StreamPI: Streaming Multimodal Temporal Modeling for Vision-Language-Action Models

Zhe Liu[1*] **Jinghua Hou**[1*] **Yuxiang Lu**[1] **Zhenya Yang**[1] **Xianzhe Fan**[1] **Junwei Luo**[1]
Junyi Li[1] **Ruihua Han**[1] **Zhi Hou**[2] **Hengshuang Zhao**[1†]
[1]The University of Hong Kong [2]ACE Robotics
https://hku-sail.github.io/StreamPI/

Abstract

Vision-Language-Action (VLA) models have demonstrated effectiveness in robot manipulation, yet state-of-the-art models such as $\pi_{0.5}$ operate under a single-frame paradigm, limiting their ability to retain past observations and develop precise spatial perception. In this paper, we propose StreamPI, a streaming multimodal temporal modeling framework that equips single-frame VLA with temporal reasoning capability without introducing any additional parameters. One core design is *instruction-anchored temporal modeling*. It treats each (visual observation, language instruction) pair as an atomic temporal unit: bidirectional attention within each pair enables cross-modal fusion, while causal attention across pairs preserves autoregressive streaming inference. This ensures the language instruction serves as a persistent semantic anchor throughout task execution. To bridge the gap between synchronous training and asynchronous real-robot deployment, we introduce a *random-interval streaming training* strategy: a proper inter-frame interval (e.g., every 3 frames) enables faster and smoother action execution. Beyond this, randomizing the interval further improves robustness to frame-timing perturbations, supporting asynchronous deployment in practice. Furthermore, by leveraging the length extrapolation capability of the LLM backbone, StreamPI seamlessly inherits pretrained single-frame weights and supports flexible single-frame and multi-frame inference. Experiments on real-robot tasks spanning memory-dependent and precise perception scenarios, as well as the simulation benchmark LIBERO, demonstrate that StreamPI outperforms $\pi_{0.5}$ across diverse tasks.

1 Introduction

Vision-Language-Action (VLA) models have emerged as a promising paradigm for generalizable robot manipulation, unifying perception, language understanding, and action generation within a single end-to-end framework [56, 18, 3, 16]. Despite their impressive performance, state-of-the-art VLA models such as π_0 and $\pi_{0.5}$ operate under a single-frame paradigm, as illustrated in Figure 1(a), where each action is predicted from a single image observation without access to any historical context. This design inherently precludes two critical capabilities, namely the ability to memorize and reason over past observations, and the capacity to develop precise spatial perception that emerges from temporal aggregation.

Incorporating temporal context addresses both limitations simultaneously. On the memory side, access to historical observations enables robots to resolve tasks that are inherently ambiguous from a single frame, such as inferring which cup conceals a target object or grasping a dynamically moving target. On the perception side, the benefit of temporal modeling for spatial understanding

*Equal contribution.
†Corresponding author.

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

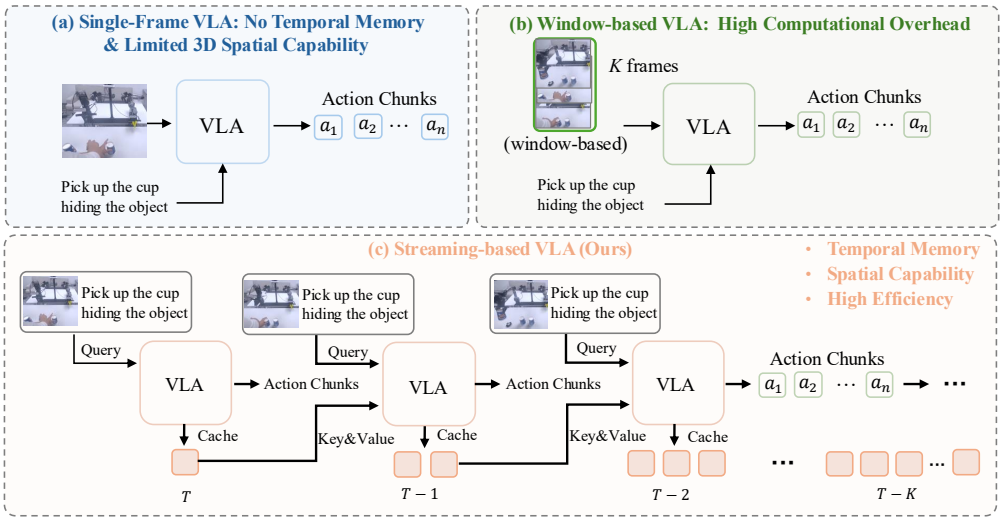

Figure 1: Comparison of three paradigms for VLA-based robot manipulation. **(a) Single-frame VLA:** Only the current observation is processed, lacking historical context and limiting temporal memory and spatial perception. **(b) Window-based VLA:** A window of K frames is processed simultaneously, enriching temporal context at the cost of high computational overhead. **(c) Streaming-based VLA (Ours):** The model queries cached Key&Value representations from previous timesteps via a lightweight KV cache, achieving temporal memory and precise spatial perception.

has been well established in autonomous driving, where multi-frame fusion methods [15, 23, 42, 14] dramatically improve 3D scene understanding, and VGGT [41, 55] series demonstrate that temporal aggregation yields substantially more accurate depth estimation and 3D reconstruction. These findings transfer naturally to embodied manipulation, where strong spatial perception is fundamental to precise object grasping and placement. To this goal, shown in Figure 1(b), some multiple-frame VLA methods [47, 39] adopt window-based visual inputs for temporal modeling. To reduce inference time, such approaches typically resort to either a smaller backbone network or a dedicated video encoder that compresses multi-frame observations into a reduced set of tokens.

Therefore, effectively incorporating temporal information into strong single-frame VLA foundation models such as $\pi_{0.5}$ is non-trivial. There are four potential challenges. **1) Computational overhead:** Concatenating frames across time causes sequence length to grow linearly, rendering inference latency prohibitive for real-time control. **2) Instruction forgetting:** Conventional VLA models inject the language instruction as a fixed set of text tokens. As the temporal horizon expands, accumulating visual tokens progressively dilute the influence of instruction tokens, causing the model to lose track of the task goal over long horizons. **3) Training-deployment mismatch:** Training relies on regularly sampled frame sequences, whereas real-robot deployment produces asynchronous observation streams with variable time gaps, degrading model robustness in online settings. **4) Representation corruption:** Introducing a video encoder adds new parameters whose feature distributions are misaligned with the powerful VLA pretrained models, risking corruption of the rich visual-language features of the base model in the embodied domain.

To address these challenges, we propose StreamPI, a streaming multi-modal temporal modeling framework, as illustrated in Figure 1(c). StreamPI incorporates four key designs to enable efficient and robust temporal reasoning. **1) Streaming inference:** Rather than processing a full observation window at every step, StreamPI adopts a streaming temporal modeling paradigm that caches Key&Value representations from past timesteps, keeping inference cost constant regardless of temporal horizon. **2) Instruction-anchored temporal modeling:** Rather than modeling temporal dependencies over visual frames alone, StreamPI treats each (visual observation, language instruction) pair as an atomic temporal unit. The language instruction is persistently coupled with every observation as a semantic anchor throughout execution. Within each pair, bidirectional attention enables thorough cross-modal fusion, while causal attention across pairs preserves the autoregressive structure for online streaming inference, ensuring the model consistently maintains awareness of the current task goal. **3) Random-interval streaming training:** To improve robustness to variable frame rates and asynchronous observation arrival at deployment time, StreamPI adopts a random-interval

streaming training strategy that exposes the model to diverse frame-timing perturbations during training. **4) No additional parameters:** By leveraging the length extrapolation capability of the LLM backbone, StreamPI seamlessly inherits all pretrained weights of $\pi_{0.5}$ without introducing any new parameters, fully preserving its representational integrity while naturally supporting both single-frame and multi-frame inference at test time.

Finally, we evaluate StreamPI on real-robot tasks spanning *precise perception-dependent tasks* and *memory-dependent tasks* and consistently outperforms $\pi_{0.5}$. On the LIBERO simulation benchmark [24], StreamPI achieves superior performance, verifying the effectiveness of our approach.

In summary, our main contributions are as follows:

- We propose StreamPI, a streaming multi-modal temporal modeling VLA framework that treats each (visual observation, language instruction) pair as an atomic temporal unit with instruction-anchored modeling to maintain persistent task awareness during execution.
- We introduce a random-interval streaming training strategy that exposes the model to diverse frame-timing perturbations, bridging the gap between synchronous training and asynchronous real-robot deployment. Besides, StreamPI seamlessly inherits all pretrained weights of $\pi_{0.5}$ via LLM length extrapolation and supports flexible single-frame and multi-frame inference.
- Extensive experiments on real-robot manipulation and the LIBERO benchmark demonstrate that StreamPI consistently outperforms $\pi_{0.5}$ on both spatial-precision and memory-dependent tasks.

