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 $\mathbf{E}_{\texttt{P}}$ 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) $\mathbf{E}_{\texttt{V}}$, they use SigLIP [82], specifically `siglip-so400m-patch14-384`. For the multi-modal projector $\mathbf{E}_{\texttt{P}}$, 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 $\mathbf{E}_{\texttt{4D}}$, 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 $\mathbf{D}_{\texttt{4DP}}$ 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

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.".

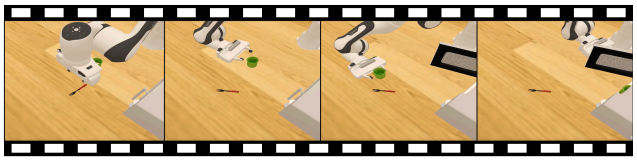


Figure A3. **An example from RoboFAC [40] training data**. The corresponding conversation is as follows: (1) *User*: "From the video, what task is the robot engaged in?"; (2) *GPT*: "Put the fork in the cup and put them in the microwave.".

(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

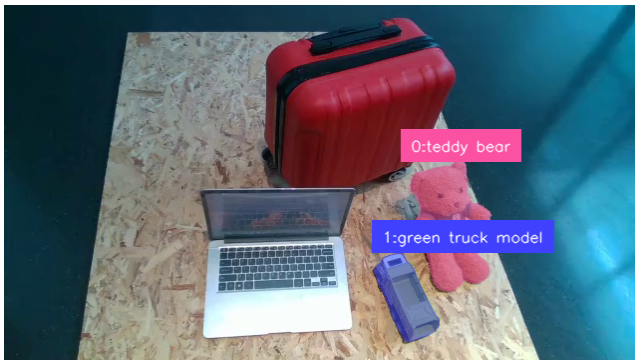


Figure A5. **An example of SoM visual input in R4D-Bench.** We apply SoM [77] on $\boldsymbol{I}^{(1)}$ to generate intermediate region-based visual inputs. The corresponding input $\boldsymbol{Q}$ is "*At 9.00 sec, what is the positional relationship of the green truck model relative to the teddy bear?*"

`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.

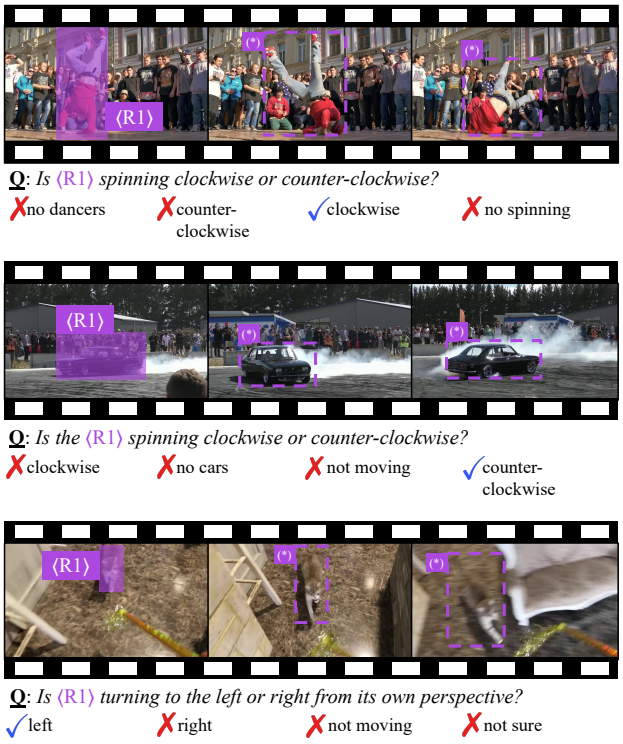


Figure A7. **Rotational questions in R4D-Bench.** We note that the regions labeled with (*) are not provided in R4D-Bench; they are visualized for readability.

