TEMPORAL FORCING: 4D REPRESENTATION ALIGNMENT FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon robotic manipulation requires vision-language-action (VLA) models to track scene states and their evolution beyond the current observation. However, simply conditioning policies on observation history does not guarantee that the history is effectively utilized: action supervision constrains what the policy should do, but only indirectly constrains what its history representations should retain. To resolve this, we present Temporal Forcing, a 4D representation alignment framework that explicitly supervises latent temporal states and their transitions. Specifically, we first introduce a history pathway that compresses past observations into compact latent tokens. We then align these tokens and current-frame features with geometric targets from a pretrained 4D foundation model, providing direct supervision at both the state and transition levels. The 4D foundation model and alignment heads are used only for training-time supervision. Temporal Forcing improves average success from 96.6% to 98.8% on LIBERO, with the largest gain on LIBERO-Long (93.8% to 97.2%), and from 53.5% to 62.8% across twelve RoboTwin 2.0 tasks. Furthermore, Temporal Forcing increases full-task success from 20.0% to 43.3% on a physical multi-stage hidden-placement task. Controlled experiments show that 4D representation alignment is crucial for making observation history beneficial to the model. Code will be publicly available.

1 INTRODUCTION

Vision-language-action (VLA) models map language instructions and visual observations to robot actions (Zitkovich et al., 2023; Kim et al., 2024; Team et al., 2024; Black et al., 2024). Long-horizon manipulation, however, requires a history representation that tracks both physical scene evolution and task progress. For example, as shown in Fig. 1(a), once one block is placed into an opaque box and becomes invisible, subsequent execution proceeds without full visibility of the earlier interaction. Later, deciding whether to close the drawer requires information about whether both placement stages have been completed, which may no longer be recoverable from the current observation alone. Such sequential decisions can benefit from representations that capture not only the current scene state, but also how it has evolved through recent interactions.

Recent VLA methods incorporate observation history through temporal modules or memory, enabling policies to use past interactions to inform current actions (Koo et al., 2026; Shi et al., 2026b;a). However, providing past observations does not directly determine what their compressed representations should retain. Specifically, action supervision constrains these representations only indirectly through action prediction, without explicitly guiding how they should encode scene states or their changes. This suggests that the bottleneck is not access to history, but supervision of latent temporal states.

To resolve this, we present Temporal Forcing, a 4D representation alignment framework that explicitly supervises latent temporal states and their transitions (Fig. 1(c)). We first introduce a history pathway that compresses a bounded window of past observations into compact history tokens. To supervise this pathway, we use a pretrained 4D foundation model (Zhuo et al., 2025) to extract history-conditioned geometric targets from a longer causal context. We align per-timestep history features with these targets and introduce explicit latent-transition supervision, directly matching inter-timestep changes in the learned history representations to the corresponding changes in causal 4D geometric features. We further extend this geometric supervision to the VLA backbone through

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

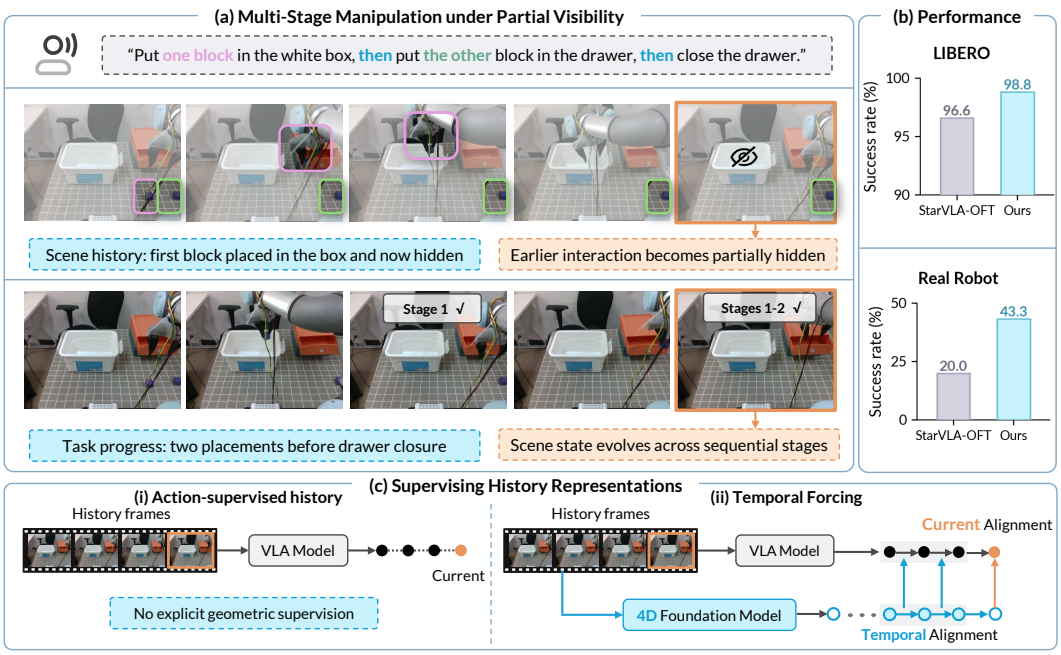

Figure 1: **Motivation and results of Temporal Forcing.** (a) A physical multi-stage manipulation task with object occlusion and sequential execution. (b) Average success on LIBERO and full-task success on the real-robot task. (c) Temporal Forcing explicitly supervises latent temporal states and their transitions with causal 4D geometric targets, with complementary current-frame alignment.

dense current-frame alignment. The 4D foundation model and alignment heads are used only during training.

Temporal Forcing consistently improves its base model across LIBERO, RoboTwin 2.0, and the real-robot task, with the largest gains in long-horizon or occlusion-heavy settings. Controlled experiments further show that these gains depend on 4D-supervised history rather than history input alone. Our main contributions are as follows:

- We identify a supervision bottleneck in history-aware VLAs: action supervision alone does not ensure effective use of observation history, motivating direct supervision of what history representations retain.
- We introduce Temporal Forcing, which aligns latent temporal states and their transitions with causal 4D geometric representations, together with a compact History Pathway whose 4D supervision is used only during training.
- We validate Temporal Forcing across simulation and real robots, improving LIBERO, RoboTwin 2.0, and multi-stage physical manipulation; ablations and history interventions further confirm that the learned history directly contributes to action prediction.

2 Related Work

History-augmented VLAs. Before the emergence of VLAs, visuomotor policies already incorporated observation history through recurrent models or short observation windows (Mandlekar et al., 2021; Chi et al., 2025). Recent VLA methods incorporate longer history through temporal modules or memory. HAMLET (Koo et al., 2026) introduces contrastively initialized history tokens, while MemoryVLA (Shi et al., 2026b) maintains a perceptual-cognitive memory and MemoryVLA++ (Shi et al., 2026a) further combines memory with future imagination. Other methods encode 3D or 4D history on the input side (Zhang et al., 2025a; Xiao et al., 2026; Zhou et al., 2025). These methods provide temporal context to the policy. Temporal Forcing instead directly supervises the latent representations formed from observation history using history-conditioned 4D geometric features.

