Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

AR-VLA: Autoregressive Action Expert for Vision–Language–Action Models

Yutong Hu*†§ Jan-Nico Zaech* Nikolay Nikolov* Yuanqi Yao* Sombit Dey* Giuliano Albanese*
Renaud Detry†‡§ Luc Van Gool* Danda Paudel*
*INSAIT, Sofia University "St. Kliment Ohridski"
†KU Leuven, Dept. Mechanical Engineering, Research unit Robotics, Automation and Mechatronics
‡KU Leuven, Dept. Electrical Engineering, Research unit Processing Speech and Images
§Flanders Make@KU Leuven

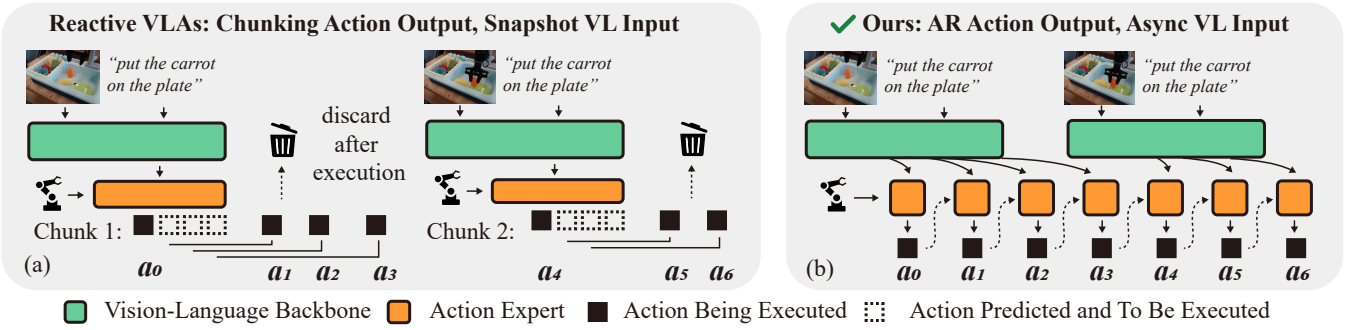


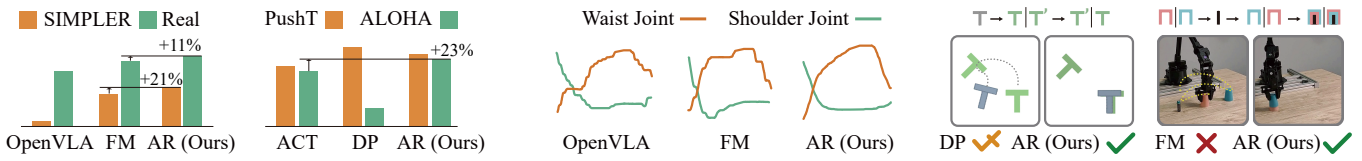
Fig. 1: (a) The prevalent approach in Vision-Language-Action models predicts action chunks based only on the current snapshot of information. It discards the temporal context and drops the state update within an action chunk execution, leading to reactive, memoryless prediction. (b) In contrast, we propose AR-VLA that leverages an autoregressive action expert that maintains its own history through a long-lived memory with inherent context-awareness. Visual-language conditions are updated asynchronously without interrupting the action stream.

***Abstract*—We propose a standalone autoregressive (AR) Action Expert that generates actions as a continuous causal sequence while conditioning on refreshable vision-language prefixes. In contrast to existing Vision-Language-Action (VLA) models and diffusion policies that reset temporal context with each new observation and predict actions reactively, our Action Expert maintains its own history through a long-lived memory and is inherently context-aware. This structure addresses the frequency mismatch between fast control and slow reasoning, enabling efficient independent pretraining of kinematic syntax and modular integration with heavy perception backbones, naturally ensuring spatio-temporally consistent action generation across frames. To synchronize these asynchronous hybrid V-L-A modalities, we utilize a re-anchoring mechanism that mathematically accounts for perception staleness during both training and inference. Experiments on simulated and real-robot manipulation tasks demonstrate that the proposed method can effectively replace traditional chunk-based action heads for both specialist and generalist policies. AR-VLA exhibits superior history awareness and substantially smoother action trajectories while maintaining or exceeding the task success rates of state-of-the-art reactive VLAs. Overall, our work introduces a scalable, context-aware action generation schema that provides a robust structural foundation for training effective robotic policies. Code and Videos available at https://arvla.insait.ai/**

I. Introduction

The "next-token prediction" paradigm has emerged as one of the primary engines of modern artificial intelligence. Large-scale autoregressive models, such as LLMs [8] and VLMs [3], demonstrate that the synergy of causal sequence modeling, scalable attention, and massive computation is essential for the appearance of emergent reasoning and robust generalization. Naturally, this paradigm is now being extended from sequences of words to sequences of actions via Vision-Language-Action (VLA) models. However, while recent VLA architectures (e.g., OpenVLA [19], RT-2 [50], Pi-0-FAST [31]) are frequently labeled "autoregressive", this terminology is deceptive in the context of robotic control. These models utilize autoregression only to generate tokens within a single inference step. Effectively, they do not autoregress across time.

Current state-of-the-art robot learning methods, including Diffusion Policies [9] and existing VLAs, treat action generation not as a continuous stream, but as a series of isolated events. As shown in Fig. 1(a), these models typically employ "action chunking" [47]: predicting a static block of actions at once, directly or through iterative denoising. While effective for short-horizon smoothness, these approaches remain structurally reactive: at every perception step, the model acts as if it is "waking up" for the first time, re-encoding the visual context and generating a trajectory chunk without a persistent internal state of its own perception and action history. Consequently, they suffer from "Markovian amnesia", discarding temporal continuity and degrading fluid control to a series of disjointed, snapshot-conditioned responses.



(a) Benchmark Performance **(b) Execution Smoothness** **(c) Context-awareness**

Fig. 2: **Performance Overview.** (a) Quantitative Results: In both generalist (left) and specialist (right) benchmarks, AR-VLA achieves competitive or superior performance compared to state-of-the-art policies, including OpenVLA, Flow-Matching (FM), ACT, and Diffusion Policy (DP), details in Sec.IV-A. (b) Trajectory Quality: Qualitative visualization of joint trajectories over time reveals that AR-VLA produces significantly smoother and more kinematically consistent motion compared to reactive baselines that reset context at each step (analysis in Sec.IV-B). (c) Long-Horizon Capability: AR-VLA successfully completes long-horizon tasks where baselines like DP and FM fail due to a lack of temporal context awareness. Detailed task defination and explanation in Sec. IV-C.