2 Related Work

Vision-Language-Action Models. The emergence of large-scale pre-trained vision-language models (VLMs) has catalyzed a new generation of robot policies that unify perception, language understanding, and action generation within a single framework [56, 18, 3, 13, 38, 16, 30, 53, 4, 43, 45, 44, 6, 32, 25, 5, 50]. RT-2 [56] pioneered this direction by co-finetuning a VLM on robot demonstration data, demonstrating that web-scale visual-linguistic knowledge can be directly transferred to robotic control. OpenVLA [18] extended this paradigm with an open-source framework, enabling broader community adoption and systematic study of VLA design choices. More recently, π_0 [3] introduced a flow-matching action head decoupled from the VLM backbone, achieving high-frequency dexterous control while preserving the semantic reasoning capabilities of the language model. $\pi_{0.5}$ [16] further scales this approach with improved data diversity and task generalization. Despite these advances, existing VLA models predominantly operate in a single-frame inference paradigm, processing each observation independently without maintaining temporal context across time steps. This motivates our work on streaming temporal modeling for VLA inference.

Temporal Modeling for Robot Manipulation. Incorporating temporal context into robot policies has long been recognized as essential for tasks requiring spatial reasoning and long-horizon planning [8, 52]. Diffusion Policy [8] and ACT [52] demonstrate that conditioning on short observation histories significantly improves action consistency and task success rates. Recent work [47, 20, 34, 28, 49, 19, 17, 54] has explored multi-frame modeling within VLA frameworks. CronusVLA [20] systematically investigates the design space of multi-frame VLA models, showing that incorporating historical observations yields substantial gains on manipulation benchmarks requiring spatial precision. World model approaches [10, 22, 11, 37, 48, 1] further argue that temporal modeling is indispensable for agents to anticipate future states and reason about action consequences. However, these methods either introduce significant computational overhead through full attention over all historical tokens, or decouple visual observations from their corresponding language instructions during temporal aggregation, which is a design flaw that leads to instruction forgetting over long task horizons. StreamPI addresses both limitations through its instruction-anchored atomic temporal unit design, which preserves cross-modal binding while enabling efficient causal streaming inference.

Streaming Inference. The challenge of efficient streaming inference has been extensively studied in the NLP community. StreamingLLM [46] identifies the "attention sink" phenomenon and proposes retaining a small set of initial tokens alongside a sliding window of recent tokens, enabling LLMs to process infinitely long sequences without recomputation. LongLoRA [7] demonstrates that transformer models can generalize to sequence lengths far beyond those seen during training. In the robotics domain, recent work [9, 40, 27, 35] on asynchronous execution highlights the practical challenge of variable-rate observation streams, where fixed-interval assumptions break down under

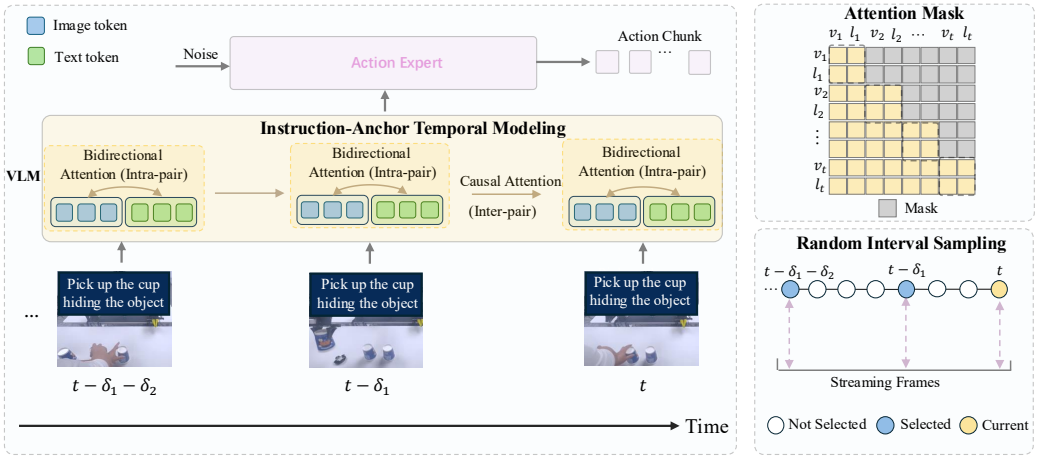

Figure 2: The pipeline of StreamPI. To fully unleash the potential of multi-modal interaction, we use bidirectional attention for image-text pairs and causal attention for inter-frames with block-wise causal attention mask. We additional use the random interval sampling to improve the temporal robustness in the real-world deployment.

real-world deployment conditions. Our random-interval streaming training strategy directly addresses this gap, exposing the model to diverse temporal spacings during training to improve robustness to the asynchronous observation streams encountered on physical robots.

3 Method

In this section, we present StreamPI, a streaming multi-modal temporal modeling framework for Vision-Language-Action models (*e.g.*, $\pi_{0.5}$), as illustrated in Figure 2. We begin by revisiting the single-frame inference paradigm of $\pi_{0.5}$ and identifying its key limitations (Sec. 3.1). We then introduce our core architectural design, which encapsulates visual observations and language instructions into *(image, text)* pairs as atomic temporal units, and organizes them through a causal attention mechanism (Sec. 3.2). Finally, we describe our random-interval streaming training strategy, which bridges the gap between training and asynchronous real-robot deployment (Sec. 3.3).

3.1 Preliminaries

$\pi_{0.5}$ is a Vision-Language-Action (VLA) model that processes a single observation at each inference step. At time t, the model receives a language instruction l and a set of multi-view visual observations $\mathbf{V}_t = \{v_t^f,\, v_t^l,\, v_t^r\}$, comprising a front-view, left-view and right-view wrist camera images, where each of $v_t^f, v_t^l, v_t^r \in \mathbb{R}^{H\times W\times 3}$. These inputs are tokenized and concatenated into a sequence: $\mathbf{x}_t = [\mathbf{V}_t, l_t]$. The model then samples an action $\mathbf{a}_t$ from the learned policy conditioned on $\mathbf{x}_t$: $\mathbf{a}_t \sim \pi_\theta(\cdot \mid \mathbf{x}_t)$, where π_θ denotes the policy parameterized by the pre-trained weights. $\pi_{0.5}$ employs a transformer backbone with full bidirectional attention over the input tokens, enabling rich cross-modal fusion between the language instruction and all visual observations within a single frame.

However, a single-frame observation provides insufficient geometric context for tasks requiring precise spatial perception, whereas temporal observations can reveal richer latent geometric cues through motion parallax and structural consistency across frames. Moreover, when extended to temporal settings, naively concatenating historical visual observations without re-anchoring each frame to the language instruction causes the instruction signal to be progressively diluted by the expanding visual token sequence. These limitations motivate our instruction-anchored streaming framework StreamPI.

3.2 Instruction-Anchored Temporal Modeling

A naive approach to temporal aggregation is to concatenate historical visual observations into a single growing sequence, which introduces two compounding problems. First, the sequence length grows linearly with the temporal horizon, making both training memory consumption and inference latency prohibitive for real-time control. Second, as visual tokens accumulate, the language instruction maybe be progressively overshadowed, causing the model to lose track of the task goal over long horizons. To address these issues, we propose instruction-anchored temporal modeling in a streaming fashion, shown in Figure 2. Specifically, each visual observation is paired with the task instruction, forming an instruction-anchored temporal unit. We apply bidirectional attention within each instruction-observation pair to capture multimodal interactions, and causal attention across pairs to model temporal dependencies.

First, we define each time step's input as an *atomic temporal unit* that jointly encodes the multi-view visual observations and the language instruction:

$$\mathbf{u}_t = (\mathbf{V}_t,\, l_t) \tag{1}$$

where l_t denotes the language instruction corresponding to time t. By treating $\mathbf{u}_t$ as an indivisible unit, the instruction remains persistently anchored to its associated visual context at each time step, preventing instruction forgetting regardless of the temporal horizon length. At the current time step, we define the streaming frames as the latest T sampling observations, including the current one. The input sequence is then constructed by concatenating T atomic temporal units:

$$\mathbf{U} = [\mathbf{u}_{t-T+1},\, \mathbf{u}_{t-T+2},\, \ldots,\, \mathbf{u}_t] \tag{2}$$

To capture both cross-modal fusion within each time step and temporal dependencies across time steps, we organize the attention structure over $\mathbf{U}$ from two levels.

Intra-pair Bidirectional Attention. Within each atomic unit $\mathbf{u}_\tau$, all tokens from $\mathbf{V}_\tau$ and l_τ attend to each other bidirectionally:

$$\mathbf{h}_\tau = \mathrm{Attn}_{\mathrm{bi}}(\mathbf{V}_\tau,\, l_\tau) \tag{3}$$

This ensures thorough cross-modal fusion between the multi-view visual observations and the language instruction at each individual time step, producing a semantically grounded representation $\mathbf{h}_\tau$ for downstream temporal reasoning.