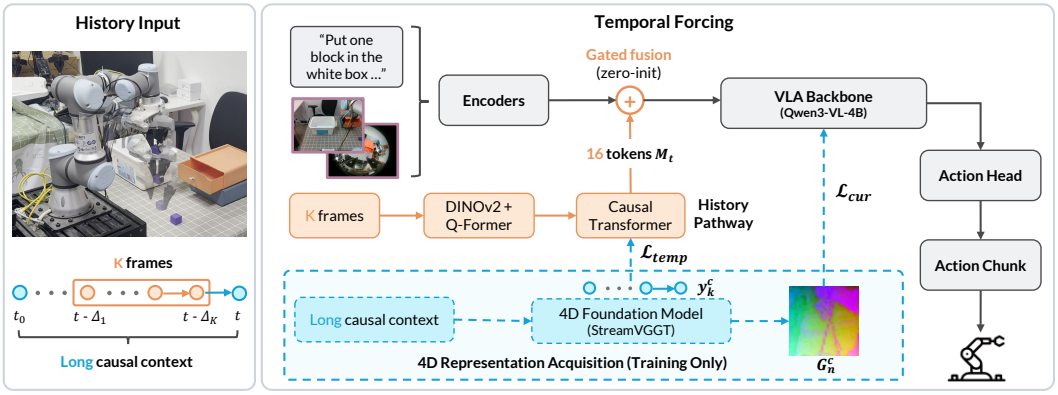

Figure 2: **Overview of Temporal Forcing.** The History Pathway summarizes K sampled past frames into history tokens that are fused into the VLA model. A streaming 4D foundation model processes a longer causal context and provides geometric targets for temporal alignment ($\mathcal{L}_{\text{temp}}$) and current-frame alignment ($\mathcal{L}_{\text{cur}}$). Blue dashed components are used only during training.

Training-time representation alignment for VLAs. Recent work has also explored supervising VLA representations with features from pretrained models. Aligning intermediate representations with features from pretrained encoders has improved generative models (Yu et al., 2024; Wu et al., 2025). Spatial Forcing (Li et al., 2026b) aligns image-token features with VGGT (Wang et al., 2025), GLaD (Guo et al., 2025) and ROCKET (Sun et al., 2026) extend geometric alignment to multiple layers, and QDepth-VLA (Li et al., 2025) predicts quantized depth tokens. Another line introduces explicit 3D inputs such as point clouds or rendered views (Li et al., 2026a; Sun et al., 2025; Singh et al., 2025), requiring additional geometric inputs or preprocessing at inference. These methods mainly model spatial structure from individual observations. Temporal Forcing applies representation alignment to latent observation history, using temporally conditioned targets to supervise both latent temporal states and their transitions.

Temporal supervision for VLAs. Temporal information can also be learned through future prediction or dynamics supervision. World-model-based VLA methods predict or generate future states (Li et al., 2026c; Zhang et al., 2025b), while VLAFlow (Xia et al., 2026) aligns future latent representations and LARA (Liu et al., 2026) aligns VLA representations with latent visual dynamics. StreamVGGT (Zhuo et al., 2025), in contrast, causally processes observed sequences and produces temporally consistent geometric representations of the evolving 3D scene. Temporal Forcing instead supervises transitions between latent temporal states, using causal 4D representations as geometric targets rather than predicting future latent states.

3 Method

3.1 Preliminaries

A vanilla VLA model builds on a pretrained vision-language model (VLM) and maps the current observation and a language instruction to robot actions. Specifically, at each timestep t, the model receives visual observations $\mathbf{o}_t = \{\mathbf{I}_t^c \mid c \in \mathcal{C}\}$ from a set of cameras $\mathcal{C}$ and a language instruction ℓ. The VLA backbone jointly encodes the visual observations and the language instruction. An MLP action head (Kim et al., 2025) then maps the backbone output to a chunk of H future actions $\hat{\mathbf{A}}_t \in \mathbb{R}^{H \times d_a}$, where d_a denotes the dimensionality of each action. Given the ground-truth action chunk $\mathbf{A}_t$, the action loss is defined as

$$\mathcal{L}_{\text{act}} = \frac{1}{Hd_a} \left\| \mathbf{A}_t - \hat{\mathbf{A}}_t \right\|_1 . \tag{1}$$

The vanilla model uses only the current visual observation and does not incorporate observation history. We therefore introduce a History Pathway to encode past observations and integrate the resulting representations into the VLA backbone.

3.2 History Pathway

We design the History Pathway to incorporate observation history with limited inference overhead while preserving the base model's behavior at initialization. It compresses past observations into a fixed number of tokens and injects them into the current-frame image tokens through gated cross-attention, without extending the backbone sequence.

History window and gist tokens. At each timestep t, we sample K past frames from each camera at uniformly spaced offsets $\Delta_1 > \cdots > \Delta_K > 0$, ordered from oldest to most recent. For each sampled frame, a frozen DINOv2 encoder (Oquab et al., 2023) extracts visual features, which are compressed by a trainable Q-Former (Li et al., 2023) into two gist tokens u_k^c for historical timestep k and camera c. Before temporal aggregation, we add time-offset and camera embeddings to each gist token to indicate its temporal position and camera identity.

Temporal summary and gated injection. A causal Transformer aggregates the gist tokens at each historical timestep with those from earlier timesteps to produce the History Latent Representation. To provide the VLA backbone with a compact representation of the history, we use 16 learnable queries to summarize these features into history tokens M_t. The current primary-camera image tokens x_t then attend to M_t through gated cross-attention before the first decoder layer. This fusion updates the image tokens as $x_t' = x_t + \tanh(\alpha)\ \mathrm{CrossAttn}(x_t, M_t)$, where α is a learnable scalar gate initialized to zero.

3.3 4D Representation Acquisition

We use StreamVGGT (Zhuo et al., 2025), a pretrained 4D foundation model, to extract geometric features from observation sequences (Fig. 2). Compared with the History Pathway, StreamVGGT processes a longer causal context ending at each anchor timestep t, reconstructing the observations in a shared coordinate system and using causal attention to condition each frame on earlier observations. This causal processing yields two complementary geometric targets: frame-level features that describe scene evolution and dense features that preserve the spatial structure of the current frame. At the K historical timesteps sampled by the History Pathway, we average the tokens of each frame and subtract a fixed per-camera mean μ_c to obtain the centered feature $\tilde{y}_k^c$, which is layer-normalized to yield the Causal Geometric Feature y_k^c. At the current timestep t, we retain the spatial features and layer-normalize each feature vector to obtain the Dense Geometric Features G_n^c, where n indexes spatial locations. Both types of features are precomputed offline at regularly spaced anchor timesteps and remain fixed during training.

3.4 4D Representation Alignment

The History Pathway gives the VLA model access to past observations, but the action loss provides only indirect supervision for the history representations. We therefore introduce 4D representation alignment to guide the model in capturing scene geometry and its changes over time.

Latent state supervision. To capture the scene state at each historical timestep, we first align the history representations with the corresponding Causal Geometric Features. For each historical timestep k and camera c, we average the two gist tokens output by the causal Transformer to obtain $\bar{u}_k^c$. To compare this feature with its geometric target y_k^c, we use a learnable projection head ψ to obtain $z_k^c = \psi(\bar{u}_k^c)$. We minimize their cosine distance using the state loss

$$\mathcal{L}_{\mathrm{state}} = \frac{1}{|\mathcal{C}|K} \sum_{c \in \mathcal{C}} \sum_{k=1}^{K} \left[1 - \cos(z_k^c, y_k^c)\right], \tag{2}$$

where $\cos(\cdot, \cdot)$ denotes cosine similarity.

