Supplementary Materials for "TRM-VLA: Temporal-Aware Chain-of-Thought Reasoning and Memorization for Vision-Language-Action Models"

Xiang Li[1,2,3] Ya-Li Li[1,2,3‡] Yuan Wang[1,2,3] Shengjin Wang[1,2,3]
[1]Department of Electronic Engineering, Tsinghua University, China
[2]Beijing National Research Center for Information Science and Technology (BNRist), China
[3]National Engineering Research Center of Dangerous Articles
and Explosives Detection Technologies, Beijing; 100084, China
l-xiang24@mails.tsinghua.edu.cn {liyali13, wgsgj}@tsinghua.edu.cn [‡]Corresponding Author

A. Overview

In our main paper, we propose TRM-VLA, a Temporal-aware Reasoning and Memorization framework, which integrates explicit temporal modeling into the VLA reasoning process. TRM-VLA consists of two core components: (1) Keyframe-Triggered Reasoning (KTR), which identifies task progress and performs hierarchical CoT reasoning at key decision points to reduce redundant inference; and (2) Granularity-adaptable Context Memory (GCM), which dynamically stores and retrieves historical reasoning trajectories to maintain inter-frame coherence and global context.s. In this supplementary materials, we extend the discussion on related works in Section B, particularly focusing on the temporal modeling for VLA and language-based CoT reasoning for VLA. In Section C, we provide an in-depth overview of dataset, implementation details, and evaluation metrics. In Section D presents additional quantitative and visual experiments that provide further validation of the effectiveness of the proposed TRM-VLA model. Section E discusses the limitations and future research directions of our approach, while Section F highlights the broader impact of our work.

B. More Details about Related Works

Temporal Modeling for VLAs. Temporal modeling in vision-language-action (VLA) systems has emerged as a crucial direction as embodied tasks increasingly require long-horizon reasoning and robustness under partial observability. Recent works address this challenge from different perspectives, including multi-frame fusion, temporal token aggregation, and memory-augmented policies. For example, ContextVLA [4] amortizes multi-frame visual observations into a compact context token, enabling efficient temporal cue integration without excessive computational cost. TTF-VLA [13] introduces a training-free temporal token fusion mechanism that selects and fuses visual tokens across frames based on pixel-level dynamics and semantic attention, enhancing temporal stability. CronusVLA [9] transforms single-frame VLA backbones into multi-frame models through feature chunking and lightweight temporal post-training, improving robustness to visual disturbances. Another important direction focuses on longer-term internal memory: MemoryVLA [16] designs perceptual-cognitive memory modules to store and retrieve hierarchical reasoning signals across time, turning VLA policies into history-aware agents. In a similar spirit, MEMER [17] scales up memory for robot control by retrieving structured experience from a large episodic database, enabling policies to leverage past task executions as contextual grounding for the current decision. Complementary to these, HAMLET [8] augments VLA models with moment-level historical context, and CoT-VLA [23] incorporates visual chain-of-thought reasoning to explicitly model intermediate visual sub-goals.

Overall, these efforts point toward three essential themes for temporal modeling in embodied VLA systems: (1) how to effectively incorporate multi-frame cues without significant overhead, (2) how to maintain or retrieve long-term memory that persists beyond immediate observations, and (3) how to integrate hierarchical temporal abstractions—from high-level planning to low-level control—into action generation. Our proposed KTR and GCM modules align directly with these themes by generating hierarchical reasoning at keyframes and maintaining a dynamically updated reasoning context, thereby enabling the model to "think appropriately" and "think temporally" throughout the task.

Language-based CoT for VLAs. A growing line of work explores language-based chain-of-thought (CoT) or reasoning traces as intermediate representations for embodied control. Embodied Chain-of-Thought (ECoT [22]) explicitly trains VLAs to generate multi-step embodied reasoning about plans, sub-tasks, motions, and visually grounded quantities (e.g., object boxes, end-effector positions) before predicting actions, substantially improving OpenVLA's [7] generalization and interpretability. EmbodiedGPT [14] fur-

ther scales this idea to pre-training: it introduces the Ego-COT dataset with video-level CoT sub-goals and shows that language-based planning signals can significantly boost downstream embodied planning and control performance.

Recent foundation VLA models incorporate internal natural-language reasoning more directly. EO-1[15] interleaves vision, text, and action tokens during pretraining, improving multimodal embodied reasoning and control via interleaved vision–text–action modeling. Gemini Robotics 1.5 [18] pushes this further with "embodied thinking": it executes multi-level natural-language reasoning between actions, effectively "thinking before acting" to decompose complex multi-step tasks and explain its decisions. InstructVLA [20] performs instruction tuning "from understanding to manipulation," aligning language-based reasoning with precise low-level actions. ChatVLA-2 [24] focuses on preserving and exploiting the open-world reasoning capabilities of a pretrained VLM, coupling them with action generation for tasks such as math-on-whiteboard manipulation and spatial reasoning. OneTwoVLA [12] unifies acting (System One) and reasoning (System Two) in a single VLA that can adaptively switch between fast reactive control and deliberative language-based reasoning. ThinkAct [3] adopts a dual-system design in which a multimodal LLM first produces reinforced visual-latent plans from CoT-style reasoning, which then condition a downstream action model for robust long-horizon execution. Embodied-R1 [21] and "Training Strategies for Efficient Embodied Reasoning" [2] systematically study how reinforced or lightweight CoT-style reasoning improves VLA representations and action prediction, demonstrating that generating and attending to intermediate language reasoning can boost both success rate and efficiency. Compared with these works, which primarily focus on what to reason in language (e.g., plans, sub-goals, or explanations), our TRM-style framework emphasizes when and how to reason over time: we introduce Keyframe-Triggered Reasoning (KTR) and Granularity-adaptable Context Memory (GCM) to align hierarchical language-based CoT with keyframes and task phases, and to maintain a dynamically updated reasoning context across timesteps, thereby reducing redundant CoT tokens while strengthening temporally consistent, embodied reasoning for action generation.

C. Data, Metrics, and Implementation Details

C.1. Datasets

As shown in Figur e. 2, we use a teacher-follower system provided by AIRBOT to collect data. The system can map the joint angle of the teacher's arm to the follower's arm in real time. Human operators can complete tasks through teleoperation while simultaneously collecting data. During data collection, we randomly alter the position and orientation of the manipulated object, as well as introduce background

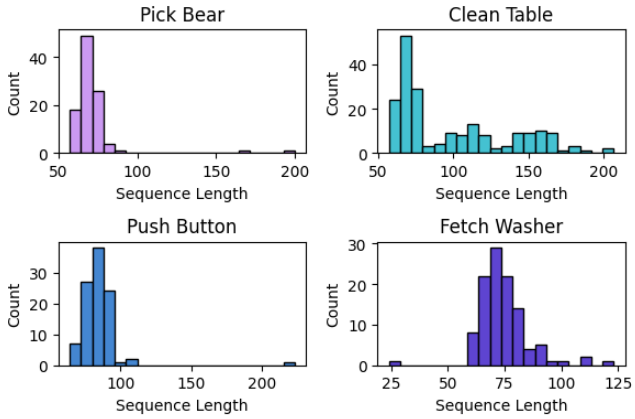

Figure 1. Data statistics of collected data in real world.

disturbances. Additionally, the operators do not strictly ensure that the trajectories are identical each time, meaning different routes or methods may be employed to complete the task. Through the system mentioned above, we collect four real-world tasks. The statistics of collected data are shown in Figure. 1. We collect 410 expert demonstration trajectories on an AIRBOT Player robot across four carefully designed real-world tasks, each targeting a distinct dimension of embodied reasoning that our model aims to improve:

1. *Scoop Washers with Shovel*. The robot must manipulate a non-end-effector tool (a handheld shovel) to scoop small metal washers from a tray. This task probes the model's ability to: generalize beyond direct gripper manipulation, infer tool affordances, and perform multi-step, contact-rich motions requiring fine trajectory control. Tool-use tasks are notoriously challenging for VLAs because visual observations differ significantly from gripper-centric motions, making this task an effective test of our model's grounded reasoning and action adaptation.

