WA-SPECDEC: WORLD-AWARE SPECULATIVE DECODING FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) policies generate robot controls autoregressively, making closed-loop latency dominated by repeated target model forward passes. Speculative decoding reduces this cost by verifying blocks of draft action tokens in parallel, and recent VLA methods relax token-level acceptance because small differences in action token space often map to similar continuous controls. However, a fixed token-distance tolerance does not account for the scene-dependent consequences of action deviations: deviations that are harmless in free space can cause collisions or grasp failures near contact. We propose WA-SpecDec, a world-aware speculative decoding framework that injects world-model-derived physical scene information during the VLA prefill stage. The resulting world-aware prefill states are shared between draft generation and target verification, while the relaxed acceptance rule remains unchanged. Evaluated across four VLA policies and three relaxed acceptance rules on LIBERO, SIMPLER-Env, and four real-robot tasks, WA-SpecDec improves the trade-off between inference speed and task success. Across five equally weighted policy–benchmark settings, it increases task success by an average of 5.3 percentage points over the speculative baselines and achieves an average inference speedup of $1.94\times$ over autoregressive decoding of the base policies. For the two policies evaluated on LIBERO, it reduces the near-contact failure (NCF) metric by 19.8% on average relative to the speculative baselines.

1 INTRODUCTION

Vision-language-action (VLA) models extend language-conditioned robot control and embodied multimodal modeling into executable robot policies (Ichter et al., 2023; Driess et al., 2023). Modern robot transformers map language-conditioned visual observations to action tokens or continuous commands that are decoded into robot control signals (Brohan et al., 2023; Zitkovich et al., 2023). Cross-embodiment datasets and open-source generalist policies further make this action-generation interface a practical foundation for embodied control (O'Neill et al., 2024; Kim et al., 2025). However, token-generating VLA policies still rely on repeated target model forward passes during autoregressive (AR) action decoding, which makes closed-loop latency a key bottleneck for fast observe–decide–act cycles.

Speculative decoding reduces this cost by using a lightweight draft model to propose multiple future tokens that are verified in parallel by the target model (Leviathan et al., 2023). Subsequent speculative sampling and feature-aware drafting methods further improve this acceleration paradigm for autoregressive decoding (Chen et al., 2023; Li et al., 2024). In VLA decoding, exact token agreement is often too restrictive because small differences in action token space can map to similar continuous controls. Recent VLA speculative decoding methods therefore adopt *relaxed acceptance*, accepting a draft token when it lies within a tolerance ρ of the target token in action token space (Wang et al., 2025). Kinematic and hybrid variants further adapt this idea to embodied action spaces (Zheng et al., 2026a;b). These methods improve accepted length and wall-clock speed, but they also shift the key reliability question from exact token matching to whether an accepted token deviation is physically safe in the current scene.

Whether an accepted action token deviation is safe is inherently scene-dependent. The same deviation can be benign during free-space motion, but near contact it may change whether the robot grasps, collides, misses the target, or enters the wrong contact phase. This motivates examining

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.

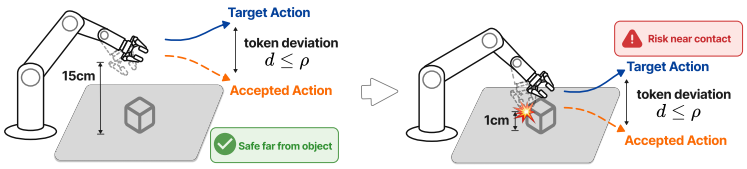


Figure 1: Motivation for world-aware speculative decoding. The same draft–target deviation can be safe in free space but risky near contact.

the representation used by relaxed speculative decoding. Before action decoding, standard VLA policies construct a multimodal prefill input from language tokens and projected visual patch embeddings, which is then encoded into hidden states for autoregressive action generation. Although image–text and self-supervised pretraining provide strong semantic and spatial representations (Zhai et al., 2023; Oquab et al., 2024), these visual features are not explicitly trained to preserve predictive physical state such as object pose evolution, gripper–object distance, or contact phase (Nair et al., 2023; Radosavovic et al., 2023). As a result, relaxed verification applies a global token-distance test while the physical cost of an accepted deviation varies across scenes. Figure 1 illustrates this mismatch. A tolerated deviation may be safe when the gripper is far from the object, but unsafe near contact. A scene-agnostic relaxed decoder must therefore keep ρ conservative to preserve task success, which limits accepted length and speedup. Looser relaxation improves acceptance, but can admit physically inconsistent actions.