Inter-pair Causal Attention. Across atomic units, the fused representation $\mathbf{h}_\tau$ attends to all preceding units via causal attention:

$$\mathbf{o}_t = \mathrm{Attn}_{\mathrm{causal}}(\mathbf{h}_{t-T+1},\, \ldots,\, \mathbf{h}_t) \tag{4}$$

This allows the model to aggregate temporal context from past observations in an autoregressive manner, while the causal structure ensures that future frames do not leak into past representations during training. The final output $\mathbf{o}_t$ is then used to condition action generation of $\mathbf{a}_t \sim \pi_\theta(\cdot \mid \mathbf{o}_t)$.

Parameter-free Temporal Extension. A key advantage of this design is that it introduces no additional parameters. The hierarchical attention pattern is entirely realized by restructuring the attention mask over the existing VLA backbone. The inter-pair causal attention over an extended token sequence is handled naturally by the LLM's length extrapolation capability, allowing StreamPI to inherit all pre-trained weights without modification. Specifically, we only extend the input token sequence for multi-frame inputs and assign extended position embeddings to all frame tokens. We then use our inter-pair causal mask to enforce intra-frame self-attention within each individual frame and inter-frame causal attention across consecutive frames, restricting the input to follow the temporal order strictly. This ensures that the rich vision-language representations learned during pre-training are effectively preserved.

3.3 Random-Interval Streaming Training Strategy

In real-robot deployment, observations arrive asynchronously and at variable frame rates. A model trained with fixed temporal intervals becomes brittle to such variation, since the temporal statistics at test time differ systematically from those encountered during training. We address this mismatch with a dedicated training strategy that improves robustness to temporal irregularity.

Random-interval Sampling. During training, rather than sampling streaming frames at a strictly fixed interval, we introduce a random interval to improve robustness. Concretely, given a base inter-frame interval $\bar{\delta}$, we add a random perturbation $\epsilon \sim \mathcal{U}(-\Delta, +\Delta)$ at each sampling step, yielding a perturbed interval $\delta = \bar{\delta} + \epsilon$, which is clipped to $[\delta_{\min}, \delta_{\max}]$. The T streaming frames are then sampled from the streaming buffer at steps of δ. This exposes the model to a diverse distribution of temporal spacings around the nominal interval, preventing it from over-relying on fixed-interval temporal cues and improving generalization to the asynchronous observation rates encountered during real-robot deployment. In the bottom-right corner of Figure 2, a schematic illustrates the proposed Random-Interval Sampling strategy, where historical frames are selected at randomized offsets $t-\delta_1$ and $t-\delta_1-\delta_2$ relative to the current frame t, rather than at fixed intervals.

Temporal Masking. To further align training with streaming inference, we adopt a temporal masking strategy. Given a full sequence of T streaming frames, we randomly sample a masking count $k \in \{0, 1, \ldots, T-1\}$. When $k = 0$, the complete sequence is visible to the model. When $k > 0$, the earliest k frames are masked, leaving only the most recent $T - k$ frames accessible. Combined with causal attention masking, this simulates the incremental observation pattern of streaming inference.

Streaming Inference. With our streaming design, StreamPI enables strong temporal modeling without introducing significant additional computational overhead. Given a sequential stream of observations at timestamps $\{t_0, t_1, \ldots, t_N\}$, at the initial timestamp t_0, the model takes the current frame $\mathbf{u}_{t_0}$ as input, predicts the action $\mathbf{a}_{t_0}$, and stores the resulting fused representation $\mathbf{h}_{t_0}$ in the KV-Cache. At each subsequent timestamp t_n ($n > 0$), only the newly arriving frame $\mathbf{u}_{t_n}$ needs to be encoded. Its representation $\mathbf{h}_{t_n}$ then attends to the cached historical representations $\{\mathbf{h}_{t_0}, \ldots, \mathbf{h}_{t_{n-1}}\}$ via cross-attention, eliminating redundant re-computation over past frames. After each step, the KV-Cache is updated by appending $\mathbf{h}_{t_n}$, which is directly reused for temporal modeling at all future timestamps. This design eliminates redundant recomputation of past frames and makes StreamPI well-suited for long-horizon robot manipulation tasks.

4 Experiments

To comprehensively evaluate StreamPI, we organize experiments around four core questions: (1) Can it handle both memory-dependent and precise perception-dependent manipulation tasks on real robots? (Sec. 4.2) (2) How does StreamPI compare with the advanced single-frame VLA methods on the LIBERO simulation benchmark? (Sec. 4.3) (3) How does each design choice contribute to overall performance? (Sec. 4.5) (4) How robust is StreamPI under varying temporal sampling intervals and cross frames? (Sec. 4.5)

4.1 Experimental Details

Benchmark. We evaluate StreamPI on the LIBERO benchmark [24], which comprises four task suites of increasing complexity: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, each containing 10 tasks with 50 trials per task. We also evaluate StreamPI on the CALVIN benchmark [29] to demonstrate its temporal modeling capacity. We follow the standard evaluation protocol and report success rates averaged as the evaluation metric. Besides, we provide details of the real-robot setup for our real-world experiments in the technical appendices.

Training & Inference. StreamPI is built on a pre-trained Vision-Language-Action model $\pi_{0.5}$ [16], and keep the same settings including learning rate and optimizer. We fully fine-tune the pre-trained weights and introduce no additional parameters and temporal modeling is achieved through the attention mask. Specifically, we adopt the same optimizer and learning rate schedule as $\pi_{0.5}$. The number of streaming frames is set to T=3/5 during training. For random-interval streaming training, the inter-frame interval δ is sampled uniformly from $[\delta_{\min}, \delta_{\max}] = [3, 7]$ at each training step. All experiments are conducted on 8 NVIDIA H100 GPUs with a batch size of 256 for LIBERO benchmark with 30k iterations and with a batch size of 128 for all real-robot tasks with 50k iterations. At inference time, StreamPI operates in a streaming manner, maintaining a rolling buffer of the T most recent observation-instruction pairs with the interval of δ. The inter-frame interval is fixed at δ=5 for simulation and sampled uniformly from $\delta \sim \mathcal{U}[3, 7]$ for real-robot deployment.



Figure 3: Visualization of real-world tasks and comparison of real-world performance. **Left**: Visualization of memory-dependent and precise perception-dependent real-world tasks. **Right**: Performance comparison of real-world tasks between $\pi_{0.5}$ and StreamPI.

4.2 Real-Robot Manipulation

Complex real-robot manipulation usually requires both precise geometric perception and persistent temporal awareness across interaction steps. To verify the effectiveness of StreamPI in addressing these demands, we conduct real-robot experiments spanning two complementary task categories: 1) *Memory-Dependent Tasks*, which require the robot to retain and act upon contextual cues accumulated over long horizons; 2) *Precise Perception-Dependent Tasks*, which focus on the model's ability to perceive fine-grained geometric relationships from multi-frame observations. Each category consists of two tasks, evaluated with success rate (%).

Memory-Dependent Tasks. These tasks require the robot to recall information from earlier observations to complete a later step, a capability fundamentally beyond the reach of single-frame inference. As shown in Figure 3(a) and (b), we evaluate on two specific tasks. *i) Shell Game*: following a shell-game protocol, the robot first observes an object being concealed beneath one of several cups, after which the cups are shuffled. The robot then retrieves the correct cup by recalling which one concealed the object, a task that requires cross-frame memory. StreamPI achieves substantial improvements on both tasks (+**36.6**% on Rolling Object Grasping and +**33.3**% on Cup Hiding and Retrieval), demonstrating that instruction-anchored temporal modeling effectively preserves task-relevant memory across extended horizons. *ii) Rolling Object Grasping*: the robot need to track a continuously moving object across frames and grasp it at the appropriate moment. Since the object's position shifts over time, a single frame provides no motion context and thus cannot support reliable interception. For more details about the tasks, please refer to our appendix.

Precise Perception-Dependent Tasks. These tasks require accurate 3D perception from single-frame observations, where a single frame provides insufficient geometric context for reliable execution. We evaluate on two specific tasks. *i) Pen Insertion into Narrow Bottle*: the robot need to guide a pen into a narrow bottle opening, demanding sub-centimeter spatial precision that a single frame cannot reliably provide due to the absence of multi-view depth cues. *ii) Cup Insertion into Cup Sleeve*: the robot need to precisely align and insert a cup into a designated cup sleeve, where accurate depth estimation and pose alignment across frames are essential to avoid misplacement. As shown in Figure 3(c) and (d), StreamPI consistently outperforms the single-frame $\pi_{0.5}$ baseline across both tasks +**26.7**% on Pen Insertion into Narrow Bottle) and (+**32.0**% on Cup Insertion into Cup Sleeve, confirming that temporal aggregation provides critical geometric cues unavailable from any single observation. For more details about the tasks, please refer to our appendix.