2. *Clean Table*. The robot clears heterogeneous household objects (toy bear, banana, cup, etc.) from a tabletop into a bin. This scenario evaluates: long-horizon planning over multiple sub-goals, object-agnostic reasoning under large appearance variations, and consistent policy execution across distractors and clutter. Because object categories vary across demonstrations, this task tests whether the model can form generalizable, abstraction-level "cleaning" strategies rather than memorizing object-specific behaviors.

3. *Push Buttons in Order*. The robot must press three spatially separated colored buttons according to a given sequence (e.g., "red → green → blue"). This task explicitly requires: temporal memory of previously completed steps, reasoning over symbolic ordering constraints, and sequential action planning with precise spatial alignment. The task is difficult for reactive VLA baselines that lack temporal grounding, making it a direct probe of the benefits introduced by our TRM framework.

(a) Follower arm (b) Teacher arm

Figure 2. The data collection system of AIRBOT.

4. *Place Toy Bear into Box*. The robot grasps a soft, deformable toy bear presented in diverse initial poses, then places it into a box. This task stresses: robustness to deformable-object geometry, manipulation under pose and shape uncertainty, and reliable object-state reasoning for accurate placement. Since the bear is non-rigid and visually variable, this task evaluates the model's ability to maintain stable perceptual understanding across the task.

Together, these four tasks cover tool use, multi-step symbolic sequencing, long-horizon planning, perception-driven reasoning, and robust precise manipulation—a comprehensive spectrum of embodied reasoning challenges. These capabilities align directly with the core contributions of our framework: temporal-aware reasoning, memory-guided action generation, and understanding of task progression.

C.2. Metrics

To comprehensively evaluate robotic performance beyond binary task success, we report both **Success Rate** and a more fine-grained **Progress Score**. While success rate indicates whether the robot completes the final goal, the progress score measures partial task completion and reveals intermediate execution quality, temporal reasoning ability, and failure modes that success rates alone cannot capture.

Success Rate. A rollout is counted as successful if the robot completes the full task objective within the allotted steps, following standard VLA evaluation protocols.

Progress Score. For each task, we decompose its execution into a sequence of semantically meaningful sub-tasks. The progress score is computed as:

$$\text{Progress} = \frac{\text{Completed Sub-Tasks}}{\text{Total Sub-Tasks}}. \quad (1)$$

This metric captures partial progress and enables evaluation even when the robot fails to reach the final goal, providing a more discriminative measure of multi-step execution.

Sub-Task Definitions. We define sub-tasks for each real-world task as follows:

- **Pick Bear.** The task is divided into three stages: (1) navigating to a position above the bear, (2) grasping the bear, (3) placing it into the bin.
- **Clean Table.** Each object placed into the bin counts as one completed sub-task. Since objects vary in shape and appearance, this metric measures consistent multi-object clearing behavior and long-horizon planning.
- **Push Buttons in Order.** Pressing each button in the correct sequence is counted as a completed sub-task. This task explicitly tests temporal memory, symbolic order reasoning, and robustness across multiple sequential decisions.
- **Fetch Washers with Shovel.** We decompose the task into three stages: (1) reaching a position above the washer container, (2) performing a correct scooping motion with the shovel, (3) successfully removing washers. This evaluates tool-based manipulation, motion coordination, and grounded affordance understanding.

Together, success rate and progress score offer complementary perspectives on robotic performance: success rate measures final task competence, while progress score captures intermediate execution quality, temporal consistency, and sub-task transition reliability.

C.3. Training Details

In addition to the original action denoising objective [10], we introduce a chain-of-thought reasoning loss with a weighting factor of 0.1, encouraging the model to align its temporal reasoning with the annotated embodied CoT signals. The Action Expert uses 4 denoising steps and predicts an action chunk of length 16, meaning that each forward pass generates the next 16 future actions. Dataset preprocessing follows the procedures introduced in CogACT [10]. For temporal-aware reasoning (KTR), we apply tag-wise dropout with a probability of 0.3 during training, which we find significantly improves model robustness. The early, middle, and late phases are determined based on relative trajectory progress: the first 10% of timesteps form the early phase t_e, the next 10%–20% form the middle phase t_m, and the remaining portion constitutes the late phase t_l.

We initialize our model from pretrained checkpoint [10] and fine-tune all parameters end-to-end to fully exploit temporal reasoning and action generation synergy. We use the Adam optimizer with a learning rate of 2×10^{-5} and a batch size of 256, training the model for $20{,}000$ iterations. Standard data augmentation is applied to improve generalization across real-world variations. All experiments are conducted on a cluster with 8 NVIDIA A100 GPUs (80GB each).

C.4. Inference Details

A major challenge in reasoning-augmented VLA models is maintaining fast inference speed. Following the strategy introduced in ECoT [22], we adopt a **frozen reasoning** mechanism to reduce computational overhead during rollout.

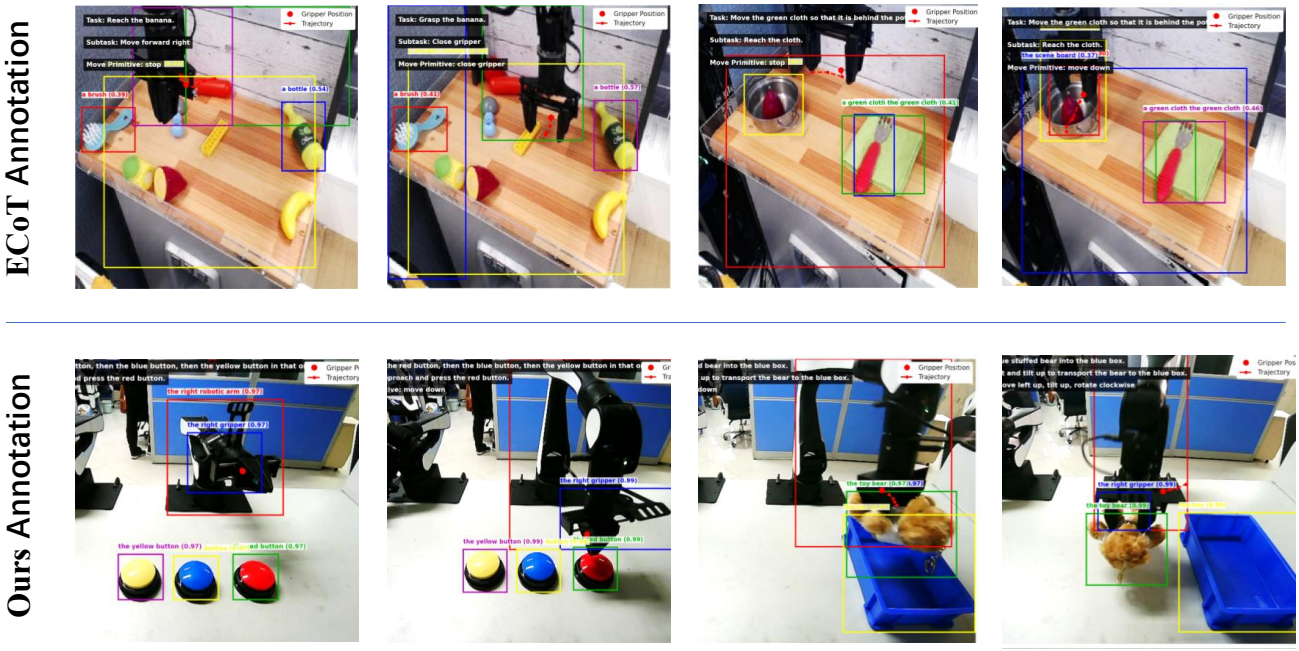

Figure 3. The quality comparision of CoT annotation. Top: the annotation visualization from ECoT [22] pipeline on BridgeV2 Dataset. Bottom: the visualizaiton annotation from our improved reasoning annotation in real-world, which is more accurate and consistent.