We argue that manipulation is not merely a stack of separate visual-motor snapshots; it is a problem of streaming control. To act effectively, a policy requires two distinct forms of awareness: **situational** awareness (semantic understanding of "what" is in the workspace and "where" the robot is) and **temporal** awareness (kinematic understanding of "what" has already occurred and "how" the end-effector is accelerating). While VLMs provide the former, they are structurally ill-suited for the latter due to their high latency and episodic nature. The missing piece here is a truly Autoregressive Action Expert – just as an LLM predicts the next word based on the "flow" of a conversation, a robot policy should predict the next pose based on the "momentum" of its trajectory.

By treating action as a "language of motion", a true Autoregressive Action Expert provides three transformative benefits to the VLA paradigm, as in Fig. 2. **(1) It is naturally context-aware,** as its internal state captures the causal dependencies of the entire trajectory rather than reacting to a local snapshot. **(2) It is naturally decoupled from the VLM backbone,** allowing the motor thread to run at high frequencies with temporal consistency, regardless of perception latency. **(3) It facilitates independent pretraining using only the action labels**, enabling the model to master the syntax of movement (dynamics, joint constraints, and physical causality) on large-scale kinematic data before the visual alignment phase.

To realize these potentials, we introduce AR-VLA, a unified framework that instantiates such an action expert within a single architecture for both robot specialists and generalists. As in Fig. 1(b), AR-VLA structurally decouples the high-level semantic reasoning of vision-language models from the high-frequency temporal consistency of robot control. Rather than treating the action head as a dependent appendage of a VLM, we formulate it as an independent expert that maintains a continuous, evolving memory of its own history. This design preserves long-horizon intent while allowing the model to asynchronously attend to the latest visual-language features provided by a VLM. This architecture bridges a fundamental frequency mismatch in robotics, providing a solution one step closer to the **system 1/2** [16, 2] dichotomy: the "brain" (semantic perception) updates slowly, while the "cerebellum" (motor control) streams high-frequency commands.

Our core contribution is **the formulation of an Autoregressive Action Expert**, which treats action generation as a causal sequence modeling problem across time. By maintaining a long-lived context of past actions, our model inherently resolves the temporal inconsistency and "jitter" prevalent in reactive policies, outperforming denoise/chunk-based baselines in trajectory smoothness and long-horizon stability. To instantiate this expert, we propose two technical pillars: **(1) Hybrid Key-Value (HKV) Cache:** A novel Transformer decoder architecture that manages two distinct memory streams: a rolling, token-wise FIFO for high-frequency actions and a block-wise, refreshable buffer for low-frequency visual semantics. This allows the action stream to function as an independent expert that is "guided" rather than "blocked" by perception. **(2) Dynamic Temporal Re-anchoring (DTR):** We solve the synchronization challenge of asynchronous streams via DTR, a mechanism that explicitly anchors visual keys based on their capture-time index. This ensures the model mathematically understands the "staleness" of a visual frame, bridging the gap between short-context training and long-horizon inference.

II. Related Work

A. *Vision-Language-Action models (VLAs).*

Traditional robot imitation learning [34, 10, 9, 21, 36] has typically relied on task-specific data, which covers only narrow distributions of environments and instructions. To mitigate this, recent research [50, 12, 19, 31, 4, 38, 5, 13, 45, 33, 43, 17] suggests constructing generalist policies by injecting internet-scale VLM priors into low-level action generation, effectively transferring broad semantic understanding to physical control. Initial VLA models [50, 19, 11, 12, 20, 7] often discretized action spaces to treat control as a token prediction task. Subsequent developments [5, 4, 24, 6, 37] have utilized VLM embeddings to condition continuous action generators via diffusion or flow-matching. For example, π_0 [5] conditions a flow-matching head on VLM features to generate multi-step action chunks, while CogAct [24] demonstrates the scalability of diffusion action transformers when grounded in VLM representations. However, these architectures remain predominantly reactive, basing control decisions on the immediate observation and often ignoring the temporal context vital for accurate state estimation. In this work, we focus on enhancing current VLAs by integrating persistent historical context into

the autoregressive process, without changing any of the Vision-Language perception part.

B. Action Representation and Pretraining.

Ensuring that features from different domains reside in well-structured representation spaces is foundational for effective cross-modality alignment. In the action domain, given its relatively low dimensionality, this is traditionally achieved through classical methods such as statistical normalization [5], categorical binning [19], or discretization via k-means clustering [34]. Distinct from these approaches, there is a growing interest in modern representation learning that treats actions as a temporal sequence. By leveraging large-scale trajectory datasets prior to visual conditioning, these models learn to capture the fundamental motion primitives and dynamics of the embodiment. Recent advances in action tokenization demonstrate that robust motion priors can be learned efficiently through explicit modeling, such as Fast [31], FASTer [26], BEAST [49], and OmniSAT [27] or implicit neural architectures like Vector Quantized Variational Autoencoders (VQ-VAE) [21, 44]. These approaches successfully capture the underlying syntax of robot kinematics to facilitate the final action prediction. Our approach extends this idea further by treating the pretrained action model not merely as a passive token translator, but as a standalone autoregressive expert. This formulation enables the model to perform implicit sequence modeling of motion priors while simultaneously allowing for asynchronous coupling with high-latency, heavy perception modules.

C. Architectures with context awareness

While many current robotic datasets and benchmarks are limited to short-horizon tasks, most real-world applications are inherently long-horizon and non-Markovian, necessitating memory mechanisms that allow policies to utilize historical observations and actions. In reinforcement learning, recurrent policies [14] and Transformer-based variants [30] established memory as a primary tool for performance in partially observable environments. Similarly, modern natural language models leverage KV-caching to remain aware of historical context. Explicit memory structures, ranging from end-to-end memory networks [40] to retrieval-based models [18, 22], have been thoroughly investigated to bolster long-context reasoning. Despite these developments, memory mechanism in VLA models remains under-explored. While most VLAs are context-unaware, MemoryVLA [35] proposes architectures inspired by human cognitive systems but requires training from scratch. HAMLET [28] augments pretrained VLAs with learnable tokens and memory modules to achieve history-awareness without full retraining. Distinct from these, our approach utilizes a true autoregressive model for the action expert, making it naturally holds a record of the system's evolution from historical states and providing an innate context awareness.