Explicit latent-transition supervision. Per-timestep alignment constrains what each history representation encodes, but does not explicitly constrain how the latent state evolves between observations. We therefore directly supervise latent transitions by matching representation changes between consecutive historical timesteps. For each camera, we compute $\delta z_k^c = z_{k+1}^c - z_k^c$ and

$\delta y_k^c = \tilde{y}_{k+1}^c - \tilde{y}_k^c$, where the target difference is taken on the centered features without layer normalization, and align their directions using the change loss

$$\mathcal{L}_{\text{change}} = \frac{1}{|\mathcal{C}|(K-1)} \sum_{c \in \mathcal{C}} \sum_{k=1}^{K-1} \left[1 - \cos(\delta z_k^c, \delta y_k^c)\right]. \tag{3}$$

Unlike pointwise feature alignment, $\mathcal{L}_{\text{change}}$ directly constrains the direction of latent-state evolution across adjacent history states.

History readout supervision. The state and change losses supervise the per-timestep features, while the VLA backbone receives the 16 summarized history tokens. To supervise these tokens as well, we average them into $\bar{m}$ and align the projected feature $\psi(\bar{m})$ with the geometric targets at the most recent historical timestep using the readout loss

$$\mathcal{L}_{\text{read}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \left[1 - \cos(\psi(\bar{m}), y_K^c)\right]. \tag{4}$$

This loss provides a direct training signal to the learnable summary queries even when the fusion gate is closed.

We combine the three terms into the **temporal alignment** objective:

$$\mathcal{L}_{\text{temp}} = \frac{1}{6} \left(\mathcal{L}_{\text{state}} + 4\mathcal{L}_{\text{change}} + \mathcal{L}_{\text{read}}\right). \tag{5}$$

Current-frame alignment. As a complementary objective, we align the current-frame backbone features with dense geometric features from the same 4D foundation model. For each camera c, we extract the current-frame image features h_n^c from an intermediate backbone layer and project them with a learnable head ϕ. We match each projected feature to its corresponding geometric target using

$$\mathcal{L}_{\text{cur}} = \frac{1}{|\mathcal{C}|N} \sum_{c \in \mathcal{C}} \sum_{n=1}^{N} \left[1 - \cos(\phi(h_n^c), G_n^c)\right], \tag{6}$$

where N is the number of spatial locations per camera. Since StreamVGGT processes the preceding observations when extracting G_n^c, the current-frame features receive geometric supervision that incorporates observation history.

Training and inference. We train the model with the action loss and the two alignment objectives:

$$\mathcal{L} = \mathcal{L}_{\text{act}} + \lambda_{\text{cur}}\mathcal{L}_{\text{cur}} + \lambda_{\text{temp}}\mathcal{L}_{\text{temp}}, \tag{7}$$

where $\lambda_{\text{cur}} = \lambda_{\text{temp}} = 0.5$. The alignment losses are applied at anchor timesteps with precomputed geometric targets; other timesteps use the action loss alone.

At inference, we retain the base VLA model, the History Pathway, and gated fusion; StreamVGGT and the alignment heads are not used. We cache the per-frame gist tokens before adding time-offset embeddings and reuse them across control steps.

4 Experiments

We evaluate Temporal Forcing on LIBERO, RoboTwin 2.0, and a real-robot multi-stage manipulation task. Ablation studies and history-intervention analyses further demonstrate the role of 4D representation alignment in learning to use observation history.

4.1 Experimental Setup

Benchmarks. We evaluate on LIBERO (Liu et al., 2023) and RoboTwin 2.0 (Chen et al., 2025). On LIBERO, we train a single model jointly on the Spatial, Object, Goal, and Long suites and evaluate 50 trials per task, totaling 500 trials per suite. On RoboTwin 2.0, we evaluate twelve bimanual tasks, using 50 clean demonstrations per task for training and 100 trials per task under the easy (clean) setting. We report success rates and their unweighted averages over tasks.

Table 1: **Performance comparison on LIBERO (success rate, %).** The gray-text row evaluates the same Temporal Forcing checkpoint with the injection gate closed at inference. Best results in each column are in bold. † Average recomputed over the four reported suites.

Method	Spatial	Object	Goal	Long	Average
General VLA					
TraceVLA (Zheng et al., 2024)	84.6	85.2	75.1	54.1	74.8
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5
SpatialVLA (Qu et al., 2025)	88.2	89.9	78.6	55.5	78.1
CogACT (Li et al., 2024)	97.2	98.0	90.2	88.8	93.6
π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2
Framewise 3D alignment					
GLaD (Guo et al., 2025)	95.0	97.4	94.4	89.4	94.1
Spatial Forcing (Li et al., 2026b)	99.4	99.6	98.8	96.0	98.5
History-augmented VLA					
4D-VLA (Zhang et al., 2025a)	88.9	95.2	90.9	79.1	88.6
MemoryVLA† (Shi et al., 2026b)	98.4	98.4	96.4	93.4	96.7
HAMLET (Koo et al., 2026)	99.0	**100.0**	**99.2**	92.2	97.6
Same base model (Qwen3-VL-OFT)					
StarVLA-OFT (base; Community, 2026)	97.8	98.6	96.2	93.8	96.6
Temporal Forcing (Ours)	**99.6**	99.8	98.4	**97.2**	**98.8**
w/o 4D alignment (action loss only)	99.0	98.2	97.6	94.8	97.4
w/o history at inference (gate closed)	98.6	97.0	99.0	94.2	97.2

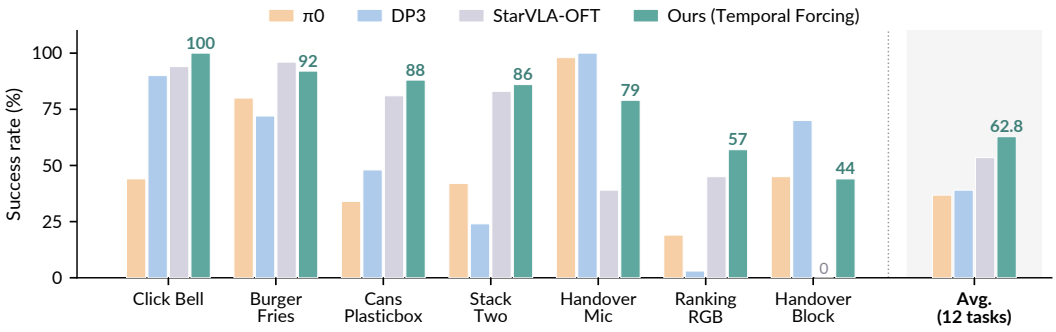

Figure 3: **Performance comparison on RoboTwin 2.0.** Results include seven representative tasks and the average success rate over all twelve evaluated tasks. Baselines are π_0 (Black et al., 2024), DP3 (Ze et al., 2024), and StarVLA-OFT (Community, 2026).

Implementation Details. We use the QwenOFT implementation of StarVLA (Community, 2026) as the base model, with a Qwen3-VL-4B backbone (Bai et al., 2025) and an MLP action head. For the main LIBERO experiments, we follow the official training recipe for 50k steps with a global batch size of 128 on eight NVIDIA A100 GPUs. The History Pathway uses $K=7$ past frames per camera, spanning 2.8 seconds on LIBERO and 3.7 seconds on RoboTwin 2.0. History encoding and geometric alignment use the primary and wrist cameras on LIBERO and the head camera on RoboTwin 2.0. The ablation training protocol is specified in Sec. 4.3.

4.2 Main Results

As shown in Table 1, Temporal Forcing achieves the highest average success rate of 98.8% among the compared methods, outperforming its base model by 2.2 percentage points. The improvement holds across all four suites and is most pronounced on Long, where Temporal Forcing surpasses the

Table 2: **Ablation studies on LIBERO (success rate, %).** (a) Model components and geometric targets. (b) Decomposition of temporal supervision. All variants are trained for 10k steps. For (b), the tuple in $\mathcal{L}_{\text{temp}}$ denotes $(w_{\text{state}}, w_{\text{change}}, w_{\text{read}})$ before the common factor of $1/6$. Best results in each column are in bold within each panel.