After each reasoning step, the generated chain-of-thought is frozen and reused for subsequent timesteps. For the next T frames, if the model does not detect a keyframe trigger for temporal-aware reasoning (KTR), the previously frozen reasoning is directly fed into the model without regeneration. In practice, we set $T \in \{6, 12\}$ depending on the task horizon.

For the Temporal-Aware Memory (GCM) module, different reasoning levels are maintained with distinct lifetimes. High-level reasoning (e.g., *task*, *plan*) remains in the memory buffer indefinitely until replaced by a new reasoning entry of the same tag. Mid-level reasoning (e.g., *perception*, *subtask_reason*) persists for up to 24 frames if not updated, after which it is automatically discarded. To preserve responsiveness to real-time control signals, low-level reasoning (e.g., *move*, *move_reason*) is assigned the shortest lifetime and is removed after 6 frames. This inference strategy significantly reduces redundant reasoning generation while preserving temporal consistency and responsiveness, enabling efficient deployment of our reasoning-augmented VLA model.

C.5. Improved ECoT-Style Reasoning Annotation

The original ECoT [22] annotation pipeline involves multiple perception modules and several LLM stages, resulting in high complexity and inconsistent outputs. To address these limitations, we redesign the entire pipeline into four steps, all executed using a single multimodal large model, Qwen3-VL [1], together with a robust trajectory tracking module [5]. This simplifies the workflow, reduces noise accumulation, and significantly improves annotation quality.

Comparison with ECoT Annotation Pipeline. Fig. 3 compares the CoT annotation quality produced by the original ECoT pipeline (top) with our improved annotation procedure (bottom). As shown, the ECoT-generated annotations exhibit several limitations: (i) *Noisy and inaccurate detection.* The bounding boxes produced by ECoT are often cluttered, contain false positives, and frequently include irrelevant background objects. Such noise increases the burden on the reasoning model and distracts it from the task-relevant entities. (ii) *Temporal inconsistency.* Even for consecutive frames, ECoT yields unstable and inconsistent detections, undermining temporal coherence and making it difficult for policies to maintain reliable object tracking. (iii) *Misalignment between CoT reasoning and executed actions.* The annotated motion primitives and gripper trajectories are often inconsistent with the robot's actual behavior (e.g., incorrect red trajectories), reducing the effectiveness of the supervision signal. In contrast, our improved pipeline—leveraging unified Qwen3-VL grounding, CoTracker3-based trajectory extraction, and structured reasoning templates—produces clean, accurate, and temporally stable annotations. The detections remain consistent across frames, irrelevant objects are effectively filtered out, and the reasoning content aligns closely with both visual evidence and actual robot motion. These improvements significantly enhance annotation reliability and contribute to the superior temporal reasoning capabilities of TRM-VLA.

Step 1: Scene and Spatial Relation Captioning. We first generate a concise description of the task scene, including all relevant objects and their spatial relations. Compared with the PrismaticVLM [6] used in ECoT, Qwen3-VL provides more accurate scene captions. We use the following prompt:

```
The robot task is: {instruction}.
Briefly describe the things in this
scene and their spatial relations to
each other.
```

Here, `instruction` is the natural-language task instruction. This step establishes the global grounding necessary for subsequent reasoning stages.

Step 2: Object Detection For every demonstration frame, we ask Qwen3-VL to output bounding boxes, classification, and confidence scores using the prompt:

```
Detect objects related
to the following objects:
{detection_prompt}. Output bounding
boxes in the following format:
[(confidence_score, 'object_name',
[x_min, y_min, x_max, y_max]), ... ].
```

We then parse the output to JSON format and filter out malformed entries. Compared to the open-vocabulary detector used in ECoT, this unified detection approach provides: (i) higher accuracy, (ii) more consistent naming, and (iii) fewer model calls, leading to a substantially simpler and more reliable annotation pipeline.

Step 3: Gripper Trajectory Extraction. ECoT [22] extracts gripper trajectories using per-frame detection and segmentation, which is vulnerable to noise and causes large fluctuations when detection quality varies. Instead, we adopt the tracking model [5]: (1) we detect the gripper center in the first frame, (2) create four additional tracking points around it (up, down, left, right), and (3) track all positive points jointly. This multi-point strategy yields smooth and robust trajectories, avoiding detection-induced jitter and preserving essential temporal consistency for motion reasoning.

Step 4: Generating Full Chain-of-Thought Reasoning. Finally, we combine the caption (Step 1), bounding boxes (Step 2), and extracted trajectory (Step 3), and feed them into Qwen3-VL to generate full embodied chain-of-thought reasoning. We enforce strict formatting to obtain the six reasoning tags required by ECoT (task, plan, subtask, subtask_reason, move, move_reason):

```
 The reasoning string for each step
MUST be formatted as: step_id:
<task>...</task> <plan>...</plan>
<subtask>...</subtask>
<subtask_reason>...</subtask_reason>
<move>...</move>
<move_reason>...</move_reason>
without quotation marks. There must
be exactly six tags in this order,
and each must be closed with the
correct tag name.
```

Method	Reasoning Pattern	Success Rate(%)
CogACT [10]	w/o CoT	61.5
	Full CoT	56.3
	Partial CoT	62.3
	Full CoT @ Fixed Intervals	59.5
	Partial CoT @ Fixed Intervals	65.6
TRM-VLA	Hierarch CoT @ Keyframe	**72.9**

Table 1. Comparison of different reasoning patterns on SIMPLER.

Final Output. We integrate all components—scene caption, object detections, trajectory, and six reasoning fields—to produce a complete **8-field embodied reasoning annotation** for every timestep. This redesigned pipeline yields higher annotation accuracy, reduced noise, and improved temporal consistency, forming the foundation for our KTR and GCM modules.

D. More Experimental Results

D.1. Effectiveness of Different Reasoning Design.

As shown in Table 1, to investigate the impact of different chain-of-thought reasoning patterns on performance, we conduct an ablation study on the SIMPLER benchmark [11], using CogACT [10] as the base model. All models are trained for 20K iterations on the BridgeData V2 dataset [19].

w/o CoT: The baseline model that directly regresses actions without any intermediate reasoning — identical to the original CogACT. Full CoT: Generates a complete reasoning trace containing all semantic tags (e.g., task, plan, subtask, perception, gripper) at every timestep. Partial CoT: Only generates a subset of reasoning tags — in this case, primarily `subtask` and `move`. Fixed Interval Triggering: CoT is generated only at fixed temporal intervals (every 12 frames), with the reasoning module frozen during non-triggered steps to avoid interference. Hierarch CoT and Keyframe: Our TRM framework generates CoT reasoning most relevant to the current scene only at the critical timestep.

Full CoT degrades performance (56.3%) compared to w/o CoT (61.5%), suggesting that generating full, redundant reasoning at every frame introduces *context pollution* — overwhelming the policy with irrelevant or noisy information, thereby impairing action generation. Partial CoT improves performance (62.3%), indicating that selective,

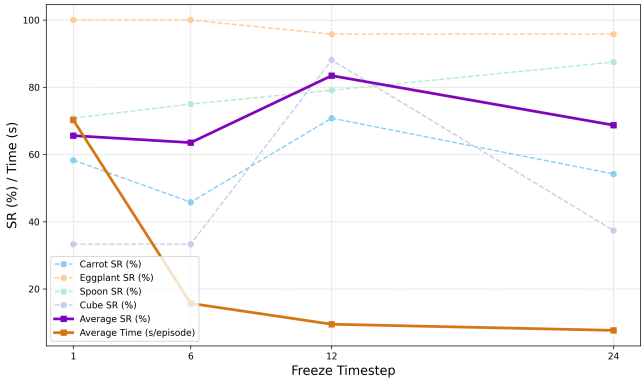

Figure 4. The impact of freeze timestep T on SIMPLER.

task-focused reasoning enhances decision quality by filtering out unnecessary details. Fixed-interval triggering further boosts performance for both Full and Partial CoT (up to 65.6% for Partial CoT), likely because it enforces temporal consistency within each interval, allowing the model to maintain stable execution without frequent reasoning updates. TRM-VLA achieves the best performance (72.9%) by combining *hierarchical* CoT with *keyframe-triggered* generation. This design avoids redundancy, preserves global context via memory, and aligns reasoning with critical decision points — validating our core hypothesis that *sparse, structured, and temporally grounded reasoning* is key to long-horizon robotic control.