III. Methodology

The AR-VLA framework bridges the divide between high-latency semantic perception and high-frequency motor control through a stateful, two-stage architecture. At its core is a standalone autoregressive action expert that maintains kinematic continuity via a **Hybrid Key-Value Cache (HKV)**, allowing it to marry a rolling proprioceptive history with a refreshable visual-linguistic context. To align these asynchronous streams, we introduce **Dynamic Temporal Re-anchoring (DTR)**, a position-encoding mechanism that maps static visual-language features onto the dynamic action timeline. Our training protocol unfolds in two phases: (1) action-only pretraining to master the syntax of motion, and (2) cross-modal alignment to ground that motion in visual perception.

A. Problem Formulation

We formalize a robot trajectory $\{\tau\}$ as a sequence of observations and actions $\{(o_t, a_t)_{t=0}^{T}\}$. The observation o_t is decomposed into exteroceptive inputs (visual frames v_t and language instructions l) and proprioceptive states s_t.

A common workaround to circumvent the limitations of stateless architectures involves stacking observations temporally and predicting actions in chunks. While this constructs a "pseudo-history," it still fails to cure the underlying Markovian bottleneck; the model have to re-infer history intent and even current velocity from scratch at every new window, often resulting in jittery control and temporal incoherence.

Definition 1: The Reactive Actor. Standard VLAs typically map the *current* observation to the *current* action, resetting their context memory at each step t:

$$P_{\text{react}}(\tau) = \prod_{t=1}^{T} P(a_t \mid \Phi(v_t, l), s_t), \tag{1}$$

where $\Phi(\cdot)$ denotes the perception encoder (e.g., a VLM or a smaller backbone) used to embed the observations.

Definition 2: The Autoregressive (AR) Actor. We define the AR Actor as a sequence model where one of the prediction dependencies is the continuous kinematic history, while remaining conditioned on the most recently available visual-language prefix from Φ. Let $i \leq t$ be the index of the most recent visual frame processed:

$$P_{\text{ar}}(\tau) = \prod_{t=1}^{T} P(a_t \mid \underbrace{\Phi(v_i, l)}_{\text{VL Prefix}}, \underbrace{a_{<t}, s_{<t}}_{\text{Kinematic History}}). \tag{2}$$

In this formulation, a_t depends explicitly on the continuous causal chain $a_{<t}, s_{<t}$. This persistent memory ensures kinematic smoothness and robustness to visual-language (VL) latency.

B. Model Structure

AR-VLA is instantiated as a unified Transformer decoder designed to process a hybrid stream of continuous proprioceptive data and high-dimensional VL embeddings.

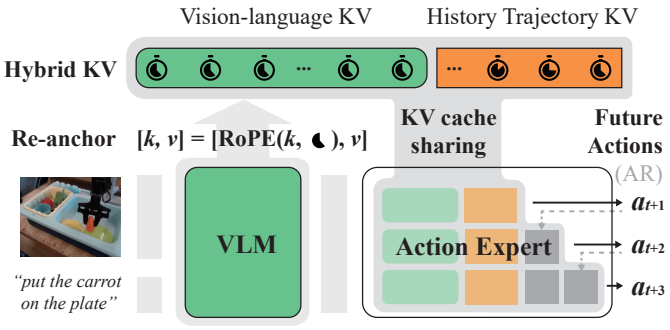


Fig. 3: **The AR-VLA Framework**. The system bridges an VLM backbone with a autoregressive Action Expert asynchronously. Atemporal features from the VLM are explicitly injected with temporal context via Dynamic Temporal Re-anchoring (DTR). Within the Hybrid KV Cache, re-anchored VL tokens (green) serve as a semantic prefix to the rolling kinematic history (orange). The Action Expert generates future action sequences by querying this shared cache using incrementally advancing step embeddings.

Continuous Action Representation. To preserve the precision required for low-level manipulation, we follow the convention of continuous action regression. Yet the AR training naturally fits discrete token supervision. Robot actions $a_t \in \mathbb{R}^d$ represent target end-effector pose deltas or joint velocities. Within the Transformer, these vectors are projected to the model dimension D via a linear layer, treating each timestep's vector as a single token. The model output is regressed to a_{t+1} via a deterministic prediction head. Because s and a vary across robots and tasks, we denote arbitrary action-modality tokens as x to maintain a consistent autoregressive notation. In the following part, we use X to mark the proprioceptive stream and VL to mark the language stream.

Unified Decoder with Hybrid Cache. The architecture relies on a Hybrid Key-Value (HKV) Cache that manages two heterogeneous sources of context. As in Fig.3, we apply distinct update rules, enabling the structural decoupling of perception and control: (1) Proprioceptive Stream (KV^X): A rolling FIFO buffer storing the KV pairs of the robot's state and action history. This long-lived window is significantly longer than the history stacks (1∼4) used in reactive VLAs, capturing the momentum necessary for stability. (2) Visual-Language Stream (KV^{VL}): A single-slot buffer storing KV pairs projected from the VLM backbone. This acts as a refreshable semantic prefix, replaced entirely whenever a new frame is processed.

Dynamic Temporal Reanchoring (DTR). The fundamental challenge in decoupling these threads is temporal alignment. We introduce DTR, which leverages the mathematical properties of Rotary Positional Embeddings (RoPE [39]) to encode the relative distance between the high-frequency action stream and the inherently atemporal VL context retrieved from the VLM.

In our unified decoder, the attention output for an action query at temporal index m is calculated over the combined set of cached proprioceptive and VL key-value pairs:

$$\text{Attn}(q_m, \boldsymbol{K}, \boldsymbol{V}) = \sum_n \text{softmax}\left(\frac{\text{Score}(q_m, k_n)}{\sqrt{d}}\right) v_n, \quad (3)$$

where $\text{Score}(q_m, k_n)$ denotes the position-aware inner product. RoPE implements this by applying a rotation matrix $\boldsymbol{R}(p)$ to the feature vectors at position p, such that the attention score depends only on the relative distance. Crucially, because VLM embeddings are generated independently of the robot's trajectory, we manually assign indices to bridge the training-inference gap:

$$k_n = \boldsymbol{R}(n)\boldsymbol{k}, \quad v_n = \boldsymbol{v}, \quad (4)$$

where action tokens $\boldsymbol{k}^X$ and VL tokens $\boldsymbol{k}^{VL}$ are treated differently when assigning their anchor index n: (1) Action Tokens: $\boldsymbol{k}^X$ follow the robot's causal timeline. They are assigned the sequential index n corresponding to the timestep at which they were executed. (2) VL Tokens: $\boldsymbol{k}^{VL}$ from VLM backbone are inherently atemporal. Without extra processing, those tokens only form a static semantic snapshot unaware of the robot's current step. To encode their temporal validity, we assign them the fixed index n corresponding to the timestep when the image was captured.