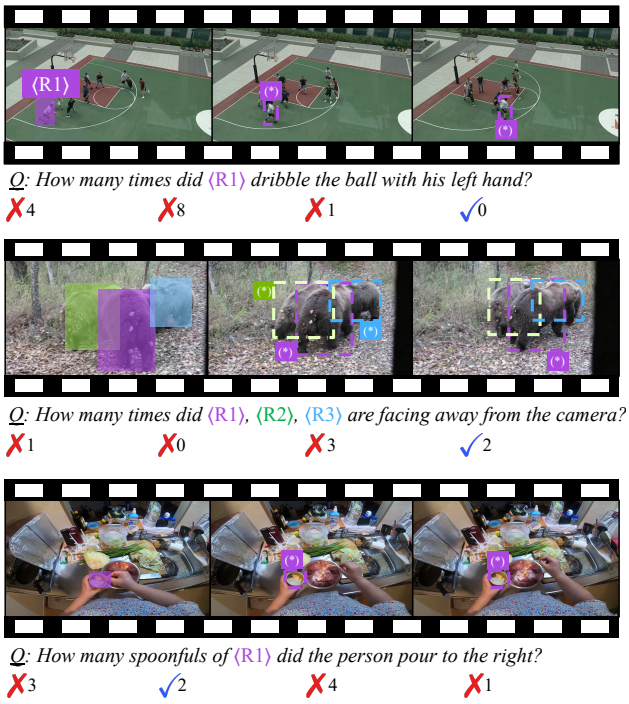


Figure A8. **Counting questions in R4D-Bench.** We note that the regions labeled with (*), (*), or (*) are not provided in R4D-Bench; they are visualized for readability.

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

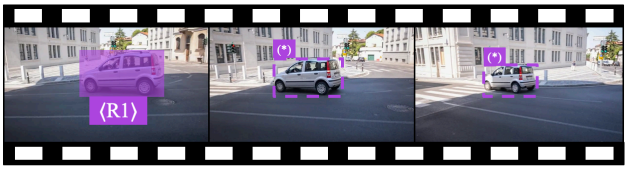
More NVILA variants. In Tab. A1 and Tab. A2, we provide additional results using NVILA-Lite-15B as the base MLLM on non-region-based 4D VQA and R4D-Bench, respectively. We observe consistent performance improvements across various benchmarks.

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 $0.25\times$ to $4\times$ of the actual time duration.

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]).

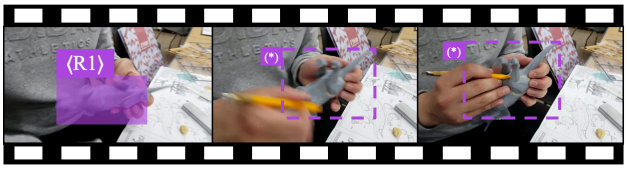
Methods	STI	V4D	MMSI	OS	SAT	VSTI
NVILA-Lite-8B	33.8	46.5	31.3	37.2	62.0	45.2
4D-RGPT-8B (Ours)	37.6 +3.8	52.7 +6.2	33.3 +2.0	40.4 +3.2	64.7 +2.7	59.1 +13.9
NVILA-Lite-15B	34.2	45.1	29.5	41.0	62.7	42.4
4D-RGPT-15B (Ours)	38.1 +3.9	53.7 +8.6	31.7 +2.2	42.7 +1.7	65.3 +2.6	58.6 +16.2

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



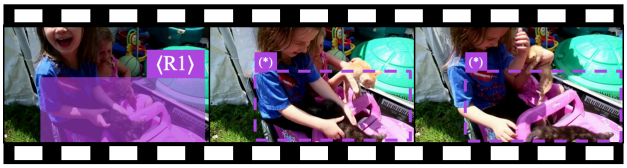
Q: *Is ⟨R1⟩ spinning clockwise or counter-clockwise?*

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



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

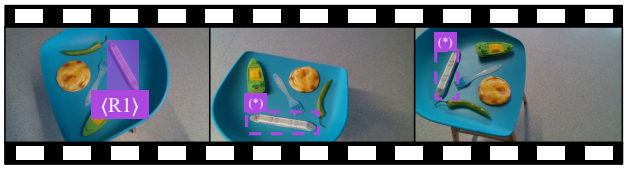
✓staying in place ✗left ✗right ✗towards



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, °).*

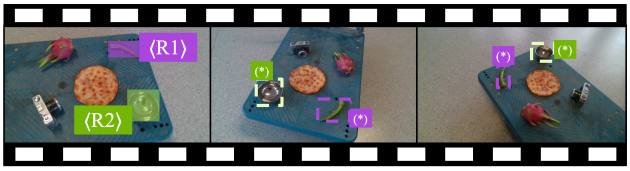
✓
```
{
  dimensions: [23.62, 3.51, 2.79],
  central_point: [5.88, 9.73, 51.40],
  orientation:
  {
      yaw: 167.42,
      pitch: 15.93,
      roll: 59.99
  }
}
```