Table 1: Performance comparison on LIBERO [24]. Success rates (%) are reported across four suites.

Method	Libero-Spatial	Libero-Object	Libero-Goal	Libero-Long	Avg. Success Rate
Diffusion Policy [8]	78.3	92.5	68.3	50.5	72.4
Octo [12]	78.9	85.7	84.6	51.1	75.1
SpatialVLA [32]	88.2	89.9	78.6	55.5	71.7
TraceVLA [54]	84.6	85.2	75.1	54.1	74.8
OpenVLA [18]	84.7	88.4	79.2	53.7	75.9
CoT-VLA [51]	87.5	91.6	87.6	69.0	81.1
π_0-FAST* [31]	96.4	96.8	88.6	60.2	85.0
SmolVLA [36]	93.0	94.0	91.0	77.0	88.8
GR00T-N1 [2]	94.4	97.6	93.0	90.6	93.9
UniVLA [4]	95.4	98.8	93.6	94.0	95.4
FLOWER [33]	97.1	96.7	95.6	93.5	95.7
CronusVLA [20]	90.1	94.7	91.3	68.7	86.2
TriVLA [26]	91.2	93.8	89.8	73.2	87.0
4D-VLA [49]	93.8	92.8	95.6	86.5	92.2
CogACT [21]	87.5	90.2	80.2	53.2	77.8
ST-π [28]	98.4	98.3	96.9	94.3	97.3
MemoryVLA [34]	98.4	98.4	96.4	93.4	96.5
π_0 [3]	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$ [16]	98.8	98.2	96.8	92.4	96.9
StreamPI ($T = 3$)	**98.6**	98.6	98.6	93.8	97.5
StreamPI ($T = 5$)	**98.8**	**99.8**	**99.6**	**95.0**	**98.3**

Table 2: Performance comparison on CALVIN [29]. Success rates (%) at each sequence position and average sequence length are reported.

Method	1	2	3	4	5	Avg. Seq. Len
MemoryVLA [34]	94.8	87.4	81.4	75.9	69.4	4.090
$\pi_{0.5}$ [16]	94.2	88.7	85.7	83.2	79.5	4.313
StreamPI ($T = 5$)	**96.9**	**93.6**	**90.7**	**88.5**	**85.0**	**4.547**

4.3 LIBERO Simulation Benchmark

Beyond real-robot experiments, we further validate StreamPI on the LIBERO benchmark [24], a widely adopted simulation suite comprising four task suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long. This enables a systematic comparison against existing VLA methods under standardized evaluation conditions. We place particular emphasis on **LIBERO-Long**, which requires executing sequences of 2 or more sub-tasks and most directly tests long-horizon temporal memory, aligning closely with the memory-dependent challenges addressed by StreamPI. As shown in Table 1, the LIBERO benchmark is largely saturated, with the baseline already exceeding 95% success rate on most suites, making further gains increasingly difficult to obtain. Nevertheless, StreamPI achieves an average improvement of 1.4% over the single-frame $\pi_{0.5}$ baseline across all four suites, demonstrating the effectiveness of temporal reasoning even in near-saturated regimes. The gain is most pronounced on LIBERO-Long (**+2.6**%), where the lack of temporal memory is particularly detrimental. Notably, although a single frame already provides sufficient context for many LIBERO-Goal tasks, StreamPI still achieves a promising gain of 2.8% on LIBERO-Goal, suggesting that the benefits of multi-frame modeling extend beyond memory-dependent tasks to perception-heavy scenarios involving dynamic object interactions and subtle state transitions. We observe no improvement on LIBERO-Spatial, where task success is often determined by static spatial relations that are already well captured in a single frame. In such cases, additional temporal context may introduce intermediate motion cues that are less relevant to the final geometric configuration. Overall, these results confirm that the temporal reasoning capability introduced by StreamPI generalizes robustly from real-robot manipulation to simulation environments.

4.4 CALVIN Benchmark

To further evaluate the long-horizon temporal reasoning capabilities of StreamPI, we conduct experiments on the CALVIN benchmark [29], a challenging multi-task robotic manipulation suite that requires executing sequences of up to 5 consecutive tasks. Unlike single-task benchmarks, CALVIN

Table 3: Ablation on attention direction. Intra-pair denotes attention between image and text tokens within each temporal unit. Inter-pair denotes attention across temporal units and is always causal. Results are average success rates (%) on the LIBERO benchmark.

T (Streaming Frames)	Intra-Pair	Inter-Pair	Spatial	Object	Goal	Long	Avg. Success Rate
1	Causal	—	97.4	97.8	96.8	91.0	95.8
	Bidirectional	—	98.4	98.2	97.4	92.0	96.5
3	Causal	Causal	98.6	96.0	95.4	91.0	95.3
	Bidirectional	Causal	98.8	98.6	98.6	93.8	97.5
5	Causal	Causal	98.0	97.6	96.8	90.6	95.5
	Bidirectional	Causal	**98.8**	**99.8**	**99.6**	**95.0**	**98.3**

Table 4: Effect of random-interval streaming training. Models are trained with fixed (δ=1) or random interval ($\delta \sim \mathcal{U}[3, 7]$) and evaluated at $\delta \in \{1, 3, 5\}$.

Training δ	Training T	Test T	Spatial	Object	Goal	Long	Avg. Success Rate
$\pi_{0.5}$ (δ=1)	1	1	98.4	98.2	97.4	92.0	96.5
Fixed (δ=1)	3	3	98.0	98.4	96.8	92.2	96.4
	5	5	98.4	97.4	98.6	93.4	97.0
Random δ	3	3	98.8	98.6	98.6	93.8	97.5
	5	5	**98.8**	**99.8**	**99.6**	**95.0**	**98.3**

explicitly tests the ability to compose actions across extended temporal horizons and maintain task progress over lengthy interactions.

As shown in Table 2, StreamPI ($T = 5$) achieves an average sequence length of 4.547, substantially outperforming both the single-frame $\pi_{0.5}$ baseline (4.313) and MemoryVLA [34] (4.090). While all methods perform comparably on the first task, StreamPI maintains significantly higher success rates at later stages (85.0% vs. 79.5% for $\pi_{0.5}$ and 69.4% for MemoryVLA on the 5th task). Notably, MemoryVLA, despite being designed for temporal memory, exhibits severe performance degradation in later sequence positions, suggesting that its memory mechanism struggles with error accumulation over long horizons. In contrast, StreamPI demonstrates robust and consistent gains at every sequence position, indicating the effectiveness of multi-frame temporal modeling.

4.5 Ablation Studies

Unless otherwise specified, all ablation studies are conducted on the LIBERO benchmark to validate the key design choices of StreamPI.

Effectiveness of Instruction-Anchored Temporal Modeling. We ablate the instruction-anchored temporal modeling at two levels: intra-pair (image-text fusion within each temporal unit) and inter-pair (aggregation across temporal units). As shown in Table 3, both dimensions contribute meaningfully to the final performance. For intra-pair attention, replacing bidirectional with causal attention leads to a consistent performance drop across all four LIBERO suites at every T, with the gap widening as T increases ($-$**5.6**% on LIBERO-Long at T=5). Causal intra-pair attention may prevent visual tokens from attending to the instruction, breaking the semantic coupling that StreamPI relies upon and causing instruction forgetting over long horizons. For inter-pair attention, its benefit is evident when comparing T=1 (no inter-pair attention) against T=5 under bidirectional intra-pair attention, yielding a **+1.8**% average gain and **+3.0**% on LIBERO-Long. This illustrates causal cross-temporal aggregation effectively accumulates task-relevant context as more streaming frames are processed. These results verify that both bidirectional intra-pair fusion and causal inter-pair aggregation are essential components of StreamPI's instruction-anchored temporal modeling.

Effect of Random-Interval Streaming Training. We study the effect of random-interval streaming training in the LIBERO simulation benchmark by comparing models trained with a fixed interval δ=1 and those trained with random temporal intervals. As shown in Table 4, random-interval training consistently improves performance over fixed-interval training under the same temporal length. For T=3, the average success rate increases from 96.4 to 97.5, while for T=5, it further improves from 97.0 to 98.3. The gains are especially clear on long-horizon tasks, suggesting that exposure to diverse

Table 5: Cross-stream generalization. A model trained with T=5 is evaluated at $T \in \{1, 3, 5\}$. Results are average success rates (%) on LIBERO-Long.

Training T	Test T	Spatial	Object	Goal	Long	Avg. Success Rate
$\pi_{0.5}$ (baseline)	1	98.4	98.2	97.4	92.0	96.5
T=3	1	98.0	98.8	97.8	93.4	97.0
	3	98.8	98.6	98.6	93.8	97.5
T=5	1	97.4	98.8	98.4	93.6	97.1
	3	**99.0**	98.8	98.0	93.8	97.4
	5	**98.8**	**99.8**	**99.6**	**95.0**	**98.3**

Table 6: The inference time of different streaming frames in streaming inference.