These findings align with our real-world experiments (Table 3 in main text), where TRM-VLA also significantly outperforms Full/Partial CoT baselines — confirming that excessive or unstructured reasoning harms performance, while strategic, memory-augmented reasoning enhances both efficiency and robustness.

D.2. Analysis of Freeze Timestep Ablation.

As mentioned in Section C.4, we study the effect of the reasoning freeze interval T on inference efficiency and task success on SIMPLER. As shown in Figure. 4, decreasing the frequency of reasoning regeneration substantially accelerates inference, but an excessively large freeze window may lead to outdated reasoning and degraded performance.

When $T = 1$, the model regenerates reasoning at every frame, resulting in the highest computational cost (60–80s per episode). Although the success rate is competitive for some tasks, the overall system is prohibitively slow for real-time deployment. Increasing to $T = 6$ yields a large speedup (15–17s per episode), but also introduces noticeable drops in success rate for challenging tasks such as Carrot and Cube, indicating that frequent partial freezing still incurs non-trivial overhead without ensuring temporal stability. The best trade-off is achieved at $T = 12$. With this setting, inference becomes up to $8\times$ faster than $T = 1$, and the per-frame latency falls below 0.1s. Meanwhile, success rates remain high across all tasks, achieving an average of **72.9%**, substantially outperforming $T = 6$ and approaching or surpassing the performance obtained at $T = 1$. This demonstrates that a 12-frame freeze window is long enough to remove redundant reasoning computation while remaining short enough to keep reasoning information up-to-date as the environment evolves. Further increasing the interval to $T = 24$ yields additional speed gains but leads to clear deterioration in success rates for several tasks, evidencing the negative impact of stale reasoning. Overall, these results validate $T = 12$ **as the optimal balance** between computational efficiency and decision accuracy, and we adopt this setting in all main experiments.

Method	VLM	Action Expert	Success Rate(%)
CogACT [10]	CogACT	DiT-Large	61.4
	CogACT	DiT-Base	61.5
TRM-VLA	Openvla [7]	w/o	65.4
	CogACT	DiT-Large	69.6
	CogACT	DiT-Base	72.9

Table 2. Effect of Pretrained Models on SIMPLER.

D.3. Effect of Pretrained Models.

Table 2 summarizes the influence of different pretrained components on SIMPLER. For the CogACT baseline, using the CogACT-pretrained VLM together with either the DiT-Large or DiT-Base pretrained Action Expert yields similar performance (61.4% vs. 61.5%), indicating that scaling the Action Expert alone provides limited benefit without stronger temporal reasoning. Replacing the VLM with the OpenVLA-pretrained checkpoint, even without a pretrained Action Expert, improves performance to 65.4%, showing that pretrained multimodal representations are more critical than Action Expert initialization. For our full TRM-VLA model, combining temporal-aware reasoning with CogACT-pretrained components results in substantial gains. Using the CogACT VLM with the DiT-Large Action Expert achieves 69.6%, and switching to the DiT-Base pretrained expert further increases the success rate to 72.9%, the best among all settings. These findings demonstrate that: (i) temporal-aware reasoning (KTR + GCM) amplifies the benefit of pretrained VLM and Action Expert, (ii) TRM-VLA leverages pretrained representations more effectively than the CogACT baseline, and (iii) the model becomes less sensitive to Action Expert scale when equipped with structured temporal reasoning. Overall, fully pretrained TRM-VLA offers the strongest combination of robustness and performance.

D.4. Qualitative Results of GCM

To better understand how our GCM module maintains and updates historical reasoning across long-horizon tasks, we

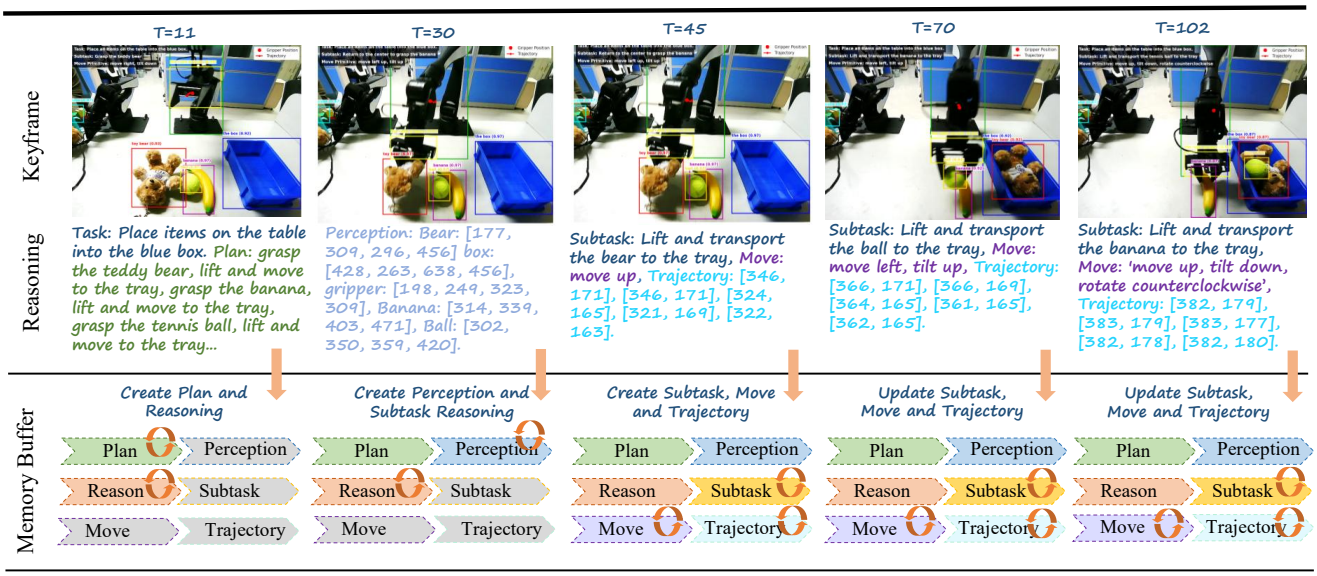

Figure 5. The visualization of how GCM's memory buffer stores and updates reasoning in the task of Clean the Table.