✗
```
{
  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.

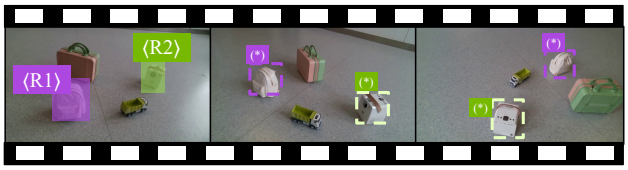
We observe that both *P4D+mark* and *P4D+prompt* can greatly improve the performance on *TimeBench*, which is



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

✗left ✓right ✗front ✗back ✗up

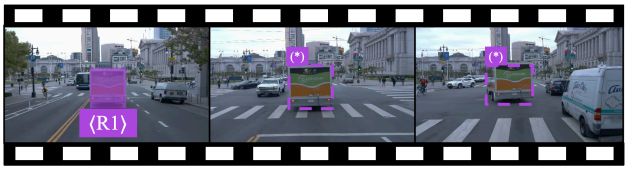
Figure A11. **Spatial relation questions in R4D-Bench.** The question asks about the spatial relationship at 7 seconds, which corresponds to the middle frame out of the three frames shown. We note that the regions labeled with (*) or (*) are not provided in R4D-Bench; they are visualized for readability.



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**

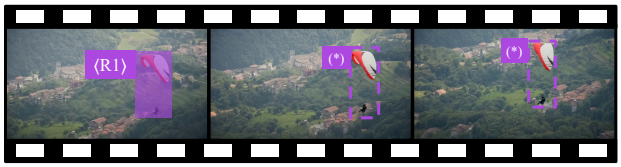
More Qualitative Results. Following the format in Fig. 4, we provide additional qualitative results on R4D-Bench in Fig. A16, Fig. A17, Fig. A18, and Fig. A19.

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.

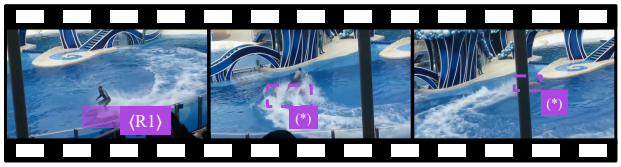
Q: *Are ⟨R1⟩ picking up or putting down the ⟨R2⟩?*

✓ **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?*

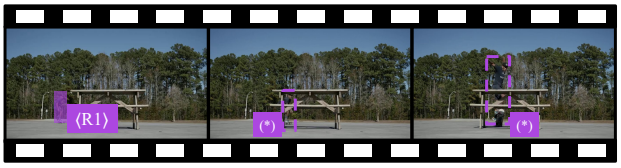
✓ **Ours**: clockwise ✗ **GPT**: counter-clockwise

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

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

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

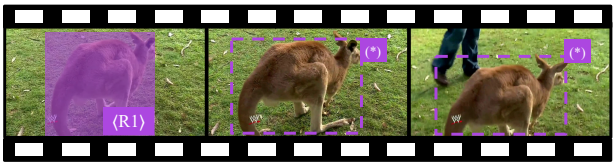
✓ **Ours**: 5 ✗ **GPT**: 3



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.

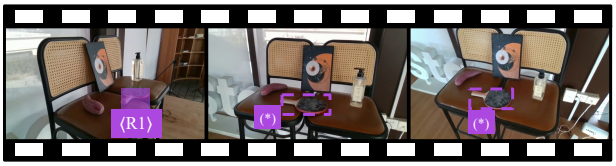


Q: *What direction is ⟨R1⟩ moving towards?*

✓ **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
}
```