By defining n in this way, the relative distance $(m-n)$ between the current query m and the VL key n mathematically represents the data staleness, as in Fig.3. The resulting interaction becomes $\text{Score}(q_m, k_n^{VL}) = \boldsymbol{q}^\top \boldsymbol{R}(m-n)\boldsymbol{k}^{VL}$. A vital property of this formulation is that the score remains identical under a global time shift T:

$$\text{Score}(q_{m+T}, k_{n+T}^{VL}) = \text{Score}(q_m, k_n^{VL}). \quad (5)$$

Because the VL values v^{VL} remain constant and un-rotated, the resulting weighted sum in the attention mechanism is purely a function of the relative staleness $\Delta t = m - n$.

This mechanism is critical for resolving the discrepancy between training and deployment. During training, we typically sample short batches (e.g., current step $m = 25$ and an image anchored at $n = 20$). In this scenario, the model learns to act based on a VL context with a staleness of $\Delta t = 5$. During real-world inference, the robot may reach global step 500 while still processing a VL update from step 495. Though the staleness of $\Delta t = 5$ remains in-distribution, the absolute indices $(500, 495)$ would fall far outside the distribution seen during training if we are not using DTR with RoPE, leading to unpredictable behavior. By exploiting the shift-invariance property ($T = 475$), DTR ensures that $\text{Score}(q_{500}, k_{495}^{VL}) = \text{Score}(q_{25}, k_{20}^{VL})$. This allows the model to apply the same visual grounding logic whether a seen situation occurs at step 25 or step 500.

C. Training Details

Our training protocol follows a two-phase regime to master motion syntax before grounding it in perception.

Phase 1: Action-Only Pretraining: The actor is first optimized on large-scale trajectories as a standalone autoregressive

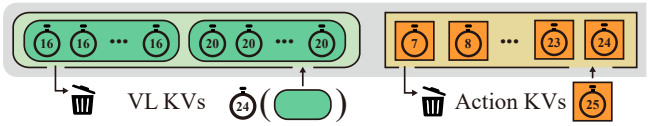


Fig. 4: **Heterogeneous FIFO Update Rules for the Hybrid KV Cache.** The framework manages memory through two distinct queueing strategies to ensure efficient context utilization. The VL Stream (green) operates as a short-lived, block-wise FIFO: In contrast, the Action Stream (orange) maintains a token-wise rolling FIFO, continuously appending the single latest action prediction while evicting the oldest kinematic state.

action sequence generator. Using a causal mask and sequential RoPE indices, we optimize the sequence modeling objective

$$\mathcal{L}_{\text{Phase1}} = \sum_{t=1}^{T} \mathcal{L}(x_t \mid x_{<t}). \tag{6}$$

This establishes a "proprioceptive expert" that masters kinematic syntax (e.g., joint limits, profiles, common move patterns) independently of VL data.

Phase 2: VL-Action Alignment: We connect the VLM backbone to the expert using DTR, as in Fig.3. Given a training sample $(\boldsymbol{x}_{\text{past}}, v, l, \boldsymbol{x}_{\text{fut}})$ where v is an observation at time H, we do: (1) Priming: History $\boldsymbol{x}_{\text{past}}$ is fed into the actor with indices $\{0, \ldots, H-1\}$. (2) Anchoring: VL features from (v, l) are assigned the fixed index H, anchoring the perception to the history-future junction. (3) Stochastic Supervision with Historical Dropout: The model predicts a horizon of M future actions $\boldsymbol{x}_{\text{fut}}$ starting from the temporal anchor. To simulate execution noise and prevent parasitic over-reliance on history, we apply a unique random binary mask $\mathcal{M}_k \in \{0,1\}^H$ for every individual future token $k \in \{0, \ldots, M-1\}$. This forces the model to attend to the VL prefix when historical context is corrupted or missing. The final loss is formulated as,

$$\mathcal{L}_{\text{Phase2}} = \sum_{k=0}^{M-1} \mathcal{L}(x_{H+k} \mid \mathcal{M}_k \odot \boldsymbol{x}_{\text{past}}, \Phi(v_H, l_H), \boldsymbol{x}_{H:H+k-1}). \tag{7}$$

D. Inference Details

Following training, the model effectively functions as a conditional next-token predictor $P(x_t \mid x_{<t}, \Phi(v_i, l_i))$, capable of generating precise actions even when the visual-language prefix $\Phi(v_i, l_i)$ is "outdated" relative to the current timestep t ($i \leq t$). This capability stems from the DTR mechanism, which ensures the attention mechanism generalizes to varying temporal offsets $\Delta t = t - i$, and the teacher-forcing training regime. By supervising the autoregressive prediction of a future horizon rather than a single step, the model learns to utilize visual-language prefixes anchored at various historical offsets (e.g., $t, t-1, \ldots, t-k$), ensuring robustness to latency.

Consequently, the next action prediction relies purely on a hybrid KV cache comprising the up-to-date action domain history and the most recently available visual-language tokens. This runtime Hybrid KV cache is constructed and maintained dynamically during inference, as in Fig. 4. The action domain cache KV^X operates as a persistent FIFO buffer, preserving the long-term history of the trajectory. Conversely, the visual-language cache KV^{VL} is treated as a refreshable snapshot (functionally a single-slot FIFO buffer). Whenever a new visual embedding is available (captured at time i), we explicitly apply the DTR to the keys and replace the KV^{VL} content, effectively "snapping" the perception to the new timestamp.

During deployment, such a hybrid composition of the KV cache allows the VLM and the Action Expert of AR-VLA execute their forward loop asynchronously. Therefore, AR-VLA supports both serial and parallel execution modes via a decoupled dual-thread architecture: **(1) Action Thread:** operates at a high control frequency. It autoregressively generates actions a_t, updates the KV^X cache, and increments the global time index t. **(2) Perception Thread:** operates at the native frequency of the VLM. It processes the latest frame v and asynchronously pushes updates to the KV^{VL} buffer.

This structural decoupling provides a significant advantage over denoise-based chunking models. Even in serial execution, our lightweight action head offers lower latency per step than full denoising inference. In parallel execution, this design further guarantees a consistent control frequency, as the action prediction eliminates the blocking dependency on specific visual-language frames. The model simply conditions on the internal kinematic model (KV^X) and the latest valid VL prefix until a new one becomes available, ensuring a native smooth, uninterrupted actuation.