Streaming Frames	Inference Time (ms)
1	94.4 ± 3.4
3	97.9 ± 5.1
5	103.6 ± 6.3
8	110.9 ± 10.2
10	117.9 ± 16.5

temporal spacings helps the policy better capture temporal dependencies. Moreover, the interval δ also controls the frequency of streaming inference: δ=1 requires the policy to process every incoming frame, whereas a larger interval, e.g., δ=5, invokes inference only once every five frames. Therefore, maintaining strong performance under random and large intervals is important for reducing online inference overhead and leaving more time for action execution in streaming robot control.

Cross-Stream Generalization. We investigate whether a model trained with T=5 historical frames generalizes to inference with fewer frames (T=3 and T=1). As reported in Table 5, the trained model with T=5 retains strong performance when evaluated at T=3, with only a marginal degradation, suggesting that the model has internalized temporal patterns that remain partially effective even under reduced context. Performance at T=1 degrades more noticeably, yet still surpasses the $\pi_{0.5}$ single-frame baseline, indicating that temporal structure learned during training provides a residual benefit even when no historical frames are available at inference time.

4.6 Streaming Inference

The pseudo-code for the streaming inference of StreamPI is provided in **Algorithm 1**. To evaluate inference efficiency, we conduct 20 trials on a real-robot platform equipped with a single NVIDIA GeForce RTX 4090 GPU and report the mean latency with standard deviation in Table 6. The single-frame baseline incurs a latency of 94.4 ± 3.4 ms. Extending the temporal context to 3 frames adds only 3.5 ms of overhead (97.9 ± 5.1 ms total), and scaling further to 5, 8, and 10 frames yields 103.6 ± 6.3 ms, 110.9 ± 10.2 ms, and 117.9 ± 16.5 ms, respectively. The 5-frame setting only brings an increase of merely 9.2 ms over the baseline. These results confirm that StreamPI achieves efficient streaming inference, maintaining high responsiveness even as the temporal context grows.

5 Conclusion

In this work, we present StreamPI, a streaming multi-modal temporal modeling framework that equips VLA models with robust temporal awareness for robot manipulation. By treating *(image, text)* pairs as atomic temporal units and combining intra-pair bidirectional attention with inter-pair causal attention, StreamPI effectively captures cross-frame geometric context and maintains persistent instruction grounding over long horizons, without introducing additional parameters. To reduce the mismatch between fixed-interval training and asynchronous deployment, we further introduce Random-Interval Streaming Training, which exposes the model to diverse temporal spacings during training. Extensive experiments on real-robot manipulation tasks and the LIBERO benchmark show that StreamPI consistently outperforms the single-frame baseline, with notable gains on perception-sensitive and memory-dependent tasks. We hope StreamPI provides a simple and effective recipe for endowing future VLA models with persistent temporal reasoning.

Acknowledgments and Disclosure of Funding

This work is supported by the National Natural Science Foundation of China (No. 62422606, 62441615), Hong Kong Research Grant Council General Research Fund (No. 17213925), and Shenzhen Loop Area Institute Fund (No. FPF10120250006). There is no competing interest.

References

[1] Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.

[2] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: an open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *RSS*, 2025.

[4] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to act anywhere with task-centric latent actions. 2025.

[5] Jun Cen, Siteng Huang, Yuqian Yuan, Kehan Li, Hangjie Yuan, Chaohui Yu, Yuming Jiang, Jiayan Guo, Xin Li, Hao Luo, et al. Rynnvla-002: A unified vision-language-action and world model. *arXiv preprint arXiv:2511.17502*, 2025.

[6] Jiahong Chen, Jing Wang, Long Chen, Chuwei Cai, and Jinghui Lu. Nanovla: Routing decoupled vision-language understanding for nano-sized generalist robotic policies. *arXiv preprint arXiv:2510.25122*, 2025.

[7] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023.

[8] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[9] Yufei Duan, Hang Yin, and Danica Kragic. Real-time iteration scheme for diffusion policy. In *IROS*, 2025.

[10] Pascale Fung, Yoram Bachrach, Asli Celikyilmaz, Kamalika Chaudhuri, Delong Chen, Willy Chung, Emmanuel Dupoux, Hongyu Gong, Hervé Jégou, Alessandro Lazaric, et al. Embodied ai agents: Modeling the world. *arXiv preprint arXiv:2506.22355*, 2025.

[11] Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

[12] Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *RSS*, 2024.

[13] Ziyang Gong, Zehang Luo, Anke Tang, Zhe Liu, Shi Fu, Zhi Hou, Ganlin Yang, Weiyun Wang, Xiaofeng Wang, Jianbo Liu, et al. Ace-brain-0: Spatial intelligence as a shared scaffold for universal embodiments. *arXiv preprint arXiv:2603.03198*, 2026.

[14] Jinghua Hou, Tong Wang, Xiaoqing Ye, Zhe Liu, Shi Gong, Xiao Tan, Errui Ding, Jingdong Wang, and Xiang Bai. Open: Object-wise position embedding for multi-view 3d object detection. In *ECCV*, 2024.

[15] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint arXiv:2203.17054*, 2022.

[16] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[17] 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.

[18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. In *CoRL*, 2024.

[19] Hang Li, Fengyi Shen, Dong Chen, Liudi Yang, Xudong Wang, Jinkui Shi, Zhenshan Bing, Ziyuan Liu, and Alois Knoll. Remem-vla: Empowering vision-language-action model with memory via dual-level recurrent queries. *arXiv preprint arXiv:2603.12942*, 2026.

[20] Hao Li, Shuai Yang, Yilun Chen, Xinyi Chen, Xiaoda Yang, Yang Tian, Hanqing Wang, Tai Wang, Dahua Lin, Feng Zhao, et al. Towards efficient and robust manipulation via multi-frame vision-language-action modeling. In *AAAI*, 2026.

[21] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[22] Xinqing Li, Xin He, Le Zhang, Min Wu, Xiaoli Li, and Yun Liu. A comprehensive survey on world models for embodied ai. *arXiv preprint arXiv:2510.16732*, 2025.

[23] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. 2024.

[24] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS*, pages 44776–44791, 2023.

[25] Zhe Liu, Runhui Huang, Rui Yang, Siming Yan, Zining Wang, Lu Hou, Di Lin, Xiang Bai, and Hengshuang Zhao. Drivepi: Spatial-aware 4d mllm for unified autonomous driving understanding, perception, prediction and planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3688–3698, 2026.

[26] Zhenyang Liu, Yongchong Gu, Sixiao Zheng, Yanwei Fu, Xiangyang Xue, and Yu-Gang Jiang. Trivla: A triple-system-based unified vision-language-action model with episodic world modeling for general robot control. *arXiv preprint arXiv:2507.01424*, 2025.

[27] Yuxiang Lu, Zhe Liu, Xianzhe Fan, Zhenya Yang, Jinghua Hou, Junyi Li, Kaixin Ding, and Hengshuang Zhao. Faster: Rethinking real-time flow vlas. *arXiv preprint arXiv:2603.19199*, 2026.

[28] Chuanhao Ma, Hanyu Zhou, Shihan Peng, Yan Li, Tao Gu, and Luxin Yan. St-π: Structured spatiotemporal vla for robotic manipulation. *arXiv preprint arXiv:2604.17880*, 2026.

[29] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *RAL*, 2022.

[30] Chaojun Ni, Cheng Chen, Xiaofeng Wang, Zheng Zhu, Wenzhao Zheng, Boyuan Wang, Tianrun Chen, Guosheng Zhao, Haoyun Li, Zhehao Dong, et al. Swiftvla: Unlocking spatiotemporal dynamics for lightweight vla models at minimal overhead. *arXiv preprint arXiv:2512.00903*, 2025.

[31] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[32] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

[33] Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. Flower: Democratizing generalist robot policies with efficient vision-language-action flow policies. In *CoRL*, 2025.

[34] 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.

[35] Yiran Shi, Dongqi Guo, Tianchen Zhao, Feng Gao, Liangzhi Shi, Chao Yu, ZhiJian Mo, Qihua Xiao, XiaoShuai Peng, Qingmin Liao, et al. Streamingvla: Streaming vision-language-action model with action flow matching and adaptive early observation. *arXiv preprint arXiv:2603.28565*, 2026.

[36] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[37] Wenxuan Song, Jiayi Chen, Shuai Chen, Jingbo Wang, Pengxiang Ding, Han Zhao, Yikai Qin, Xinhu Zheng, Donglin Wang, Yan Wang, et al. Fast-dvla: Accelerating discrete diffusion vla to real-time performance. *arXiv preprint arXiv:2603.25661*, 2026.