	Configuration				LIBERO suite				
ID	Target	History	$\mathcal{L}_{\text{cur}}$	$\mathcal{L}_{\text{temp}}$	Spatial	Object	Goal	Long	Average
(a) Model components and geometric targets									
1	✗	✗	✗	✗	**98.4**	97.2	91.0	76.2	90.7
2	3D	✗	✓	✗	96.6	98.0	91.6	64.6	87.7
3	4D	✗	✓	✗	97.8	99.2	**96.6**	68.2	90.5
4	✗	✓	✗	✗	97.8	93.2	88.8	72.8	88.2
5	4D	✓	✓	✗	94.6	98.2	79.0	63.0	83.7
6	4D	✓	✗	✓	97.2	97.4	91.4	77.2	90.8
7	3D	✓	✓	✓	97.2	98.6	81.0	63.2	85.0
8	4D	✓	✓	✓	**98.4**	**99.6**	92.6	**83.8**	**93.6**
(b) Temporal supervision: $\mathcal{L}_{\text{temp}}$ given as $(w_{\text{state}}, w_{\text{change}}, w_{\text{read}})$									
5	4D	✓	✓	$(0,0,0)$	94.6	98.2	79.0	63.0	83.7
9	4D	✓	✓	$(1,0,1)$	97.0	**99.6**	**93.4**	76.0	91.5
10	4D	✓	✓	$(0,4,0)$	97.0	**99.6**	91.4	**84.8**	93.2
11	4D	✓	✓	$(1,1,1)$	96.6	99.0	89.4	83.6	92.2
8	4D	✓	✓	$(1,4,1)$	**98.4**	**99.6**	92.6	83.8	**93.6**

base model by 3.4 points (97.2% vs. 93.8%). These consistent gains demonstrate the effectiveness of Temporal Forcing for robotic manipulation, particularly on long-horizon tasks.

On RoboTwin 2.0 (Fig. 3), Temporal Forcing achieves an average success rate of 62.8%, outperforming the base model by 9.3 percentage points and improving nine of the twelve tasks. The largest gains occur on Handover Block and Handover Mic, where success rates increase from 0.0% to 44.0% and from 39.0% to 79.0%, respectively. Both tasks involve object occlusions during transfer between the arms and show some of the largest gains from Temporal Forcing. Fig. 3 presents seven representative tasks and the average over all twelve tasks.

4.3 Ablations And History Analysis

We conduct controlled ablations on LIBERO to examine the history pathway, geometric targets, and temporal supervision, and to analyze what the learned history tokens encode and whether they contribute to action prediction. All variants in Table 2 are trained for 10k steps with a global batch size of 64 on four NVIDIA A800 GPUs, using the same backbone initialization, training data, optimizer, learning-rate schedule, and evaluation protocol.

Model Components. We first evaluate the contributions of the history pathway and alignment objectives in Table 2(a). Adding the history pathway alone (row 4) reduces the average success rate from 90.7% to 88.2%, while current-frame alignment with 4D targets alone (row 3) achieves 90.5%. Combining the history pathway with both alignment objectives (row 8) raises the average to 93.6%, with the largest gain on Long (83.8% vs. 76.2%). History inputs alone do not improve the baseline; the gain comes from pairing them with 4D representation supervision.

We then remove each alignment objective from the full model. Without temporal alignment $\mathcal{L}_{\text{temp}}$ (row 5), the average success rate drops from 93.6% to 83.7%; removing current-frame alignment $\mathcal{L}_{\text{cur}}$ (row 6) reduces it to 90.8%. Both removals cause the largest degradation on Long, from 83.8% to 63.0% and 77.2%, respectively. Current-frame alignment improves the history-augmented policy when paired with temporal supervision, but degrades it without that supervision (rows 4–6 and 8).

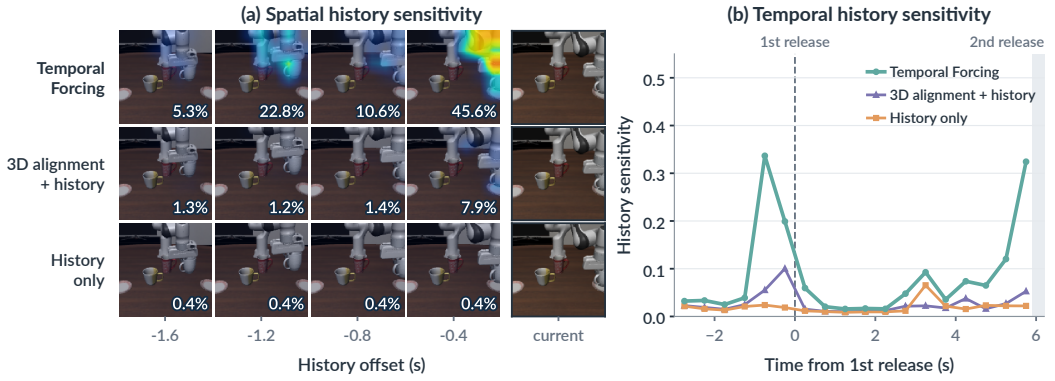

Figure 4: **History utilization under controlled interventions.** (a) Patch-level responses: warmer colors indicate larger relative action changes on a shared 0–50% scale; numbers show the maximum response per frame. (b) Relative responses summed over six separate frame interventions and averaged across six episodes aligned to the first release. The dashed line marks the first release; shading begins at the earliest second release.

Geometric Targets. Context-conditioned 4D targets outperform framewise 3D targets under both alignment configurations in Table 2(a). We obtain the 3D targets by processing each frame independently with the same StreamVGGT architecture and pretrained weights. With current-frame alignment alone, 4D targets raise the average success rate from 87.7% to 90.5% (rows 2 and 3). The advantage grows to 8.6 percentage points in the full history-alignment model, from 85.0% to 93.6% (rows 7 and 8).

Latent-Transition Supervision. We next examine the terms of $\mathcal{L}_{\text{temp}}$ in Table 2(b), keeping the history pathway, 4D targets, and current-frame alignment fixed. Removing the change loss while retaining state and readout supervision (row 9) reduces average success from 93.6% to 91.5% and Long success from 83.8% to 76.0%. Change supervision alone (row 10) achieves 93.2% average success and 84.8% on Long, showing that matching temporal feature differences provides an effective training signal for the history pathway. Equal weighting of the three terms (row 11) yields 92.2% average success. The default $(1, 4, 1)$ weighting achieves the highest average across suites, while change-only supervision performs best on Long.

History Utilization. To examine what the learned history retains, we linearly probe the 16 history tokens for end-effector and object displacement between the oldest and most recent history frames, using simulator-state targets independent of the 4D foundation model. Action supervision alone yields representations no more predictive than an untrained pathway (R^2 0.44 vs. 0.48), whereas Temporal Forcing makes both displacements substantially more decodable (R^2 0.83 and 0.90; Table 3), showing that history access alone does not determine what the pathway retains.

Table 3: Linear probing of motion over the observed history interval from history tokens on LIBERO-Long (R^2).