IV. Experiments

The objective of our experimental evaluation is to assess AR-VLA's capability for both specialist and generalist robot policies. Our comprehensive experiments aim to demonstrate that the autoregressive architecture provides a competitive alternative to existing action experts while offering superior history awareness and high-frequency control capabilities. The experiments are designed to answer the following questions: label=4)

1) How well does AR Actor perform when replacing standard action heads in specialist and generalist policies?
2) How well does AR-VLA perform in terms of inference efficiency and trajectory quality?
3) Can AR-VLA effectively solve long-horizon tasks that require history awareness?
4) How do causal pretraining, RoPE anchoring, training-time history masking and inference-time AR context length contribute to the model's performance?

To answer these questions, as shown in Fig. 5 and Fig. 6, we evaluate AR-VLA across both generalist VLA policies and specialist task-specific scenarios. Firstly, we assess AR-VLA's performance when replacing standard action heads by training on BridgeV2 and evaluating on SimplerEnv and real-world WidowX robot, comparing against existing VLA policies and baseline action experts with identical VLM backbones, as well as on three specialist manipulation tasks comparing with ACT and Diffusion Policy. Secondly, we quantify AR-VLA's efficiency advantages in inference frequency and trajectory

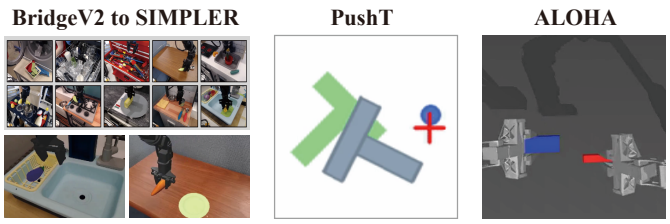


Fig. 5: **Simulation benchmarks setups.** We do simulation evaluation spanning generalist and specialist policies, with diverse embodiment, action space, and task.

smoothness. Then, we design two tasks that explicitly require history awareness: PushT2 and Stack3, where critical information becomes unobservable without maintaining action history. Finally, we conduct ablation studies to verify key design choices including causal pretraining and RoPE anchoring.

A. Generalist and Specialist Policy Performance

Evaluation Setups and Comparisons. We evaluate AR-VLA's capability to replace standard action prediction heads across generalist VLA and specialist policy settings. For generalist VLA policies, we train on the BridgeV2 dataset [42] and create three models of identical 3B + 300M scale sharing the same Paligemma-3B [3] VLM backbone with knowledge insulation [31] strategy: One predicts actions by decoding Fast tokens (i.e., a reproduced Pi-0-FAST*[31]), one predicts action chunks through multi-step flow matching (Pi-0.5*[15]), and one predicts actions autoregressively with standard next-action prediction loss (AR-VLA, Ours). We evaluate on the SimplerEnv simulator [25], which provides WidowX manipulation tasks that replicate real-world robot conditions. We compare against OpenVLA [19], Octo-Base, Octo-Small [41], SpatialVLA [32], and CogACT [23]. For real-world evaluation, we deploy models on the WidowX robot following the BridgeV2 setup at 5Hz with the VLM refreshed every 4 actions. We establish a baseline where all polices succeed on all trials of the in-distribution task "put eggplant into sink," then test on challenging tasks with 4 layout variants repeated 3 times.

For specialist policies, we evaluate AR Actor on PushT, ALOHA cube transfer, and ALOHA peg insertion using LeRobot [1], comparing with Action Chunking Transformer (ACT [47]) and Diffusion Policy (DP [9]). To investigate AR Actor's capability to replace chunk-based action experts, we maintain ACT's exact encoder-decoder architecture and parameter size, but use it in a way following the same design principle as AR-VLA: we treat the encoder as a compact "VLM" that caches intermediate key-values, while the decoder serves as an autoregressive action expert trained with next-token prediction loss. All methods share ResNet-18 vision encoder, and detailed model architecture in Appendix.

SimplerEnv Simulation Performance. Table I presents the evaluation results on SimplerEnv Visual Matching setting with four WidowX manipulation tasks: put spoon on towel, put carrot on plate, stack green block on yellow block, and put eggplant in yellow basket. On average, AR-VLA achieves the highest success rate of 61.5%, significantly outperforming the second-best policy CogACT by a +9.4% margin (52.1%). Notably, despite sharing the identical VLM backbone with Pi-0-Fast and Pi-0.5, AR-VLA demonstrates superior action sequence generation capabilities through stored history key-value caches, surpassing Pi-0-Fast (49.0%) and Pi-0.5 (51.0%) by substantial margins. The performance gains are particularly evident across individual tasks. AR-VLA achieves 75.0% success on the spoon task, exceeding Pi-0-Fast (62.5%) and Pi-0.5 (58.3%). On the carrot task requiring precise manipulation, AR-VLA reaches 54.2% success rate, substantially outperforming Pi-0-Fast (29.2%) and Pi-0.5 (33.3%). These results demonstrate that AR-VLA achieves better action sequence generation based on stored history key-value caches.

TABLE I: BridgeV2 Pretraining to SIMPLER Simulation Success Rate (%)

Model	Spoon	Carrot	Block	Eggplant	Average
OpenVLA[19]	0	0	0	4.1	1.0
Octo-Base[29]	12.5	8.3	0	43.1	16.0
Octo-Small[29]	47.2	9.7	4.2	56.9	30.0
SpatialVLA[32]	16.7	25.0	**29.2**	**100.0**	42.7
CogACT [23]	58.3	37.5	20.8	91.7	52.1
Pi-0-Fast*[31]	62.5	29.2	20.8	83.3	49.0
Pi-0.5*[15]	58.3	33.3	16.7	95.8	51.0
AR-VLA (Ours)	**75.0**	**54.2**	20.8	95.8	**61.5**

TABLE II: Specialist Policy Performance. We report average max IOU and success rate (%) for pushT, and success rate (%) for two ALOHA task training on scripted and human demonstrations. Per seed evaluation results with standard error is accessible from our project website.