[38] Brain Team, Ziyang Gong, Haoming Gu, Zehang Luo, Tianyi Zhang, Tao Tao, Yixiao Chi, Zhe Liu, Lingsi Zhu, Jingyuan Liu, et al. Ace-brain-0.5: A unified embodied foundational model for physical agentic ai. *arXiv preprint arXiv:2607.04426*, 2026.

[39] Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, et al. Mem: Multi-scale embodied memory for vision language action models. *arXiv preprint arXiv:2603.03596*, 2026.

[40] Haoxuan Wang, Gengyu Zhang, Yan Yan, Yuzhang Shang, Ramana Rao Kompella, and Gaowen Liu. Real-time robot execution with masked action chunking. *arXiv preprint arXiv:2601.20130*, 2026.

[41] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, pages 5294–5306, 2025.

[42] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. Exploring object-centric temporal modeling for efficient multi-view 3d object detection. In *CVPR*, pages 3621–3631, 2023.

[43] Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. In *AAAI*, 2025.

[44] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. TinyVLA: Towards fast, data-efficient vision-language-action models for robotic manipulation. *RAL*, 2025.

[45] Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

[46] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[47] Haozhe Xie, Beichen Wen, Jiarui Zheng, Zhaoxi Chen, Fangzhou Hong, Haiwen Diao, and Ziwei Liu. Dynamicvla: A vision-language-action model for dynamic object manipulation. *arXiv preprint arXiv:2601.22153*, 2026.

[48] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

[49] Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, et al. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. *arXiv preprint arXiv:2506.22242*, 2025.

[50] Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025.

[51] 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 *CVPR*, 2025.

[52] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023.

[53] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *ICLR*, 2026.

[54] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *ICLR*, 2024.

[55] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025.

[56] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *CoRL*, pages 2165–2183, 2023.

A Technical Appendices

The appendix provides supplementary details covering implementation specifics of streaming inference, additional real-world experimental results, qualitative visualizations, limitations, future directions, and broader impacts. A video demonstration is included in the supplemental materials.

B Implementation Details

Algorithm 1 Streaming Inference of StreamPI with KV-Cache

Input: observation stream $\{\mathbf{V}_{t_0}, \mathbf{V}_{t_1}, \ldots, \mathbf{V}_{t_N}\}$, language instruction l, maximum cache size T

Output: predicted action chunks $\mathcal{A}$
Initialize KV-Cache $\mathcal{C} \leftarrow \emptyset$, cache length $L_c \leftarrow 0$, step $n \leftarrow 0$
while $n \leq N$ **do**
  Receive current observation $\mathbf{V}_{t_n}$
  Encode atomic unit $(\mathbf{V}_{t_n}, l)$ with cached $\mathcal{C}$ to obtain fused representation $\mathbf{h}_{t_n}$
  if $L_c + 1 > T$ **then**
    Flush cache: $\mathcal{C} \leftarrow \emptyset$, $L_c \leftarrow 0$
  end if
  Update cache $\mathcal{C}$ and $L_c \leftarrow L_c + 1$
  Predict action $\mathcal{A}_n \sim \pi_\theta(\cdot \mid \mathbf{h}_{t_n}, \mathcal{C})$
  Dispatch $\mathcal{A}_n$ to client
  $n \leftarrow n + 1$
end while

B.1 Real-world Experiments

Hardware Setup. We use the AgileX PiperX 6-DoF robotic arms shown in Figure 4. The system follows the Aloha-style design [52], with leader arms for human teleoperation and follower arms for data collection and rollout. We use three cameras: one front-view camera (RealSense D455) and two wrist-mounted cameras (RealSense D435) on the follower arms.



Figure 4: AgileX PiperX robotic arms.

Tasks. We evaluate four real-world tasks to demonstrate the superiority of StreamPI. Specifically, we design two spatial-precision tasks (i.e., "Cup Insertion into Cup Sleeve", "Pen Insertion into Narrow Bottle") and memory-dependent tasks (i.e., "Rolling Object Grasping", "Shell Game"). For each task, we collect 100 demonstration episodes using human teleportation at 30 FPS.

The language instructions for the tasks are as follows:

- **Shell Game**: "Pick up the cup that contains the hidden object after the shuffles."
- **Rolling Object Grasping**: "Pick up the rolling bottle."
- **Pen Insertion into Narrow Bottle**: "Insert the pen from one bottle into another bottle."

Table 7: Performance comparison of each trial on "Shell Game" task. S.R. denotes success rate.

Model	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	S.R. (%)
$\pi_{0.5}$	1	0	1	0	0	1	1	1	1	0	1	0	0	0	0	46.7%
StreamPI	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	80.0%

Table 8: Performance comparison of each trial on "Pick up the Rolling Bottle" task. S.R. denotes success rate.

Model	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	S.R. (%)
$\pi_{0.5}$	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	1	0	26.7
StreamPI	0	1	1	0	1	0	1	0	0	1	0	1	1	0	1	0	1	1	1	1	1	0	1	1	1	1	0	0	1	1	63.3

Table 9: Performance comparison of each trial on "Pen Insertion into Narrow Bottle" task. S.R. denotes success rate.

	M-F										M-M										F-M										S.R. (%)
Model	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	
$\pi_{0.5}$	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	0	1	0	1	0	1	0	1	0	0	1	1	1	0	40.0
StreamPI	1	1	0	1	0	0	1	0	0	1	1	1	1	0	1	0	0	1	1	1	1	1	0	1	1	1	1	0	1	0	66.7

Table 10: Performance comparison of each trial on "Cup Insertion into Cup Sleeve" task. S.R. denotes success rate.

Model	Center					Left-Far					Right-Far					Right-Near					Left-Near					S.R. (%)
$\pi_{0.5}$	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	60.0
StreamPI	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	92.0

Table 11: Each pattern of the Shell Game. 'L', 'M', and 'R' is the left, middle, and right position, respectively. A→ B denotes that move the cup that contains the target object from A to B.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L→M	L→R	L→M→R	L→M→L	L→R→M	M→L	M→L→M	M→R	M→R→M	M→L→R	R→M	R→M→R	R→L	R→L→M	R→L→R

- **Cup Insertion into Cup Sleeve**: "Pick up the paper cup and put it into the cup sleeve."

Evaluation. To validate the real-world performance of StreamPI, we design detailed comparative experiments for each task. The score is defined as 1 point for success and 0 points for failure. We adopt success rate as the evaluation metric.

For "Shell Game", we design 15 different patterns. Each pattern is shown in Table 11. We provide detailed performance comparison of each trial in Table 7. StreamPI achieves 33.3% improvement compared to $\pi_{0.5}$.

For "Pick up the Rolling Bottle", we evaluate performance with 30 trails. We provide detailed performance comparison of each trial in Table 8. This task is a highly dynamic task and needs to have the ability to use temporal information to predict the motion of objects. StreamPI outperforms $\pi_{0.5}$ by 36.6%, demonstrating the effectiveness of proposed streaming temporal modeling.

For "Pen Insertion into Narrow Bottle", we set 3 different patterns. "M-F": move the pen from the middle cup to the top right cup. "M-M": move the pen from the left cup to the right cup. "F-M": move the pen from the left right cup to the middle cup. We evaluate performance with 10 trails on each pattern. We provide detailed performance comparison of each trial in Table 9. This task not only requires high precise perception to insert the pen into a narrow bottle but also needs temporal information to determine which bottle to put the pen in. Benefited from introduced temporal modeling, StreamPI is better able to handle this task than $\pi_{0.5}$.

For "Cup Insertion into Cup Sleeve", we set 5 different positions of the cup sleeve: center, left-far, right-far, left-near, and right-near and evaluate performance with 5 trails on each position. We provide detailed performance comparison of each trial in Table 10. StreamPI achieves stronger perception capability compared to $\pi_{0.5}$, especially for distant locations of the cup sleeve.