Model	End-effector	Object
Untrained pathway	0.48	0.74
History only (row 4)	0.44	0.78
w/o $\mathcal{L}_{\text{temp}}$ (row 5)	0.51	0.64
3D alignment (row 7)	0.53	0.79
Temporal Forcing (row 8)	**0.83**	**0.90**

We then close the injection gate at inference to measure the contribution of the learned history pathway. For the main model, this reduces the average success rate from 98.8% to 97.2% (Table 1). Under the 10k-step protocol, closing the gate reduces the average success rate by 5.4 percentage points for Temporal Forcing (row 8), whereas it slightly increases that of the history-only variant (row 4), by 0.3 points.

We further examine where and when history affects action prediction through interventions on past observations (Fig. 4). Temporal Forcing shows stronger patch-level responses than the history-only and 3D-alignment variants (Fig. 4(a)). Averaged over six episodes aligned to the first release, its response to frame-level interventions rises before the first release, drops immediately afterward,

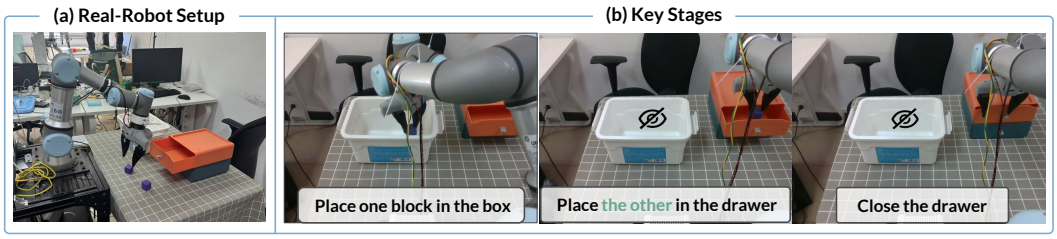

Figure 5: **Real-robot hidden-placement task.** (a) Experimental setup. (b) Three sequential stages. The crossed-out eye indicates that the first block is hidden in the box.

Table 4: **Real-robot performance (successes/trials).** Isolated stages are evaluated independently; sequential columns report cumulative success through the first, second, and third stages. Best results in each column are in bold.

Method	Isolated stages				Sequential task		
	Box	Drawer	Close	Total	S_1	$S_{1:2}$	$S_{1:3}$
OpenVLA	24/30	13/30	21/30	58/90	18/30	5/30	0/30
π_0	26/30	19/30	25/30	70/90	22/30	10/30	4/30
StarVLA-OFT (base)	28/30	21/30	**26/30**	75/90	25/30	12/30	6/30
Temporal Forcing (Ours)	**29/30**	**24/30**	25/30	**78/90**	**27/30**	**20/30**	**13/30**

and increases again before the second release (Fig. 4(b)). The influence of history is thus stage-dependent, with stronger responses around the placement transitions.

4.4 Real-Robot Experiments

We conduct real-robot experiments on a UR3 arm operating at 10 Hz, with a static RealSense D435 and a wrist-mounted fisheye camera. The task involves placing one of two identical blocks into an opaque box, placing the other into a drawer, and then closing the drawer (Fig. 5). The first block becomes hidden after placement, making this a multi-stage setting in which earlier interactions are no longer fully visible as execution proceeds. All compared models are fine-tuned on the same 100 demonstration trajectories. For each model, we evaluate each stage independently over 30 trials, using a stage-specific instruction and only the objects required for that stage. We also evaluate the complete sequence over 30 trials without intervention and report cumulative stage success, where a stage counts as successful only when all preceding stages have also succeeded.

As shown in Table 4, Temporal Forcing and the base model achieve similar success counts on the isolated stages (78/90 vs. 75/90) and the first stage of the sequence (27/30 vs. 25/30). The performance gap becomes larger as sequential execution proceeds: Temporal Forcing completes the first two stages in 20/30 trials, compared with 12/30 for the base model. Full-task success increases from 6/30 (20.0%) to 13/30 (43.3%). This pattern shows that the gains of Temporal Forcing are more pronounced in sequential execution than on the constituent manipulation stages evaluated independently. On a single NVIDIA GeForce RTX 5090 GPU, inference takes 58 ms per action chunk for Temporal Forcing and 45 ms for the base model, both within the 100 ms control interval.

5 Conclusion

We presented Temporal Forcing, which explicitly supervises VLA history representations with causal 4D geometric targets. It aligns both latent temporal states and their transitions during training, while retaining only a compact History Pathway at inference. Across LIBERO, RoboTwin 2.0, and real-robot sequential manipulation, Temporal Forcing consistently improves performance, and controlled interventions confirm that the learned history directly contributes to action prediction. These results suggest that effective history-aware VLAs require not only access to past observations, but also explicit supervision of what their history representations retain.

AI USE STATEMENT

Large Language Models (LLMs) were used for grammar and wording polishing, literature search, and feedback on experimental design and result interpretation. All experiments were conducted by the authors, and all research content and contributions are entirely the responsibility of the authors.

REPRODUCIBILITY STATEMENT

Sections 3 and 4, and Appendix A provide the model, training, evaluation, and 4D-target construction details, while Appendix B documents the controlled analyses and representation probes. The supplementary material includes real-robot rollout videos and an implementation of the proposed Temporal Forcing components. The complete codebase will be released upon publication.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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. *arXiv preprint arXiv:2410.24164*, 2024.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

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.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing, 2026. URL https://arxiv.org/abs/2604.05014.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Minghao Guo, Meng Cao, Jiachen Tao, Rongtao Xu, Yan Yan, Xiaodan Liang, Ivan Laptev, and Xiaojun Chang. Glad: Geometric latent distillation for vision-language-action models. *arXiv preprint arXiv:2512.09619*, 2025.

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. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. Hamlet: Switch your vision-language-action model into a history-aware policy. In *International Conference on Learning Representations*, volume 2026, pp. 101537–101558, 2026.

Chengmeng Li, Junjie Wen, Yaxin Peng, Yan Peng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *IEEE Robotics and Automation Letters*, 11(3):2506–2513, 2026a.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 132324–132345, 2026b.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

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.

Ying Li, Xiaobao Wei, Jiajun Cao, Hao Wang, Xiaowei Chi, Chengyu Bai, Qianpu Sun, Jiajun Li, Xiaojie Zhang, Jian Tang, et al. Wam4d: Fast 4d world action model via spatial register tokens. *arXiv preprint arXiv:2606.14048*, 2026c.

Yixuan Li, Yuhui Chen, Mingcai Zhou, Haoran Li, Zhengtao Zhang, and Dongbin Zhao. Qdepth-vla: quantized depth prediction as auxiliary supervision for vision-language-action models. *arXiv preprint arXiv:2510.14836*, 2025.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Mengya Liu, Baoxiong Jia, Jiangyong Huang, Jingze Zhang, and Siyuan Huang. Lara: Latent action representation alignment for vision-language-action models. *arXiv preprint arXiv:2606.07100*, 2026.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

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.

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.

Hao Shi, Weiye Li, Bin Xie, Yulin Wang, Renping Zhou, Tiancai Wang, Xiangyu Zhang, Ping Luo, and Gao Huang. Memoryvla++: Temporal modeling via memory and imagination in vision-language-action models. *arXiv preprint arXiv:2606.09827*, 2026a.

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. In *International Conference on Learning Representations*, volume 2026, pp. 18567–18602, 2026b.

Ishika Singh, Ankit Goyal, Stan Birchfield, Dieter Fox, Animesh Garg, and Valts Blukis. Og-vla: Orthographic image generation for 3d-aware vision-language action model. *arXiv preprint arXiv:2506.01196*, 2025.

Guoheng Sun, Tingting Du, Kaixi Feng, Chenxiang Luo, Xingguo Ding, Zheyu Shen, Ziyao Wang, Yexiao He, and Ang Li. Rocket: Residual-oriented multi-layer alignment for spatially-aware vision-language-action models. *arXiv preprint arXiv:2602.17951*, 2026.