Method	**pushT**		**aloha-cube**		**aloha-insert**	
	Max IoU	Success	Script	Human	Script	Human
DP[9]	**0.957**	**65.20**	33.33	10.00	22.67	1.66
ACT[48]	0.800	52.00	86.00	50.00	32.00	**20.00**
AR(Ours)	0.920	60.40	**97.33**	**67.33**	**54.67**	6.67

Real-world WidowX Evaluation. We compare the zero-shot real-world performance of AR-VLA against state-of-the-art generalist policies, including OpenVLA and our reimplementation of the Pi family (Fig. 6). Our Action Expert achieves a superior 89% average success rate, notably reaching 100% success on the "cup on plate" and "Lobster" tasks. A key driver of this performance is the model's inherent temporal awareness: upon an initial failure, AR-VLA gracefully lifts the end-effector to attempt a second trial. In contrast, other baselines often exhibit erratic motions near the target after a failed grasp, frequently pushing the object into configurations from which the policy cannot recover. This demonstrates that our autoregressive approach provides the closed-loop robustness necessary for reliable physical execution.

Specialist Policy Performance. Table II shows AR Actor

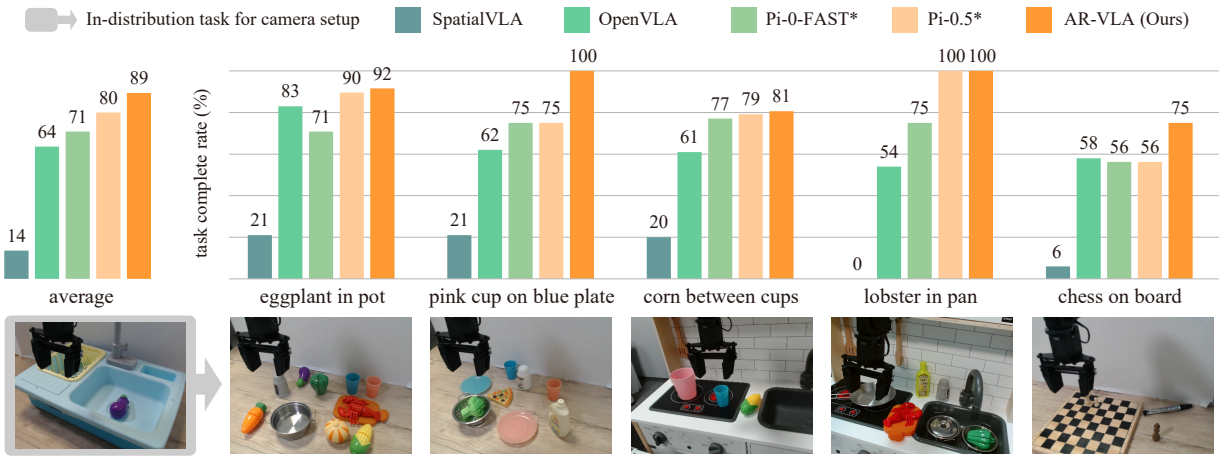


Fig. 6: **BridgeV2 pretraining to real-world WidowX Zero-Shot Performance Comparison.** We set the camera pose so that all methods, except SpatialVLA[32], reach a 100% success rate on an easy in-distribution task, then test them zero-shot on challenging tasks. Details of experiment protocol in Appendix.

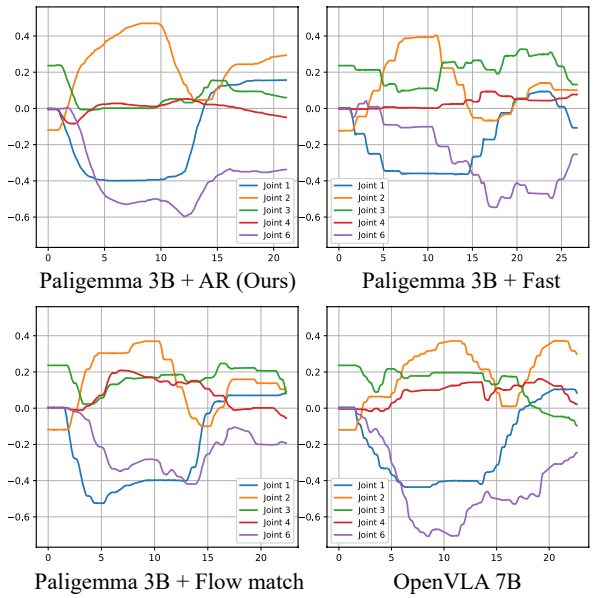


Fig. 7: **Smoothness Visualization.** Joint states captured from success execution for the same task. X-axis in seconds, y-axis in rad

achieves strong performance across all tasks, demonstrating its viability as a robust policy backbone for task-specific imitation learning. On ALOHA cube transfer, AR Actor reaches 97.33% scripted success and 67.33% human demonstration success, substantially outperforming ACT (86.0%/50.0%) and Diffusion Policy (33.33%/10.0%). On ALOHA peg insertion, AR Actor achieves 54.67% scripted success versus ACT's 32.0%. On PushT, while Diffusion Policy achieves the highest success rate of 65.20%, AR Actor reaches competitive 60.40% with 0.920 Max IoU. Notably, AR Actor maintains consistent performance across diverse task types, ranging from 2D pushing manipulation to bi-manual insertion tasks, while other methods show significant task-specific variance. Our AR policy is trained with a single objective: to maximize

TABLE III: **Inference Smoothness Metrics.** Jerk in 10^2 rad/s^3. VLM and Action Expert latencies in ms / chunk size. Total/Act represents the effective latency per single action.

Model	**Jerk (↓)**		**Latency (↓)**		
	Avg	Max	VLM	Act. Expert	Total/Act
OpenVLA	10.13	42.14	321.72 / 1		321.72
Fast	8.15	80.24	744.78 / 4		186.20
FM	9.39	45.33	69.77 / 4	267.28 / 4	84.26
AR (Ours)	**7.89**	**39.83**	69.56 / 4	**28.86 / 1**	**46.25**

the conditional likelihood of each action in a sequence. This suggests that architectural simplicity may be a key factor for consistent task-agnostic performance, whereas more complex methods like Diffusion Policy excel on specific tasks but struggle on others. Our observations align with similar results that have been reported by ARP [46].

B. Efficiency and Smoothness Analysis

We evaluate the systemic advantages of our asynchronous execution framework. By structurally decoupling the control thread from VLM inference, AR-VLA maintains a stable 29ms per action control frequency even when the perception backbone is capped at 70ms per frame. As shown qualitatively in and Fig. 7 quantitatively in Tab. III, AR-VLA achieves the lowest maximum and average jerk during task execution. This superiority arises from the model's ability to maintain rigorous spatial and temporal action consistency, alongside the lowest overall latency. In contrast, chunk-based models, when implemented naively, only guarantee intra-chunk smoothness and suffer from inter-chunk gaps and execution latency.