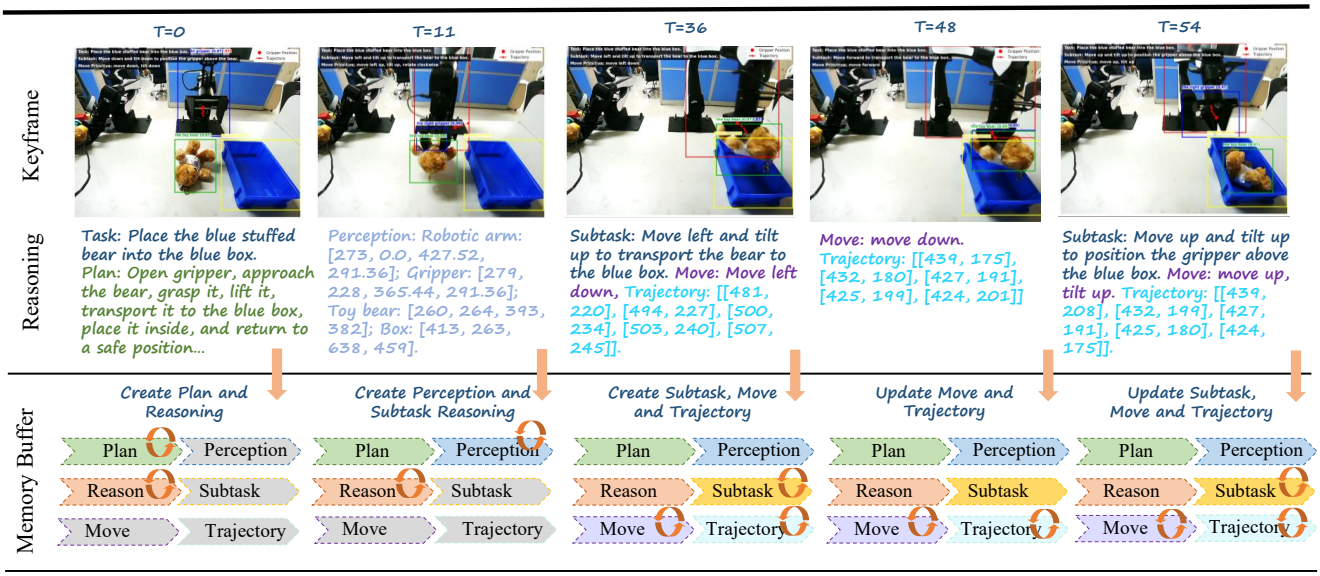

Figure 6. The visualization of how GCM's memory buffer stores and updates reasoning in the task of Place the Toy Bear.

visualize the reasoning outputs and memory buffer over four real-world episode in Figure. 5, 6, 8, 7. These examples illustrates how hierarchical reasoning evolves over time and how GCM selectively preserves, replaces, and retrieves reasoning to guide action generation.

At the beginning of the episode (T=11), the model generates high-level planning signals such as Task and Plan, which specify the global objective ("place objects into the blue box") and a sequence of intended sub-goals. GCM stores these high-level entries into the memory buffer, where they remain unchanged throughout the entire trajectory. This behavior demonstrates the long-lifespan design of high-level reasoning: since the global plan is stable across the task, keeping it persistently available enables the model to maintain coherent long-horizon behavior without regenerating redundant planning tokens at every timestep. As the episode progresses to T=30, the model produces Perception and Subtask reasoning, including localized bounding boxes for the bear, banana, cup, and gripper. These entries depend heavily on the current visual state; therefore, GCM updates them more frequently. When new perception reasoning is generated, the buffer replaces only the corresponding tag while retaining other stable reasoning entries. This validates the mid-lifespan mechanism: perception and subtask reasoning

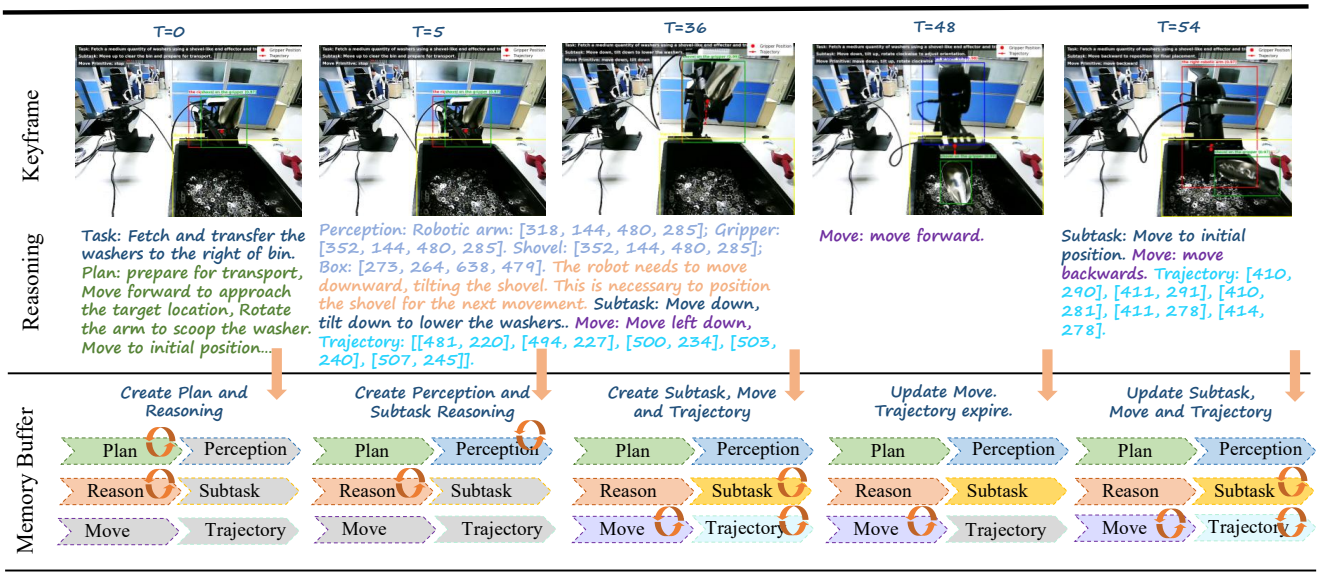

Figure 7. The visualization of how GCM's memory buffer stores and updates reasoning in the task of Scoop the Washers.

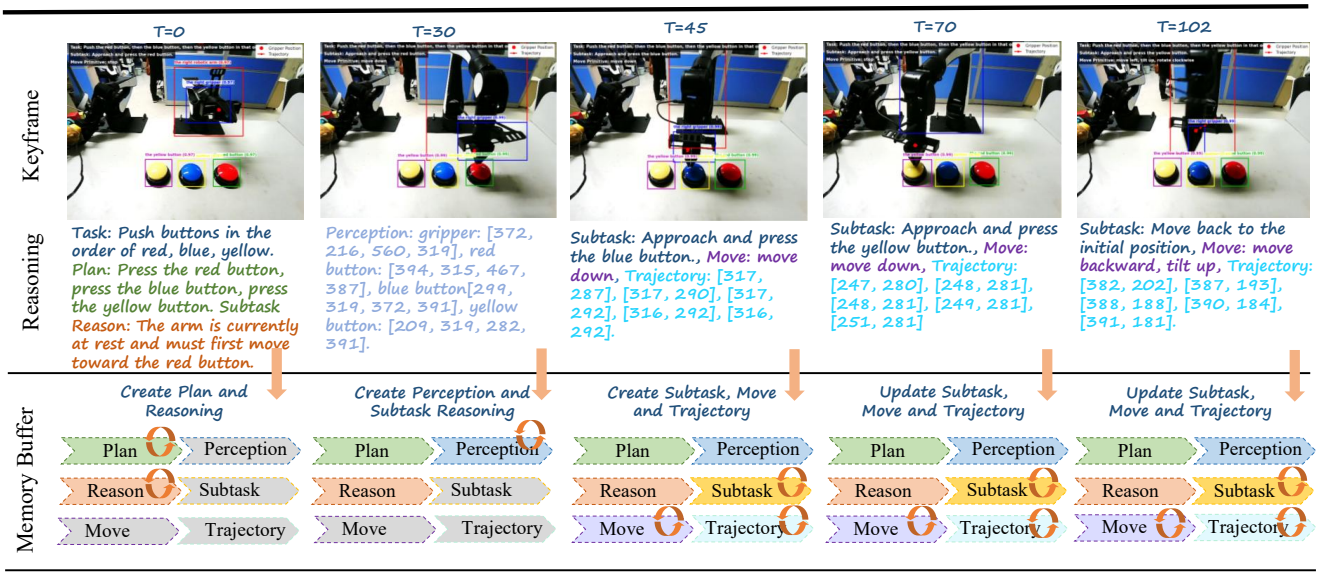

Figure 8. The visualization of how GCM's memory buffer stores and updates reasoning in the task of Push the buttons.

persist long enough to maintain temporal coherence across several frames but are refreshed when meaningful scene changes occur (e.g., when the robot begins manipulating a new object). At T=45, the model outputs Subtask ("lift and transport the bear") and Move ("move up") together with a detailed Trajectory sequence. These low-level components reflect fine-grained control signals directly influencing motion execution. Accordingly, GCM stores these entries with the shortest lifespan, updating them frequently as the robot makes incremental progress. The visualization shows clear transitions: the bear-related subtask at T=45 is replaced at T=70 when the robot switches to manipulating the ball, and again at T=102 when handling the banana. This frequent refreshing ensures that action predictions always align with the most recent physical state and manipulation target.