Lin Sun, Bin Xie, Yingfei Liu, Hao Shi, Tiancai Wang, and Jiale Cao. Geovla: Empowering 3d representations in vision-language-action models. *arXiv preprint arXiv:2508.09071*, 2025.

Octo Model Team, 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. *arXiv preprint arXiv:2405.12213*, 2024.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.

Haoyu Wu, Diankun Wu, Tianyu He, Junliang Guo, Yang Ye, Yueqi Duan, and Jiang Bian. Geometry forcing: Marrying video diffusion and 3d representation for consistent world modeling. *arXiv preprint arXiv:2507.07982*, 2025.

Guoyang Xia, Fengfa Li, Hongjin Ji, Lei Ren, Fangxiang Feng, Kun Zhan, and Yan Xie. Vlaflow: A unified training framework for vision-language-action models via co-training and future latent alignment. *arXiv preprint arXiv:2607.01586*, 2026.

Jiasong Xiao, Yutao She, Kai Li, Yuyang Sha, and Ziang Cheng. Stemvla: An open-source vision-language-action model with future 3d spatial geometry knowledge and 4d historical representation. *arXiv preprint arXiv:2602.23721*, 2026.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

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. *Advances in Neural Information Processing Systems*, 38: 33914–33937, 2025a.

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*, 2025b.

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. *arXiv preprint arXiv:2412.10345*, 2024.

Hanyu Zhou, Chuanhao Ma, and Gim Hee Lee. Vla-4d: Embedding 4d awareness into vision-language-action models for spatiotemporally coherent robotic manipulation. *arXiv preprint arXiv:2511.17199*, 2025.

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

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 *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Implementation and Training Details

4D target construction. We use StreamVGGT to extract geometric targets offline from each training trajectory in causal order. On LIBERO, the model processes causal segments of up to 12 seconds. On RoboTwin and the real robot, it uses the episode prefix available at each anchor. We extract layer-21 features every four timesteps and store them in fp16. For temporal alignment, we spatially average the feature grid at each historical timestep to obtain a pooled vector for each camera, without normalization. For current-frame alignment, we average-pool the 37×37 Dense Geometric Feature grid to 8×8.

We subtract a fixed per-camera mean μ_c, estimated once from the training set of each benchmark, from each pooled vector. The state and readout losses use the layer-normalized centered vectors as targets, while the dense targets are directly layer-normalized. The change loss uses differences between adjacent centered vectors without further normalization, since normalizing each timestep separately would distort their difference. We mask pairs whose target difference has an ℓ_2 norm below 10^{-4}; these mainly occur near the start of an episode, where the history window duplicates frames.

For the framewise 3D control in Table 2(a), we use the same StreamVGGT architecture and pretrained weights but process each frame independently. This comparison isolates the role of temporal context in the geometric targets.

History Pathway. We first encode each past frame at 224×224 resolution using a frozen DINOv2 ViT-L/14. A lightweight Q-Former then compresses its features into two 512-dimensional gist tokens. After adding time-offset and camera embeddings, a two-layer causal Transformer aggregates the ordered token sequence. Sixteen learned queries summarize these features into a fixed set of history tokens, independent of the history-window length. The history tokens are injected into the primary-camera image tokens before the first VLA decoder layer through gated cross-attention. The gate is initialized to zero, so the fusion block initially preserves the base model's output. Current-frame alignment uses VLA layer 24 and StreamVGGT layer 21. During inference, StreamVGGT and the alignment heads are removed, and per-frame gist tokens are cached before adding time-offset embeddings for reuse across control steps.

Training settings. Across experiments, we use a learning rate of 10^{-5} for the Qwen3-VL backbone and 10^{-4} for the action head, the History Pathway, and the alignment heads. Benchmark-specific training, history, and evaluation settings are summarized in Table 5.

On RoboTwin, the History Pathway and both alignment objectives use only the head camera. The frame offsets are the same as on LIBERO and correspond to 3.7 seconds at 15 frames per second. For real-robot fine-tuning, we freeze the backbone vision tower for both Temporal Forcing and the base model. OpenVLA and π_0 are fine-tuned on the same demonstrations using their official recipes.

Computational cost. The History Pathway adds 35.2M trainable parameters (about 0.7% of the base model) and a frozen DINOv2 encoder with 304.4M parameters; the alignment heads (10.5M) are used only during training. Extracting the 4D targets is a one-time preprocessing step that takes 10.9 GPU hours for LIBERO and produces a 68 GiB fp16 cache.

B Additional Analyses

4D target layer selection. We first examine depth readout and motion sensitivity at different StreamVGGT layers in Fig. 6(a). We sample 120 anchor windows from primary-camera trajectories in LIBERO-Spatial, LIBERO-Object, and LIBERO-Long. Each probe window contains eight log-spaced frames at offsets $[-64, -32, -16, -8, -4, -2, -1, 0]$; this sampling is separate from the policy's history window. For each layer, a ridge regressor with an ℓ_2 coefficient of 10 predicts patch-wise log-depth produced by the frozen StreamVGGT depth head. We use 256 sampled current-frame patch features per anchor and split anchors into 80% training and 20% test sets, reporting held-out R^2. Motion sensitivity measures the Pearson correlation between the ℓ_2 feature change over the final two frames and the corresponding patch-pooled absolute pixel change. We compute this correlation across spatial patches and average it over anchors.

Table 5: Training settings across benchmarks. Main and Controlled denote the main LIBERO experiments and the 10k-step controlled studies, respectively.

	LIBERO		RoboTwin	Real robot
	Main	Controlled		
Training steps	50k	10k	10k	8k
Global batch size	128	64	64	48
GPUs	8×A100	4×A800	4×A800	4×A800
Control frequency (Hz)	20	20	15	10
Window offsets (frames)	$-56 \ldots -8$		$-56 \ldots -8$	$-28 \ldots -4$
Window span (s)	2.8	2.8	3.7	2.8
History cameras	2	2	1 (head)	2
Alignment cameras	2	2	1 (head)	2
Anchor stride	4	4	4	4
Causal context	12 s	12 s	episode	episode
Action dim / horizon	7 / 8	7 / 8	14 / 16	7 / 8

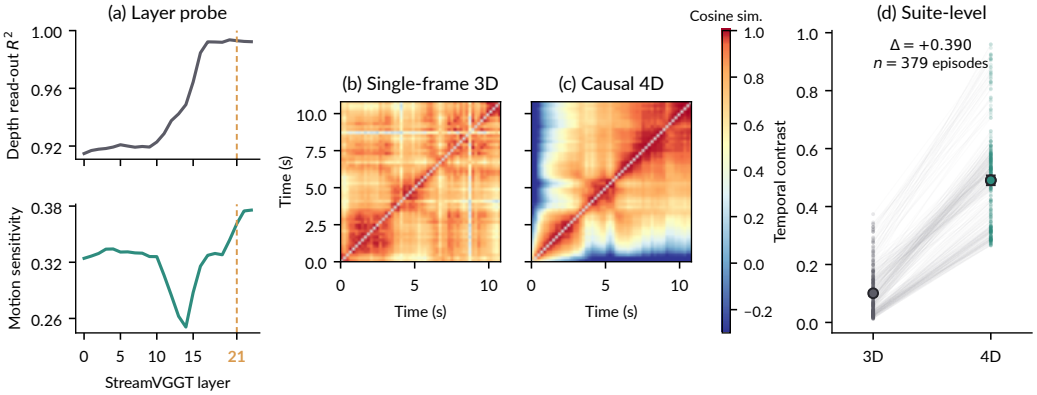

Figure 6: Analysis of StreamVGGT features. (a) Depth readout and motion sensitivity across layers. (b–c) Pairwise cosine similarity of framewise 3D and causal 4D targets from the same trajectory, with diagonals masked. (d) Temporal contrast on LIBERO-Long.