C. History-Awareness Evaluation

To validate that autoregressive action generation with stored history key-value caches enables temporal awareness for long-horizon tasks, we design two benchmark tasks requiring memory of past states. As shown in Fig. 8, *PushT2* (simulation) requires pushing a T-shaped block to two goal positions in any order, where information about which goal has been reached

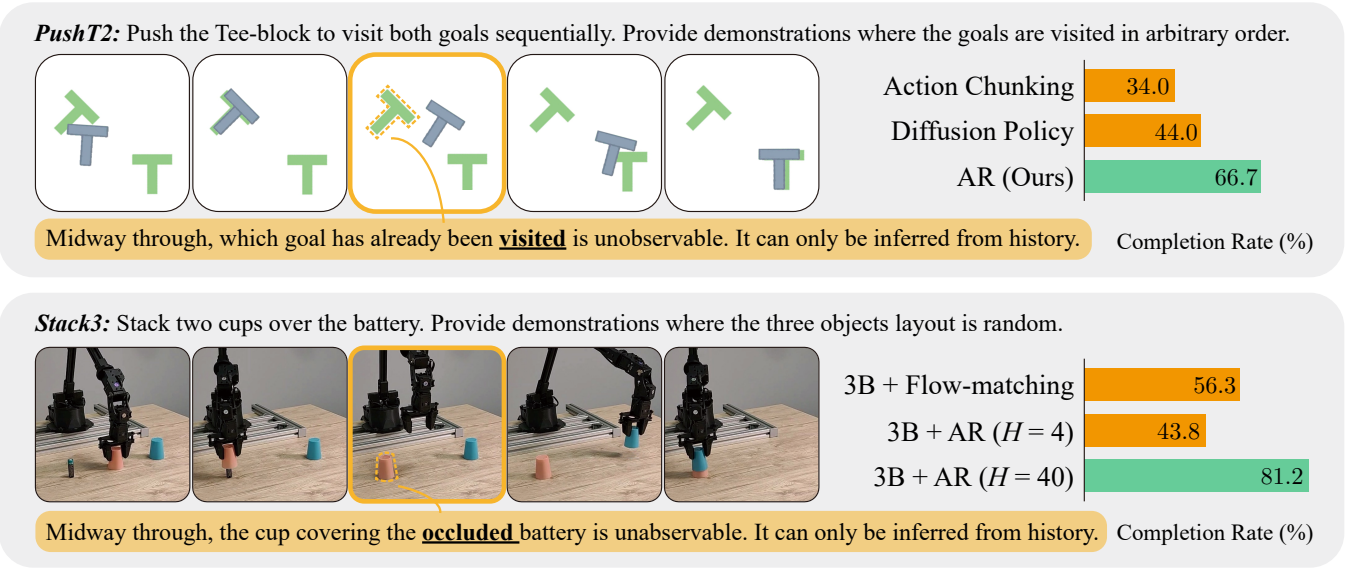


Fig. 8: **History-Awareness Evaluation.** PushT2 requires visiting both goals, but which goal has been visited is unobservable midway. Stack3 requires stacking cups over a battery that becomes occluded. Both task require memory of unobservable past states. H donates the context window length of AR-VLA. Details about task defination, data collection, training and execution in Appendix.

becomes unobservable once the block is halfway to the second goal. *Stack3* (real-world) requires covering a battery with one cup then stacking another cup on top, where the battery's location becomes unobservable once covered and can only be retrieved from action history. AR-VLA substantially outperforms existing policies on both tasks. Reactive policies exhibit "temporal amnesia," becoming trapped oscillating between sub-goals without maintaining context of completed steps. In contrast, AR-VLA's autoregressive generation naturally maintains history, momentum, and task intent through its action domain key-value cache, leading to superior success rates on these multi-stage manipulation sequences.

D. Ablations on Design Decisions

In this section, we conduct ablation studies to validate the contributions of our key design choices. Table IV presents the results across four critical components.

Causal Pretraining. We compare models with and without Phase 1 pretraining. Removing Phase 1 pretraining increases Phase 2 convergence time to 2× and still leads to inferior final performance, as the model must learn joint kinematics from scratch rather than building upon pretrained motion priors.

RoPE Re-anchoring. We ablate different positional encoding strategies for the action expert. Replacing our dynamic RoPE anchoring with fixed rotational embedding or non-rotational embedding is mathematically incorrect and results in poor inference performance. While eliminating positional embedding avoids training-inference attention score gaps, it cannot capture temporal step differences, resulting in a 52% performance drop. This validates that proper temporal position encoding is critical for autoregressive action generation.

Stochastic History Masking. We observe an intriguing causal confusion paradox across different masking rates. A 0.0 mask rate achieves the lowest validation error but 0% task success, indicating the model over-relies on its own history and fails during rollouts when predictions deviate. Our 0.6 mask rate provides the optimal balance between leveraging historical context and maintaining robustness to prediction errors.

Context Length. We vary the action KV cache length from 1 to 40 steps. The results show a clear upward trend in success rate as context length increases, directly validating that temporal awareness through extended history correlates with improved performance for autoregressive action experts.

TABLE IV: Ablation Study of AR-VLA Design Choice. Validate Error calculate from the model output and ground truth when taking dataset trajectory as inout. Success rates are percentages in total 96 SIMPLER trials after training. Bold indicates the full AR-VLA configuration.

Variant	Val. Error (↓)	Sim. Success (↑)
w/ Phase-1 **(Full)**	4.2	61.5
w/o Phase-1 (100% time)	4.5	37.5
w/o Phase-1 (200% time)	4.0	54.2
mask rate = 0.0	2.7	0.0
mask rate = 0.2	3.1	27.1
mask rate = 0.4	3.5	47.9
mask rate = 0.6 **(Full)**	4.2	61.5
mask rate = 0.8	5.7	49.0
mask rate = 1.0	2.9	28.1
w/ Static Pos. Embedding	3.0	3.1
w/o Pos. Embedding	2.9	29.2
history length = 1	6.0	36.5
history length = 5	5.2	50.0
history length = 10	4.7	59.4
history length = 20 **(Full)**	4.2	61.5
history length = 40	4.2	59.4

V. Conclusion and Discussion

We introduced a design of Autoregressive Action Expert, facilitating a structural shift from the prevailing paradigm

of reactive, snapshot-based control to continuous streaming sequences. By equipping the policy with a persistent causal history of its own, our approach resolves the temporal inconsistencies inherent in current VLA architectures and ensures action generation remains consistent across both spatial and temporal dimensions. Empirically, we demonstrate that this autoregressive structure yields significantly smoother trajectories and reduced execution latency compared to reactive baselines, with particular robustness in long-horizon tasks where maintaining historical context is critical. Beyond immediate performance gains, our architecture offers a scalable framework for embodied learning by structurally decoupling the syntax of motion from semantic perception. This allows for the independent pretraining of kinematic dynamics and the asynchronous integration of perception and control.