As shown in Figure. 8, the Push Button task explicitly evaluates temporal ordering ("red → blue → yellow"). At T=0, GCM stores the Task and Plan reasoning encoding the full symbolic order. This reasoning persists for the entire episode, acting as reliable long-term memory that guides sequential decision-making. At T=30, perception reasoning identifies the positions of the three buttons and the gripper. GCM updates only the perception slots, allowing the model to maintain symbolic ordering while refreshing spatial

grounding. From T=45 to T=102, the model successively updates Subtask and Move reasoning to reflect each completed button press. The memory buffer shows the correct transition from pressing red → pressing blue → pressing yellow, proving that GCM enables the model to remember which buttons have been pressed, update only the active subtask and maintain correct symbolic context across the full sequence. This behavior is crucial for multi-step tasks where prior actions influence future decisions. Without temporal memory, VLAs often forget earlier steps or act reactively, leading to incorrect ordering. GCM avoids these failure by providing persistent memory and localized updates.

GCM maintains a structured, tag-wise memory buffer, where each reasoning layer—(1) high-level plan, (2) mid-level perception and subtask reasoning, and (3) low-level motion and trajectory reasoning—is updated independently according to its semantic stability. This hierarchical update mechanism provides two clear advantages: 1) Reduced redundancy and more efficient reasoning. Unlike naive frame-wise CoT approaches that regenerate every reasoning token at every timestep, GCM only updates the tags that semantically change. The memory buffer effectively amortizes reasoning effort across time, resulting in significantly fewer reasoning tokens while retaining all necessary temporal context. 2) Improved temporal consistency and more accurate action generation. Because high-level intentions and mid-level perception remain accessible across multiple timesteps, action predictions become smoother and more temporally grounded. Overall, the visualization confirms that GCM organizes reasoning into a stable, interpretable temporal structure that evolves with the task. By maintaining persistent high-level plans, refreshing perception and subtask reasoning as needed, and frequently updating low-level motion details, GCM enables the model to "think temporally" and produce coherent multi-stage manipulation behavior.

D.5. Qualitative Results on LIBERO90.

Figure. 9 and Figure. 10 presents qualitative rollouts of our TRM-VLA on four representative LIBERO-90 tasks. These visualizations demonstrate that our model is capable of producing coherent, hierarchical, and temporally grounded reasoning across long-horizon manipulation sequences.

At the beginning of each episode, the model generates high-level *task* and *plan* reasoning that correctly summarizes the global objective (such as "pick up the book and place it on the shelf" or "grasp the frying pan and put it on the cabinet"). These high-level entries are stored in the memory buffer and remain stable throughout the entire rollout, providing persistent long-term guidance. As the episode progresses, the model selectively updates perception and subtask reasoning at key moments where the scene context changes. For instance,, the model precisely detects relevant objects (books, bottles, bowls, plates, frying pan) and updates their spatial relations. This demonstrates that TRM-VLA can perform accurate keyframe-triggered grounding, ensuring that subsequent manipulation steps are conditioned on up-to-date perceptual information. At mid- and late-stage timesteps, the model transitions to low-level motor reasoning, producing detailed *subtask*, *move*, and *trajectory* predictions that describe fine-grained actions such as "close gripper and move forward," "move up and back," or "rotate left while lowering the book." These low-level reasoning entries evolve rapidly and are updated frame-by-frame as the robot executes continuous motions. This behavior directly reflects the design of KTR (which outputs hierarchical reasoning according to temporal phases) and GCM (which maintains different lifetimes for high-, mid-, and low-level reasoning). The resulting reasoning traces align closely with the visual behavior of the robot arm, demonstrating temporally consistent action understanding.

Overall, these qualitative results highlight that TRM-VLA not only generates correct actions but also produces coherent, structured, and temporally grounded chain-of-thought explanations.

D.6. Qualitative Results on SIMPLER.

Figure. 11 and Figure. 12 provides qualitative visualizations of TRM-VLA on four representative SIMPLER tasks, covering long-horizon, multi-object, and contact-rich manipulation behaviors. Across all examples, our model exhibits clear keyframe-triggered hierarchical reasoning and temporally aligned action generation, demonstrating the effectiveness of KTR and GCM in real-world and simulator-based environments.

At the beginning of each episode, TRM-VLA consistently produces correct high-level *task* and *plan* descriptions, such as "move the can to the back right corner of the table" or "lift the yellow towel and put it into the washer." These high-level elements remain persistent in the memory buffer during the entire rollout, providing stable global objectives for the subsequent low-level behaviors. When the visual scene changes or new objects become relevant, the model accurately updates its perception reasoning by identifying key objects—cans, bowls, towels, microwave handles, drawers—together with their bounding boxes and spatial relations. This demonstrates that the KTR module successfully detects keyframes and regenerates perception reasoning only when necessary, avoiding redundant computation while maintaining grounding accuracy. As actions progress, the model transitions smoothly into mid- and low-level reasoning, generating precise *subtask*, *move*, and *trajectory* entries. In the "move the can" task, the model outputs detailed actions such as "move forward right" or "rotate counterclockwise," accompanied by fine-grained trajectory coordinates. Similarly, in tasks involving articulated objects (e.g., operating the microwave door or pulling a drawer), TRM-VLA correctly infers the ma-

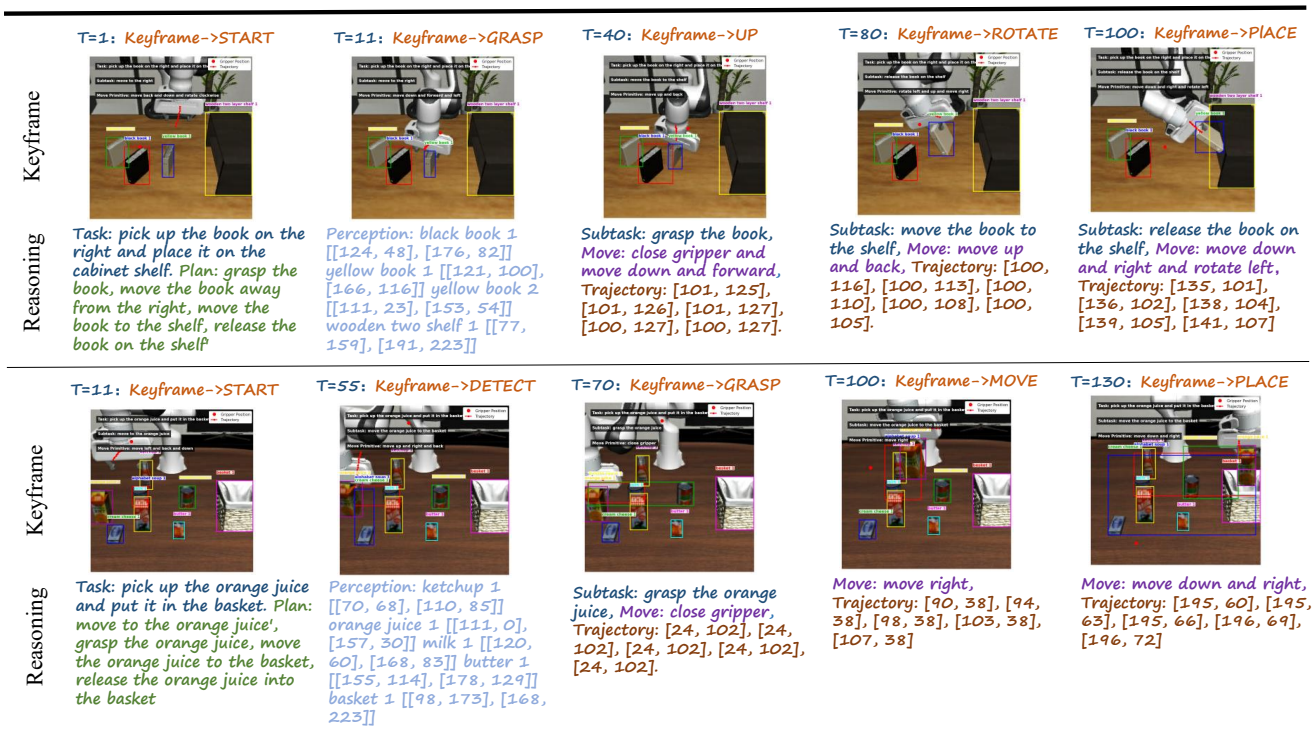

Figure 9. The visualization of reasonings on LIBERO90.