Depth readout saturates in the late layers, while motion sensitivity continues to vary. We select layer 21, the earliest layer to reach both depth $R^2 \geq 0.99$ and motion correlation ≥ 0.36. The final two layers increase motion correlation only modestly, to at most 0.376.

Temporal structure of geometric targets. Rows 2 and 7 of Table 2(a) replace the causal 4D targets with framewise 3D targets and perform worse than their 4D counterparts. To examine this difference, we compare the two targets from the same trajectory in Fig. 6(b–c). The 3D targets assign high similarity to many temporally distant observations, whereas the causal 4D targets separate early states from later ones. To quantify this difference, we define temporal contrast as the mean cosine similarity between nearby frames, with a lag of at most one eighth of a segment, minus that between distant frames, with a lag of at least one third. Over the first 60-anchor (12-second) segment of each of 379 LIBERO-Long episodes, temporal contrast increases from 0.101 for 3D targets to 0.491 for causal 4D targets (Fig. 6(d)). The causal targets thus distinguish states at different points in a trajectory more clearly, whereas differences between framewise 3D targets carry less information about how the scene has evolved.

Gate dynamics. Fig. 7(a) shows how the injection gate develops during training. With the action loss alone (row 4 of Table 2(a)), the gate magnitude remains near 0.6×10^{-3}. Adding current-frame alignment without temporal alignment (row 5) opens the gate temporarily, but its magnitude then falls to 1.1×10^{-3}, indicating that the model learns to down-weight the unsupervised pathway. With

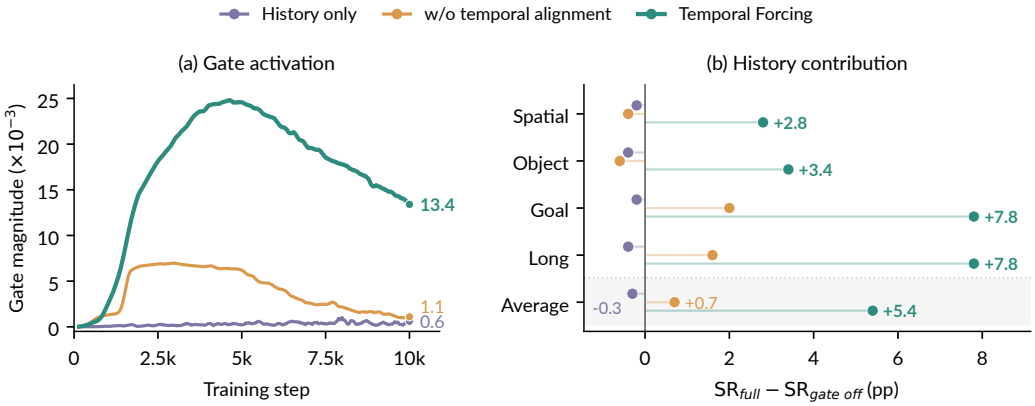

Figure 7: History utilization under the 10k-step controlled protocol. (a) Injection-gate magnitude during training. (b) Success-rate difference between the original policy and the same policy with its gate closed at inference.

temporal alignment (row 8), the gate reaches 13.4×10^{-3}, about 22 times the action-loss-only value, and remains open.

History removal at inference. To evaluate the contribution of the History Pathway, we set the injection gate to zero while keeping the current observation and model weights unchanged. The gated fusion block then becomes an identity mapping. We report the success-rate difference $\Delta\mathrm{SR} = \mathrm{SR}_{\mathrm{full}} - \mathrm{SR}_{\mathrm{gate\ off}}$ in Fig. 7(b). Under the 10k-step protocol, this difference is -0.3 percentage points for the history-only model, whose success slightly increases on all four suites when the gate is closed, and $+0.7$ for the variant without temporal alignment. For Temporal Forcing, closing the gate reduces average success by 5.4 percentage points, with the largest drops on Goal and Long (7.8 points each). These results show that the aligned History Pathway contributes to task success at inference, beyond its role during training.

History-representation probes. Table 3 probes the 16 history tokens M_t produced by the models in Table 2 under the 10k-step protocol. Each checkpoint is frozen and evaluated with the same history window used by the policy (offsets -56 to -8 frames, both cameras). We record M_t every eighth timestep after a full history window is available and flatten the 16 tokens into an 8,192-d feature. From the ten LIBERO-Long tasks, we use 12 demonstrations per task and split the resulting episodes into 70, 20, and 30 episodes for probe training, validation, and testing, respectively. The split is shared across all models.

The probe targets measure motion between the oldest and most recent observed history frames. The end-effector target is the displacement magnitude $\|\mathbf{p}_{t-8} - \mathbf{p}_{t-56}\|_2$, computed from recorded proprioception. The object target is the maximum displacement magnitude over all task objects, $\max_j \|\mathbf{q}^j_{t-8} - \mathbf{q}^j_{t-56}\|_2$, where object positions are recovered from the recorded simulator states by loading each state into the LIBERO environment.

We fit ridge regressors to standardized, PCA-reduced 256-d features, with both standardization and PCA fitted on the probe-training episodes only. The regularization strength is selected from $\{0.01, 0.1, 1, 10, 100\}$ using validation R^2, and test R^2 is reported. The untrained-pathway control reinitializes the Q-Former, temporal Transformer, and output queries of the Temporal Forcing checkpoint, yielding history tokens produced by randomly initialized temporal modules on top of frozen DINOv2 features.

Interventions on past observations. For Fig. 4, each intervention replaces part of one input frame with the content 0.4 s earlier (0.4 s later for the oldest slot) and keeps all other inputs and weights unchanged. The response is the relative change of the predicted action chunk in the normalized action space, $r = \|\hat{\mathbf{A}}'_t - \hat{\mathbf{A}}_t\|_F / \|\hat{\mathbf{A}}_t\|_F$. In the frame-level analysis (Fig. 4(b)), both camera images

Table 6: History-window ablation on LIBERO (success rate, %). Every row uses the full Temporal Forcing recipe of Table 2 (row 8) under the 10k-step protocol and changes only the history window: K past frames per camera at spacing Δt, spanning the stated interval before the current step (LIBERO runs at 20 frames per second). The teal row is the default configuration. Best results in each column are in bold.

History window			LIBERO suite				
Window (s)	K	Δt (s)	Spatial	Object	Goal	Long	Average
$[-1.4, -0.2]$	7	0.2	88.6	87.6	84.2	53.8	78.6
$[-2.8, -0.2]$	14	0.2	**99.0**	99.4	93.4	66.6	89.6
$[-2.8, -0.4]$	7	0.4	98.4	**99.6**	92.6	**83.8**	**93.6**
$[-3.2, -0.8]$	4	0.8	98.6	**99.6**	83.2	79.4	90.2
$[-5.6, -0.8]$	7	0.8	97.2	98.6	**96.8**	81.8	**93.6**
$[-11.2, -1.6]$	7	1.6	95.4	99.2	96.6	72.2	90.9

Table 7: Complete RoboTwin 2.0 success rates (%) in the easy (clean) configuration. Temporal Forcing is evaluated over 100 episodes per task. Best results in each row are in bold.