However, transitioning to this autoregressive, cache-based paradigm introduces new challenges. First, novel sequences of individually familiar states within the KV-cache can compound, pushing the policy into out-of-distribution (OOD) failures. Second, similar to flow-matching experts, direct AR action gradients degrade the VLM's pretrained semantic priors, necessitating a "knowledge insulation" [15] training strategy. Finally, while proprioceptive history is fully cached, visual processing remains restricted to standard snapshots. We detail these limitations and future directions in Appendix.

Acknowledgments

This research was partially funded by the Ministry of Education and Science of Bulgaria (support for INSAIT, part of the Bulgarian National Roadmap for Research Infrastructure), and supported by Interne Fondsen KU Leuven / Internal Funds KU Leuven (C2E/24/034).

References

[1] LeRobot: An Open-Source Library for End-to-End Robot Learning. In *The Fourteenth International Conference on Learning Representations*, October 2025.

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances, August 2022.

[3] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[4] 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.

[5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. \pi_0 : A vision-language-action flow model for general robot control. In *Robotics: Science and Systems*, 2025.

[6] Kevin Black, Manuel Y Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *arXiv preprint arXiv:2506.07339*, 2025.

[7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[9] 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*, page 02783649241273668, 2023.

[10] Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/

forum?id=c7rM7F7jQjN.

[11] Sombit Dey, Jan-Nico Zaech, Nikolay Nikolov, Luc Van Gool, and Danda Pani Paudel. Revla: Reverting visual domain limitation of robotic foundation models, 2024. URL https://arxiv.org/abs/2409.15250.

[12] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[13] Ankit Goyal, Valts Blukis, Jie Xu, Yijie Guo, Yu-Wei Chao, and Dieter Fox. Rvt2: Learning precise manipulation from few demonstrations. *RSS*, 2024.

[14] Matthew J. Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI Fall Symposia*, volume 45, page 141, 2015.

[15] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization, April 2025.

[16] Daniel Kahneman. Thinking, fast and slow. *Farrar, Straus and Giroux*, 2011.

[17] Kento Kawaharazuka, Jihoon Oh, Jun Yamada, Ingmar Posner, and Yuke Zhu. Vision-language-action models for robotics: A review towards real-world applications. *IEEE Access*, 13:162467–162504, 2025. doi: 10.1109/ACCESS.2025.3609980.

[18] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

[19] 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 *Conference on Robot Learning*, 2024.

[20] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems*, 2025.

[21] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. In *International Conference on Machine Learning*, 2024.

[22] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[23] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A Foundational Vision-Language-Action Model for Synergizing Cognition and Action in Robotic Manipulation, November 2024.

[24] 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.

[25] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating Real-World Robot Manipulation Policies in Simulation, May 2024.

[26] Yicheng Liu, Shiduo Zhang, Zibin Dong, Baijun Ye, Tianyuan Yuan, Xiaopeng Yu, Linqi Yin, Chenhao Lu, Junhao Shi, Luca Jiang-Tao Yu, Liangtao Zheng, Tao Jiang, Jingjing Gong, Xipeng Qiu, and Hang Zhao. FASTer: Toward Efficient Autoregressive Vision Language Action Modeling via Neural Action Tokenization, December 2025.

[27] Huaihai Lyu, Chaofan Chen, Senwei Xie, Pengwei Wang, Xiansheng Chen, Shanghang Zhang, and Changsheng Xu. Omnisat: Compact action token, faster auto regression, 2025. URL https://arxiv.org/abs/2510.09667.

[28] Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, Jinwoo Shin. HAMLET: Switch Your Vision-Language-Action Model into a History-Aware Policy. In *The Fourteenth International Conference on Learning Representations*, October 2025.

[29] Octo Model Team, Dibya Ghosh, Homer Walke bit, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Conference on Robot Learning*, 2024.

[30] Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, pages 7487–7498. PMLR, 2020.

[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, and Xuelong Li. SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Model, May 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-flow models. In Joseph Lim, Shuran Song, and Hae-Won Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 3736–3761. PMLR, 27–30 Sep 2025.

[34] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems*, volume 35, pages 22955–22968, 2022.

[35] Hao Shi, Bin Xie, Yingfei Liu, Lin Sun bit, 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.

[36] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[37] Haoming Song, Delin Qu, Yuanqi Yao, Qizhi Chen, Qi Lv, Yiwen Tang, Modi Shi, Guanghui Ren, Maoqing Yao, Bin Zhao, et al. Hume: Introducing system-2 thinking in visual-language-action model. *arXiv preprint arXiv:2505.21432*, 2025.

[38] Alexander Spiridonov, Jan-Nico Zaech, Nikolay Nikolov, Luc Van Gool, and Danda Pani Paudel. Generalist robot manipulation beyond action labeled data. In *9th Annual Conference on Robot Learning*, 2025. URL https://openreview.net/forum?id=ZqBXnR6ppz.

[39] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

[40] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in Neural Information Processing Systems*, 28, 2015.

[41] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy, May 2024.

[42] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A Dataset for Robot Learning at Scale, January 2024.

[43] Shuai Yang, Hao Li, Yilun Chen, Bin Wang, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation, 2025. URL https://arxiv.org/abs/2507.17520.

[44] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VYOe2eBQeh.

[45] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *8th Annual Conference on Robot Learning*, 2024.

[46] Xinyu Zhang, Yuhan Liu, Haonan Chang, Liam Schramm, and Abdeslam Boularias. Autoregressive action sequence learning for robotic manipulation, 2025. URL https://arxiv.org/abs/2410.03132.

[47] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware, April 2023.

[48] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

[49] Hongyi Zhou, Weiran Liao, Xi Huang, Yucheng Tang, Fabian Otto, Xiaogang Jia, Xinkai Jiang, Simon Hilber, Ge Li, Qian Wang, Ömer Erdinç Yağmurlu, Nils Blank, Moritz Reuss, and Rudolf Lioutikov. BEAST: Efficient tokenization of b-splines encoded action sequences for imitation learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[50] 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*, pages 2165–2183. PMLR, 2023.