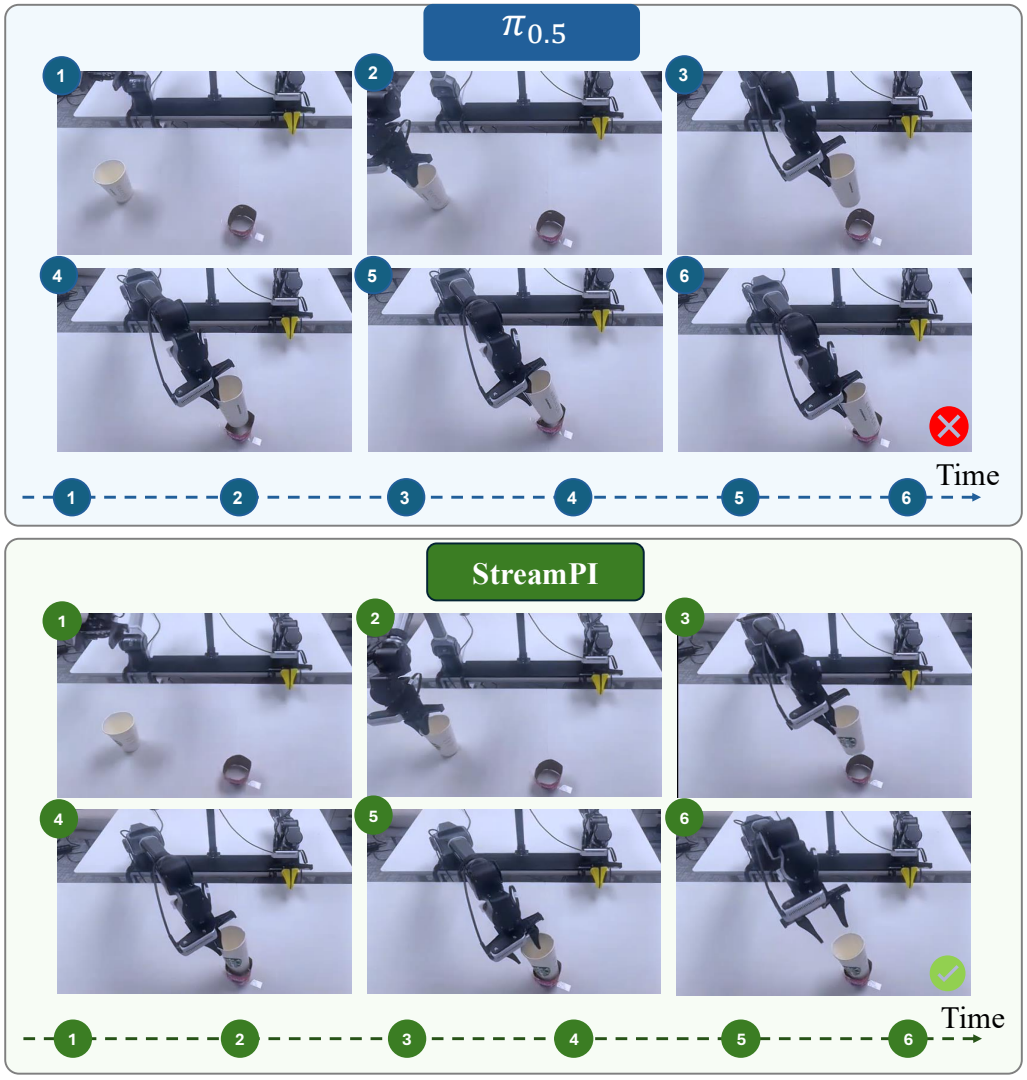

Figure 5: Comparison of "Cup Insertion into Cup Sleeve".

C Visualization Comparisons

We provide visualization comparison of $\pi_{0.5}$ and StreamPI on "Cup Insertion into Cup Sleeve", "Pen Insertion into Narrow Bottle", and "Rolling Object Grasping" tasks.

Cup Insertion into Cup Sleeve. As shown in Figure 5, $\pi_{0.5}$ struggles to accurately estimate the spatial position of distant objects, making it difficult to place the cup precisely in the cup sleeve. In contrast, StreamPI accurately locate objects and place the cup successfully.

Pen Insertion into Narrow Bottle. As shown in Figure 6, "Pen Insertion into Narrow Bottle" needs both precise perception and temporal information. Therefore, $\pi_{0.5}$ cannot determine whether the current state is the beginning or the end of a task because it lacks temporal information, causing it to mistake picking up the pen for the moment it puts the pen into the bottle. StreamPI successfully pick up the pen from one bottle and put it into another bottle, demonstrating the effectiveness of StreamPI in temporal modeling.

Pick up the rolling bottle. For "Pick up the rolling bottle", the slow reaction of $\pi_{0.5}$ results in missing the rolling bottle because it is unable to use temporal information to determine the bottle's

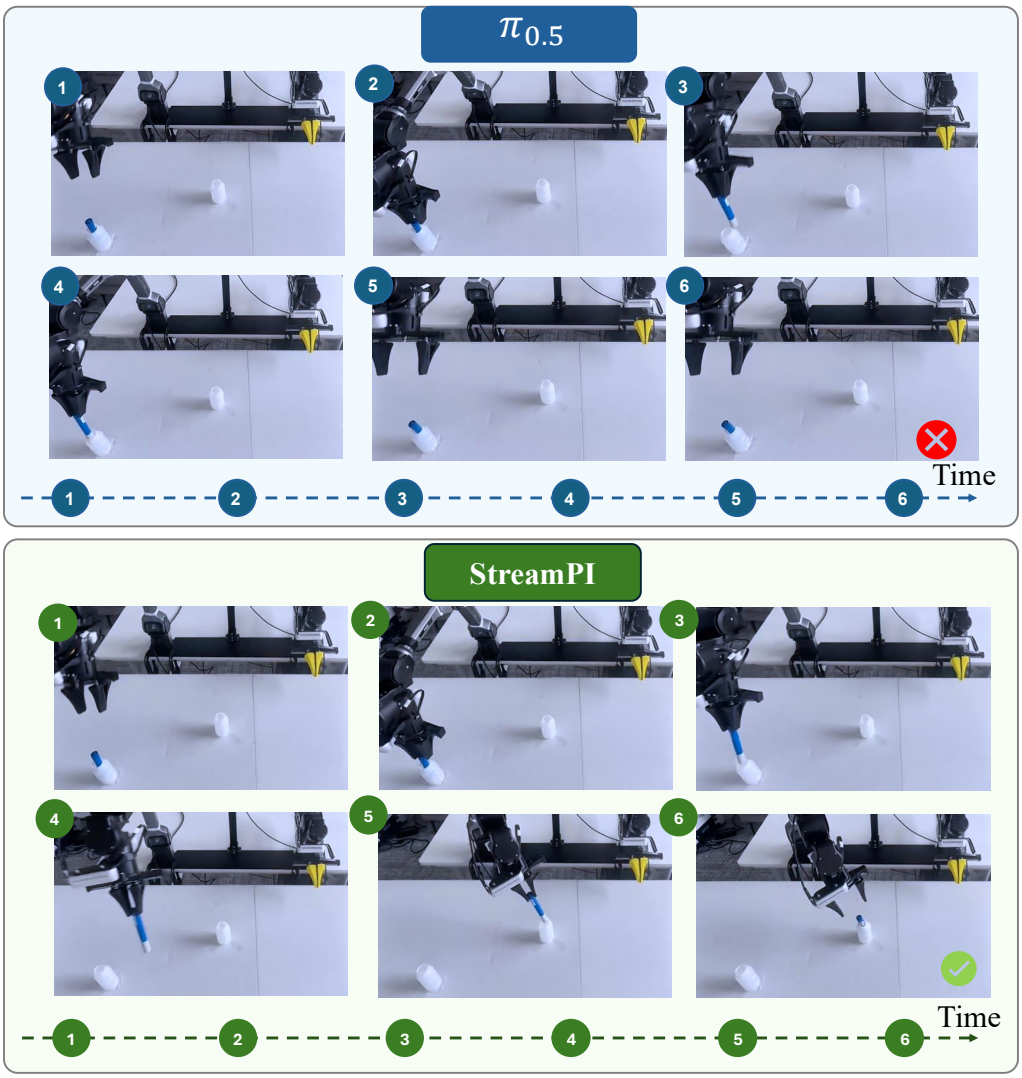

Figure 6: Comparison of "Pen Insertion into Narrow Bottle".

motion properties. As shown in Figure 7, Compared to $\pi_{0.5}$, StreamPI can generate the corresponding action in advance by predicting the movement of objects, thus successfully catching the rolling bottle.

Shell Game. To fully illustrate the effectiveness of StreamPI, we provide a qualitative visualization of the "Shell Game" performed four times consecutively in Figure 8. For convenience, we use a red circle to indicate which cup contains the object. Despite exchanging cups multiple times, StreamPI can still accurately guess which cup contains the target object.

D Limitations and Future Work

Limitations. Despite the effectiveness of StreamPI in efficient temporal modeling for VLAs, it has several limitations. First, since training requires loading all frames, the training cost of StreamPI in handling extremely long temporal horizons will be unacceptable. Second, the random-interval streaming training, while improving robustness to variable frame rates, does not fully address extreme asynchrony in real-robot deployment.