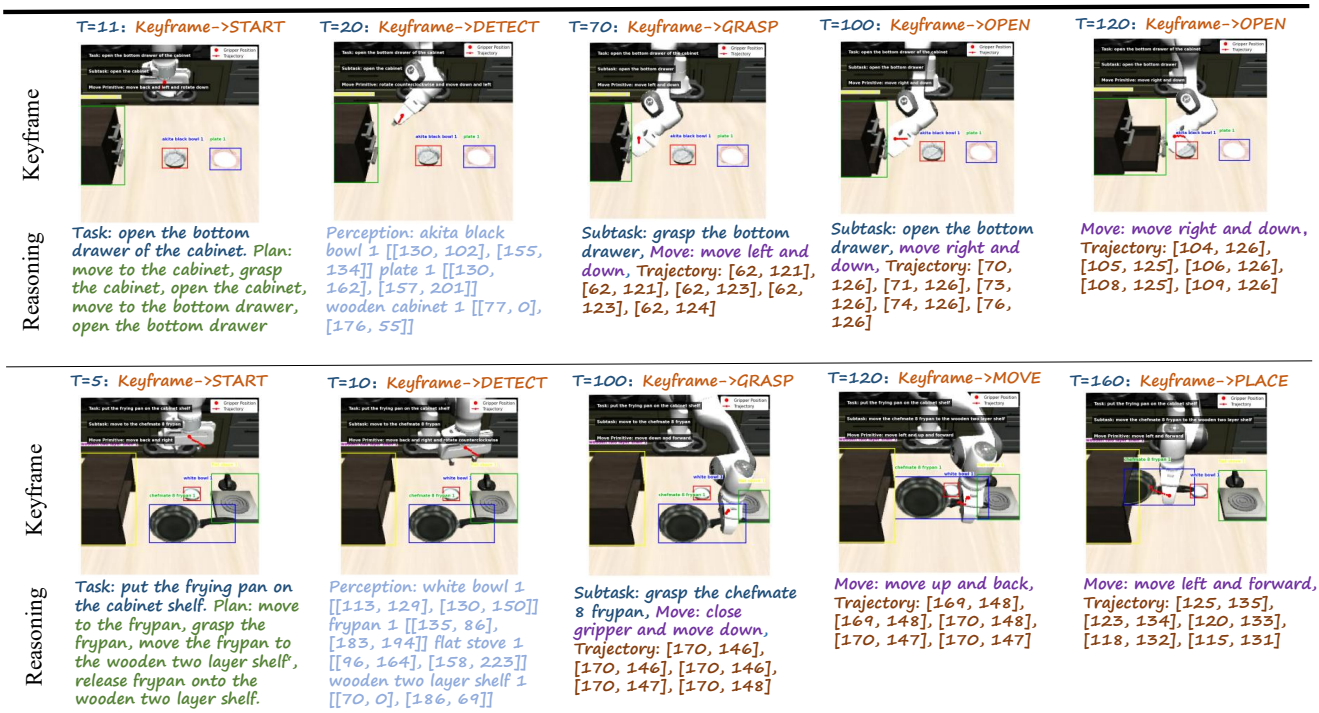

Figure 10. The visualization of reasonings on LIBERO90.

nipulation intent, producing subtask reasoning such as "turn microwave handle to close the door" or "pull the handle to open the drawer," followed by consistent motion primitives that match the robot motion.

Overall, these qualitative results demonstrate that TRM-VLA can execute multi-step manipulation tasks in SIMPLER

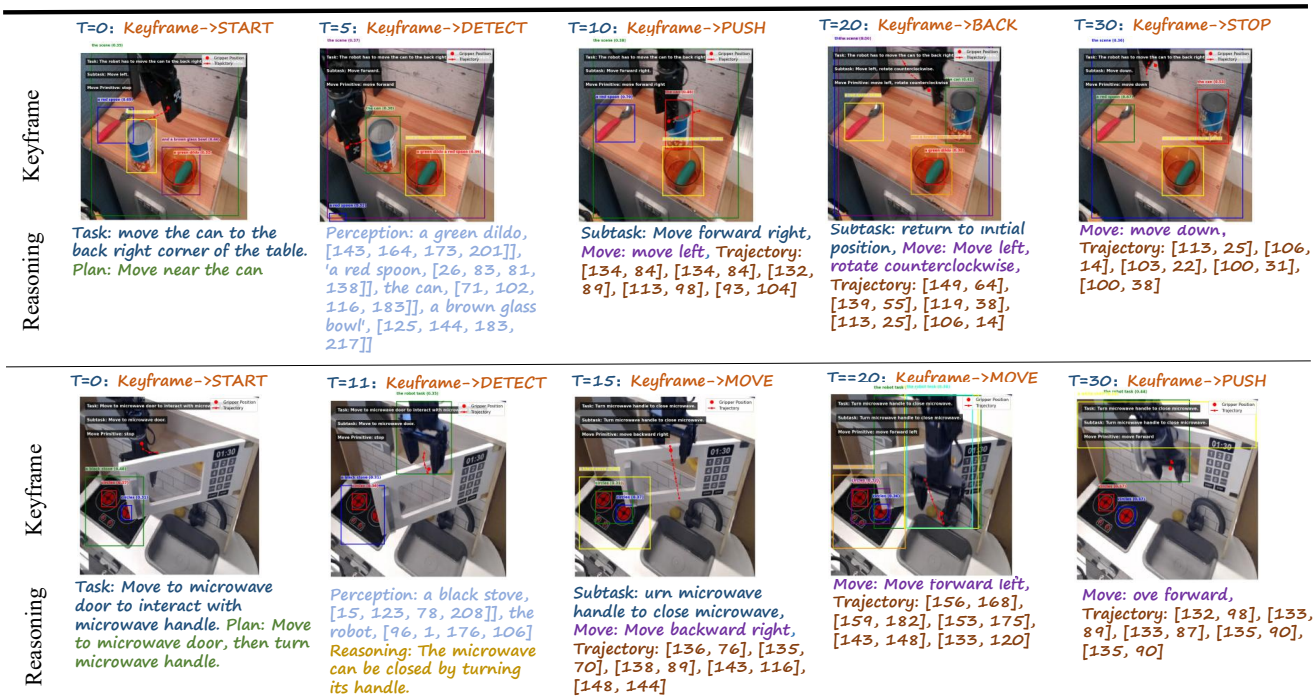

Figure 11. The visualization of reasonings on SIMPLER.