Task	π_0	DP3	StarVLA-OFT	Ours
Click Bell	44	90	94	**100**
Place Burger Fries	80	72	**96**	92
Place Cans Plasticbox	34	48	81	**88**
Stack Blocks Two	42	24	83	**86**
Handover Mic	98	**100**	39	79
Place Bread Skillet	23	19	56	**59**
Place Bread Basket	17	26	52	**58**
Blocks Ranking RGB	19	3	45	**57**
Handover Block	45	**70**	0	44
Stack Blocks Three	17	1	**41**	40
Blocks Ranking Size	7	2	27	**29**
Place Dual Shoes	15	13	**28**	22
Average	36.8	39.0	53.5	**62.8**

of one history slot are replaced, and responses are summed over the six slots from -2.8 to -0.8 s and averaged over six episodes. In the patch-level analysis (Fig. 4(a)), each of the 7×7 patches of the primary-camera image is replaced in turn, for each history slot and the current frame, on a single episode; the figure shows the four most recent slots.

History window. With the model components and geometric targets fixed, we vary the temporal coverage and sampling density of the history window in Table 6. With $K = 7$ frames, increasing the maximum lookback from 1.4 s to 2.8 s raises the average success rate from 78.6% to 93.6%. Extending it to 5.6 s maintains the same average, while 11.2 s reduces it to 90.9%. At the 2.8 s horizon, halving the sampling interval to 0.2 s (14 frames) reduces the average to 89.6% and Long success from 83.8% to 66.6%, and doubling it to 0.8 s (4 frames, 3.2 s lookback) reduces the average to 90.2%, mainly on Goal (83.2%). Longer or denser history is therefore not uniformly better. We use seven frames at 0.4 s intervals, which match the best average performance and achieve the highest success rate on Long.

C Additional Results and Visualizations

Complete RoboTwin results. Table 7 reports results on all twelve evaluated tasks. Temporal Forcing improves over the base model on nine tasks, increasing average success from 53.5% to 62.8%. The largest gains are on Handover Block and Handover Mic, from 0% to 44% and from

Table 8: Zero-shot LIBERO-Plus success rates (%). All models are trained on standard LIBERO only. Total pools results over 10,030 perturbed tasks, rather than averaging the seven dimensions equally. Baseline results for OpenVLA (Kim et al., 2024), OpenVLA-OFT (Kim et al., 2025), π_0 (Black et al., 2024), and π_0-Fast (Pertsch et al., 2025) are taken from LIBERO-Plus (Fei et al., 2025). Parentheses report gains over StarVLA-OFT in percentage points. Best results in each column are in bold.

Method	Camera	Robot	Language	Light	Background	Noise	Layout	Total
OpenVLA	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
OpenVLA-OFT	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6
π_0	13.8	6.0	58.8	85.0	81.4	79.0	68.9	53.6
π_0-Fast	**65.1**	21.6	61.0	73.2	73.2	74.4	68.8	61.6
StarVLA-OFT (base)	47.0	60.1	87.0	**96.3**	95.3	73.1	79.2	75.0
Temporal Forcing	47.2 (+0.2)	**63.7** (+3.6)	**88.7** (+1.7)	**96.3** (+0.0)	**97.4** (+2.1)	**81.3** (+8.2)	**81.6** (+2.4)	**77.8** (+2.8)

39% to 79%, respectively. Both tasks involve object occlusion during transfer between the arms. The three regressions occur on tasks in which completed stages remain visible.

Zero-shot robustness on LIBERO-Plus. LIBERO-Plus tests static distribution shifts rather than history-dependent decisions, so we use it to check that the History Pathway does not reduce robustness. Without additional training, Temporal Forcing improves pooled success from 75.0% to 77.8% (Table 8). It improves six of the seven dimensions and matches the base model on lighting. The largest gain occurs under sensor noise, where success increases from 73.1% to 81.3%.

Qualitative sequential rollouts. Fig. 8 compares complete execution sequences. After the first block is placed into the box, it is no longer visible, and the task proceeds to the remaining placement and drawer-closing stages. Temporal Forcing selects the remaining block, places it into the drawer, and closes the drawer without a reset. StarVLA-OFT also selects and transports the correct block, but releases it on top of the drawer during the second stage. Across the real-robot trials, Temporal Forcing and the base model complete 78/90 and 75/90 isolated stages, respectively, while their full-task successes are 13/30 and 6/30. Together with the isolated-stage results, these rollouts illustrate that the larger performance difference emerges during continuous multi-stage execution rather than from a large difference in individual manipulation skills.

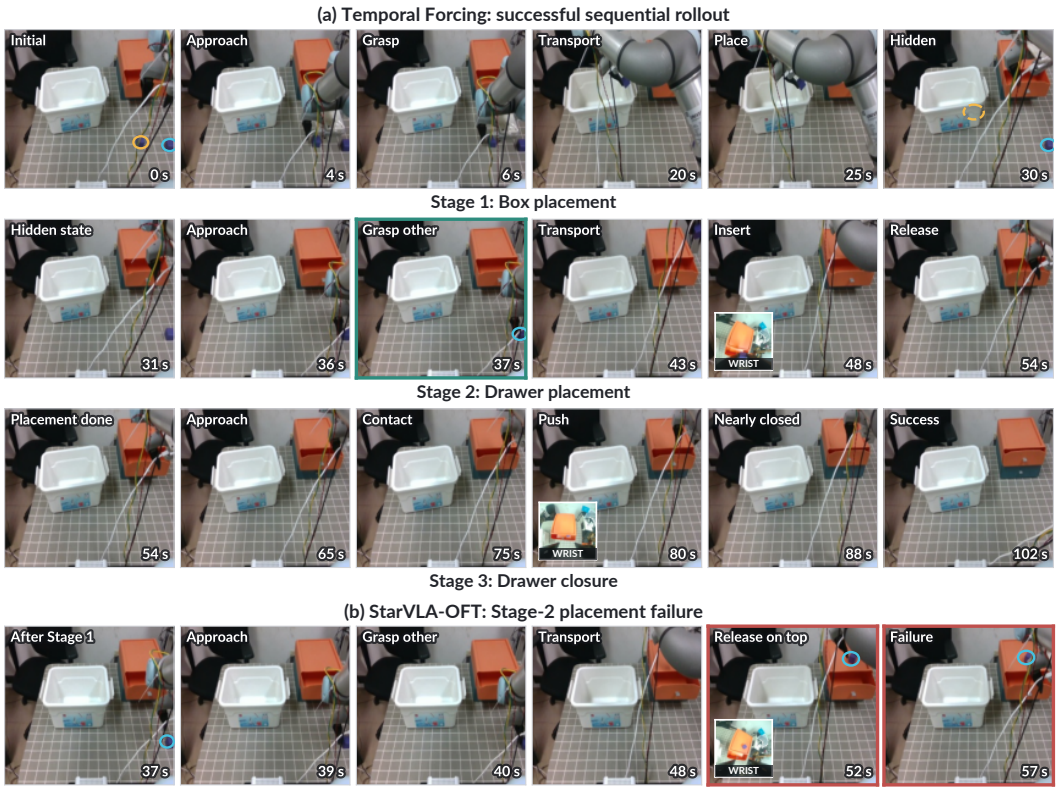

Figure 8: Sequential real-robot rollouts. (a) Temporal Forcing completes all three stages without a reset. (b) StarVLA-OFT releases the second block on top of the drawer. Green and red borders mark key decision and failure states. Circles identify the two blocks, and wrist-camera insets show occluded manipulation details. All annotations are for visualization only.