Future Work. In the future, we will design a more efficient training framework to support longer horizons (>100 frames) with lower computational cost. Then, we plan to introduce adaptive KV

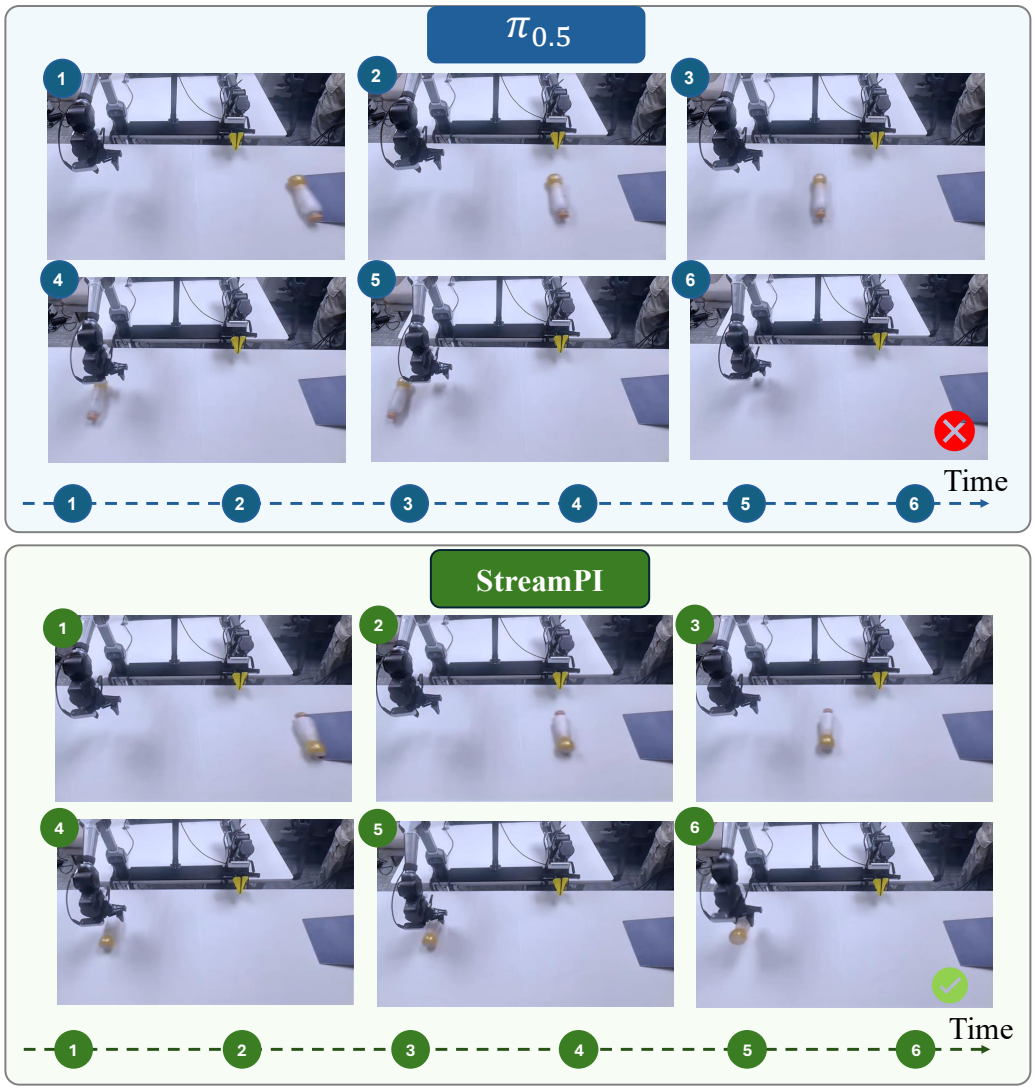

Figure 7: Comparison of "Pick up the rolling bottle".

cache pruning to maintain representation quality for ultra-long temporal horizons with negligible additional inference time.

E Broader Impacts

StreamPI advances robot manipulation by enabling efficient, robust temporal modeling without additional parameters, lowering the barrier for deploying strong foundation models in real-robot systems. Its streaming design and robustness to asynchronous observations make it suitable for real-world embodied tasks, improving efficiency and safety in human-robot interaction. However, the widespread deployment of more capable manipulation robots with our temporal modeling may affect labor markets in routine industrial tasks. Additionally, ensuring the robustness of VLAs with temporal modeling on the real-robot to out-of-distribution scenarios is critical to avoid safety risks in real-world operation, which we will address in future work.

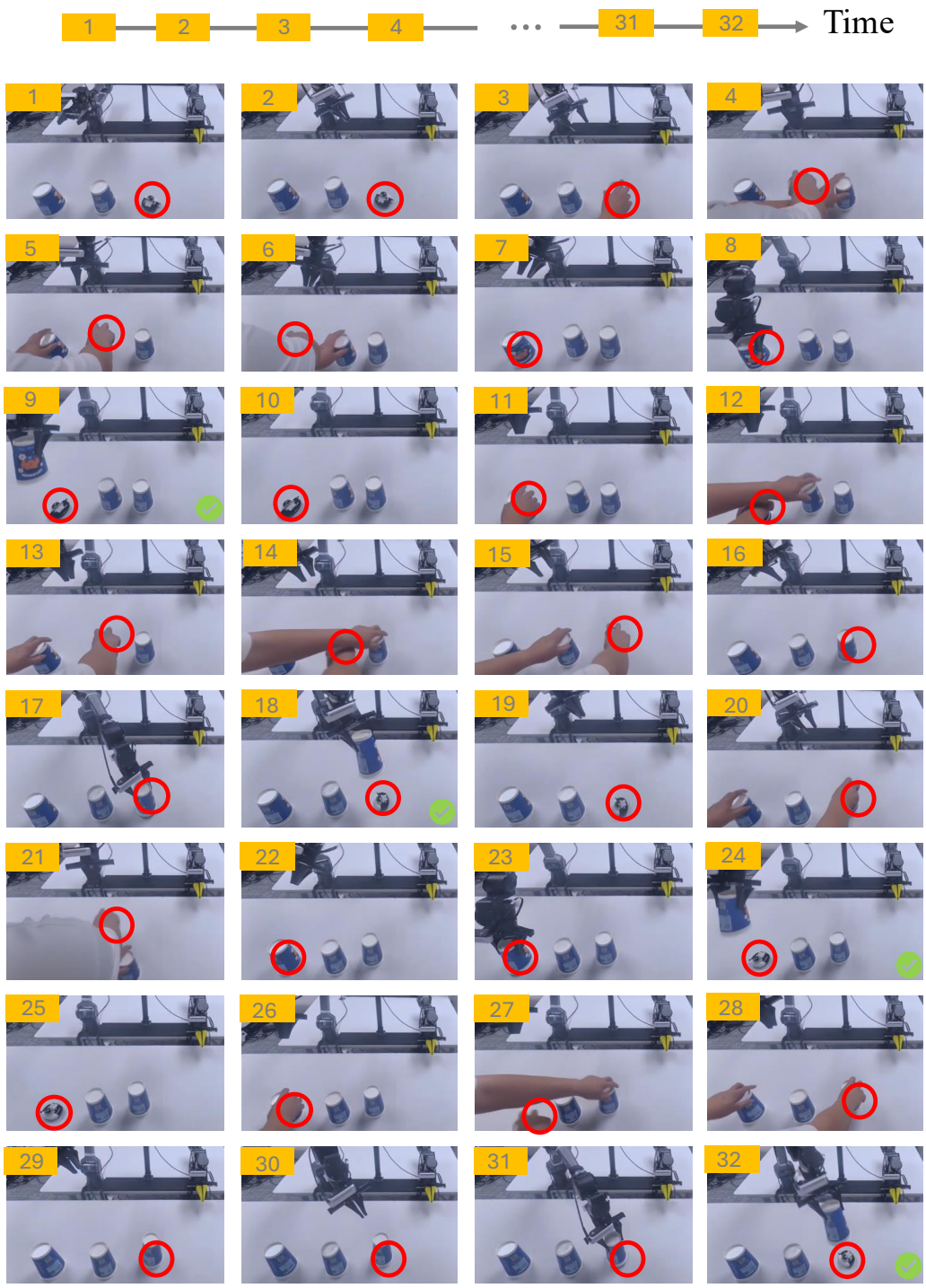

Figure 8: The continuous qualitative results of “Shell Game”. For convenience, we use a red circle to indicate which cup contains the object.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are fully consistent with the empirical contributions presented in the main paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We add the limitations in our technical appendices.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This paper does not include theoretic result.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Our paper can reproduce the main experimental results.

   Guidelines:

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

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

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

Answer: [N/A]

Justification: We will release data and code after the paper is accepted.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes],

   Justification: This paper include the training and test details.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [N/A]

   Justification: We do not contain the statistical significance of the experiments due to high computation cost.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide sufficient information on the computer resources.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We follow the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We add the broader impacts in our technical appendices.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The proposed framework, StreamPI, is built upon an existing VLA backbone (π0.5) and trained on task-specific robot demonstration data collected in controlled laboratory settings. Such data and models present minimal risk of misuse, and no additional safeguards beyond standard research practice are required.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have cited all related works of exisiting assets.

    Guidelines:

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

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [No]

    Justification: We do not involve new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: This research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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