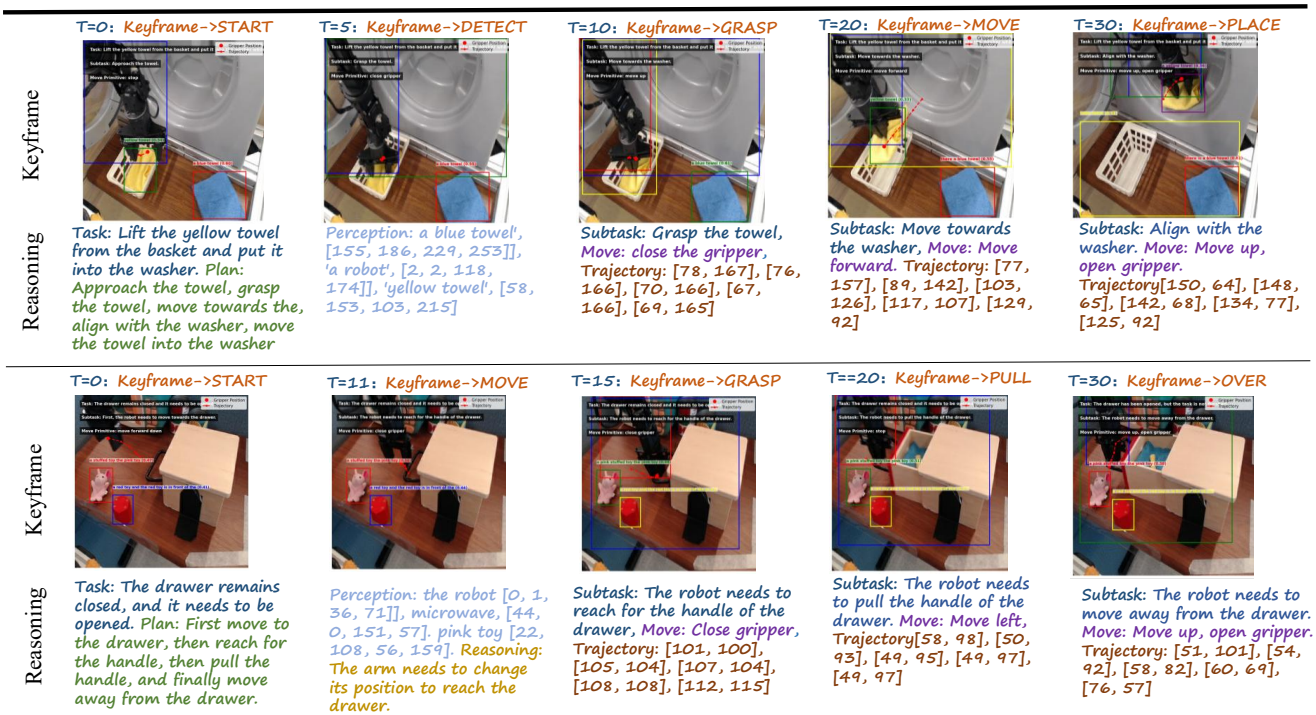

Figure 12. The visualization of reasonings on SIMPLER.

with structured and temporally consistent reasoning.

E. Discussion

Limitations. Although TRM-VLA demonstrates strong temporal-reasoning and long-horizon manipulation capabili-

ties, it still relies on large-scale robot demonstrations paired with chain-of-thought annotations. The current datasets are limited in several aspects: (i) insufficient diversity in object categories, environments, and task structures; (ii) notable variability in annotation quality, especially for fine-grained reasoning fields; and (iii) relatively short average episode lengths compared with real-world multi-stage workflows. These limitations constrain the reasoning richness the model can learn and restrict generalization to real scenarios.

Future Directions. At present, the model learns when and what reasoning to generate through supervised fine-tuning on annotated temporal reasonings. Such annotations inevitably introduce biases—both in reasoning style and in the timing of keyframe decisions—which may hinder optimal behavior. A promising future direction is to incorporate reinforcement learning to let the model self-discover effective reasoning policies. By using task success or progress as the reward, the model could actively explore (i) which forms of reasoning are genuinely beneficial for control, and (ii) when reasoning should be triggered during execution, ultimately enabling more adaptive, efficient, and emergent embodied reasoning.

F. Broader Impact

This work improves the temporal reasoning ability of robotic policies, which may enhance the reliability and transparency of real-world robot manipulation, benefiting applications such as household assistance and industrial automation. However, reasoning-augmented models remain dependent on data quality and may inherit dataset biases or fail in out-of-distribution scenarios. As with all embodied AI systems, safe deployment requires proper oversight, robust evaluation, and careful consideration of potential risks.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. arXiv preprint arXiv:2502.13923, 2025. 4

[2] William Chen, Suneel Belkhale, Suvir Mirchandani, Oier Mees, Danny Driess, Karl Pertsch, and Sergey Levine. Training strategies for efficient embodied reasoning. arXiv preprint arXiv:2505.08243, 2025. 2

[3] Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. arXiv preprint arXiv:2507.16815, 2025. 2

[4] Huiwon Jang, Sihyun Yu, Heeseung Kwon, Hojin Jeon, Younggyo Seo, and Jinwoo Shin. Contextvla: Vision-language-action model with amortized multi-frame context. arXiv preprint arXiv:2510.04246, 2025. 1

[5] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In European Conference on Computer Vision, 2024. 4, 5

[6] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. arXiv preprint arXiv:2402.07865, 2024. 5

[7] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In Conference on Robot Learning, 2024. 1, 6

[8] Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Youngyo Seo, and Jinwoo Shin. Hamlet: Switch your vision-language-action model into a history-aware policy. arXiv preprint arXiv:2510.00695, 2025. 1

[9] Hao Li, Shuai Yang, Yilun Chen, Yang Tian, Xiaoda Yang, Xinyi Chen, Hanqing Wang, Tai Wang, Feng Zhao, Dahua Lin, et al. Cronusvla: Transferring latent motion across time for multi-frame prediction in manipulation. arXiv preprint arXiv:2506.19816, 2025. 1

[10] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. arXiv preprint arXiv:2411.19650, 2024. 3, 5, 6

[11] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. arXiv preprint arXiv:2405.05941, 2024. 5

[12] Fanqi Lin, Ruiqian Nai, Yingdong Hu, Jiacheng You, Junming Zhao, and Yang Gao. Onetwovla: A unified vision-language-action model with adaptive reasoning. arXiv preprint arXiv:2505.11917, 2025. 2

[13] Chenghao Liu, Jiachen Zhang, Chengxuan Li, Zhimu Zhou, Shixin Wu, Songfang Huang, and Huiling Duan. Ttf-vla: Temporal token fusion via pixel-attention integration for vision-language-action models. arXiv preprint arXiv:2508.19257, 2025. 1

[14] Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. Advances in Neural Information Processing Systems, 36, 2024. 1

[15] Delin Qu, Haoming Song, Qizhi Chen, Zhaoqing Chen, Xianqiang Gao, Xinyi Ye, Qi Lv, Modi Shi, Guanghui Ren, Cheng Ruan, Maoqing Yao, Haoran Yang, Jiacheng Bao, Bin Zhao, and Dong Wang. Embodiedonevision: Interleaved vision-text-action pretraining for general robot control. arXiv preprint arXiv:2508.21112, 2025. 2

[16] Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. arXiv preprint arXiv:2508.19236, 2025. 1

[17] Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Memer: Scaling up memory for robot control via experience retrieval. arXiv preprint arXiv:2510.20328, 2025. 1

[18] Gemini Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montse Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauzá, et al. Gemini robotics: Bringing ai into the physical world. arXiv preprint arXiv:2503.20020, 2025. 2

[19] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In Conference on Robot Learning, pages 1723–1736, 2023. 5

[20] Shuai Yang, Hao Li, Yilun Chen, Bin Wang, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation. arXiv preprint arXiv:2507.17520, 2025. 2

[21] Yifu Yuan, Haiqin Cui, Yaoting Huang, Yibin Chen, Fei Ni, Zibin Dong, Pengyi Li, Yan Zheng, and Jianye Hao. Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. arXiv preprint arXiv:2508.13998, 2025. 2

[22] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In Conference on Robot Learning, pages 3157–3181. PMLR, 2025. 1, 3, 4, 5

[23] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2025. 1

[24] Zhongyi Zhou, Yichen Zhu, Junjie Wen, Chaomin Shen, and Yi Xu. Chatvla-2: Vision-language-action model with open-world embodied reasoning from pretrained knowledge. arXiv preprint arXiv:2505.21906, 2025. 2