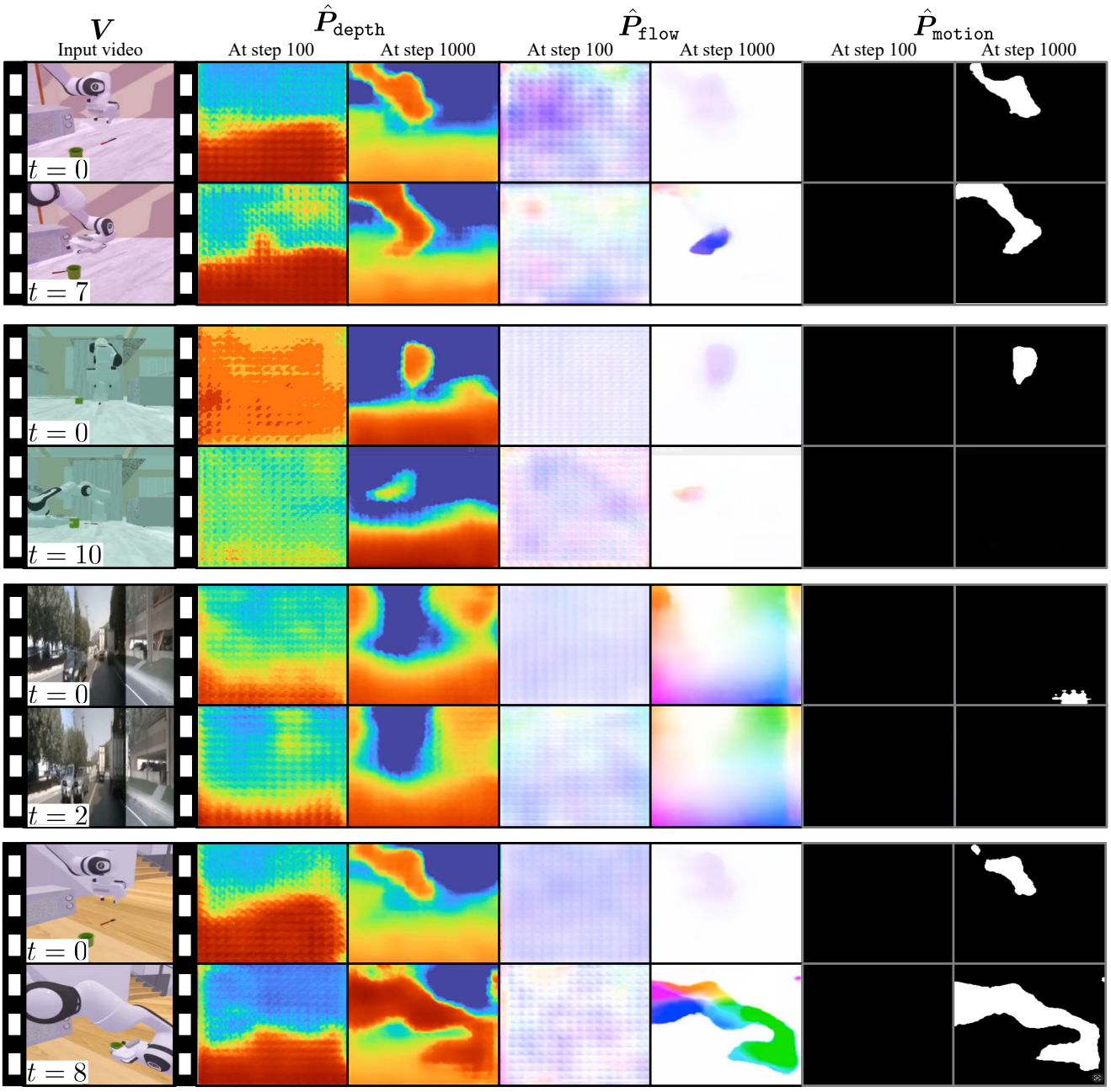


Figure A20. **More visualizations of 4D-RGPT explicit signals $\hat{P}_m$.** Similar to the format of Fig. 5, we visualize the training progress of $\hat{P}_{\texttt{depth}}$, $\hat{P}_{\texttt{flow}}$, and $\hat{P}_{\texttt{motion}}$.