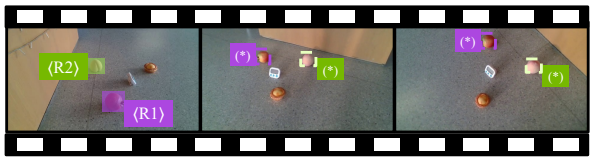
✗ **GPT**:

```
{
    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.

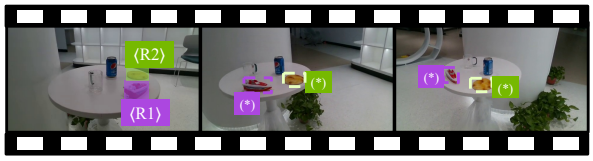
Q: *From 0.00 sec to 1.06 sec, what is the most appropriate height of ⟨R1⟩ ? (unit: m, m/s, m/s², °)*

✓ **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

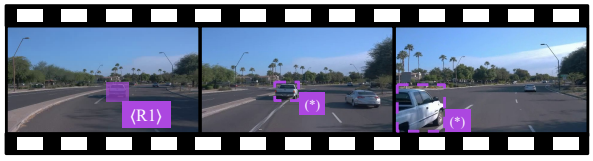


Q: *At 6.00 sec, What is the positional relationship of ⟨R1⟩ relative to ⟨R2⟩ from the observer's perspective?*

✓ **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.

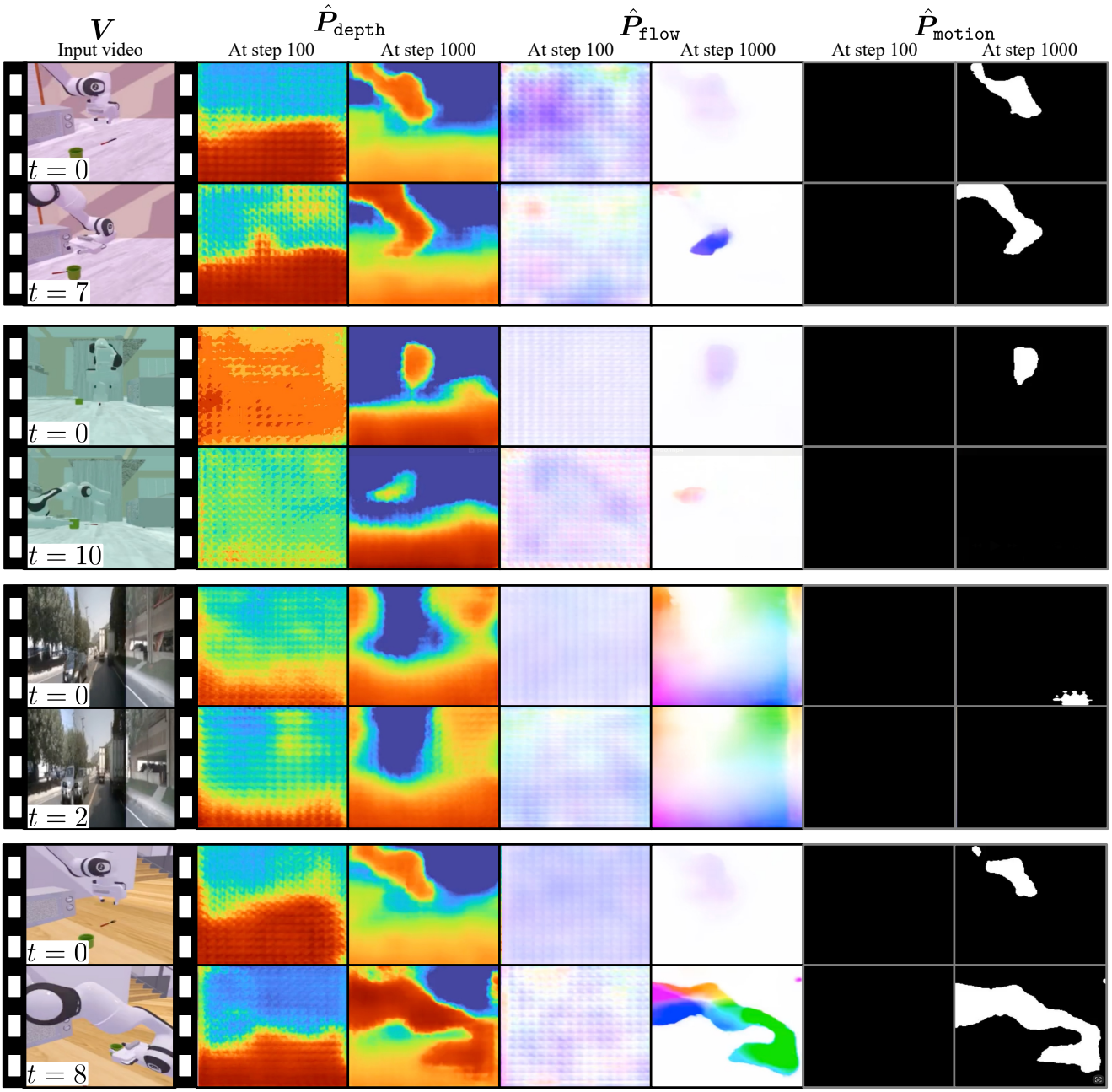


Figure A20. **More visualizations of 4D-RGPT explicit signals $\hat{P}_m$.** Similar to the format of Fig. 5, we visualize the training progress of $\hat{P}_{\texttt{depth}}$, $\hat{P}_{\texttt{flow}}$, and $\hat{P}_{\texttt{motion}}$.