To address this mismatch, we propose *World-Aware Speculative Decoding for Vision-Language-Action Models* (WA-SpecDec), a framework that introduces world-model-derived scene representations into VLA speculative decoding (Wu et al., 2023). WA-SpecDec injects this awareness before prefill computation, producing world-aware prefill hidden states shared between draft generation and target verification, rather than modifying the relaxed acceptance rule. Specifically, WA-SpecDec enriches visual patch embeddings with a World-Aware Bias module, which maps current-observation world tokens into spatially aligned biases and is trained with action prediction together with an auxiliary next-frame latent objective. At inference, it uses only the current observation and performs no future-frame prediction. By conditioning both proposal and verification on the same physically informed prefill hidden states, WA-SpecDec improves relaxed speculative decoding while preserving the original VLA action token interface. Our contributions are threefold:

- We identify a scene-dependent reliability bottleneck in relaxed VLA speculative decoding, where a fixed token-distance tolerance can accept deviations whose physical cost varies between free-space and near-contact states.
- We propose WA-SpecDec, a framework that incorporates world-model-derived physical scene awareness into VLA speculative decoding by injecting a spatially aligned World-Aware Bias before prefill computation, producing shared world-aware prefill hidden states while leaving the relaxed acceptance rule unchanged.
- We introduce near-contact failure (NCF) to quantify failures that enter task-relevant near-contact regions, and evaluate WA-SpecDec across four VLA policies, three relaxed acceptance rules, two simulation benchmarks, and four real-robot tasks. WA-SpecDec achieves a 5.3-point average success gain over Standard SpecDec and a $1.94\times$ inference speedup over the corresponding base-policy AR references. On LIBERO, it reduces NCF by 19.8% on average relative to the corresponding speculative baselines.

2 Background and Related Work

Acceleration for VLA Models. Language-conditioned robot control first connected pretrained language models to robotic affordances (Ichter et al., 2023) and embodied multimodal reasoning (Driess et al., 2023). Robot transformers then showed that large-scale sequence policies can map language-conditioned visual observations to executable robot actions across diverse manipulation tasks. More recent VLA and generalist robot policies extend this direction with cross-embodiment data (O'Neill et al., 2024), open-source action token policies (Ghosh et al., 2024; Pertsch et al., 2025), and action chunk or flow-based control heads (Black et al., 2025). These systems improve broad task transfer

and generalization, but token-generating VLA policies still inherit the latency profile of autoregressive decoding (Zitkovich et al., 2023; Kim et al., 2025).

At each control step t, a VLA policy receives a language instruction x and visual observation o_t, and predicts discrete action tokens $a_{t,1:K}$ that are detokenized into continuous robot control. A single target decoding step is

$$p_\theta(a_i \mid x, o_t, a_{<i}), \qquad i = 1, \ldots, K, \tag{1}$$

where θ denotes the target VLA, K is the number of action tokens per control query, and $a_{<i}$ are previous action tokens. The observation is encoded into visual patch embeddings and combined with text tokens to form the prefill input. Because closed-loop control repeatedly invokes this decoder over multi-token action horizons, latency scales with the number of target decoding rounds. Acceleration therefore requires reducing target forward passes while preserving the VLA action token interface and task success (Yang et al., 2026).

Speculative Decoding and Relaxed Verification. Speculative decoding accelerates autoregressive generation by using a lightweight draft model to propose several future tokens and a larger target model to verify them in parallel (Leviathan et al., 2023; Chen et al., 2023). Follow-up work improves this paradigm through Big Little Decoder (BiLD) (Kim et al., 2023), tree-based verification (Miao et al., 2024), auxiliary decoding heads (Cai et al., 2024), retrieval (He et al., 2024), feature-level drafting (Li et al., 2024), or self-speculation (Zhang et al., 2024). These methods reduce the number of target decoding rounds by accepting multiple draft tokens after one target verification pass.

For VLA policies, exact token matching is often overly restrictive because nearby action tokens can detokenize to similar continuous controls. Recent VLA speculative decoding methods therefore introduce relaxed verification (Wang et al., 2025). A basic relaxed acceptance rule is

$$\mathrm{Accept}_\rho(\hat{a}_i) = \mathbf{1}[d(\hat{a}_i, a_i^\star) \le \rho], \tag{2}$$

where $\mathbf{1}[\cdot]$ is the indicator function, $d(\cdot, \cdot)$ measures token-space or detokenized action-space distance, $a_i^\star$ is the target token, and ρ is the tolerance. Larger ρ accepts longer draft prefixes and improves speed, while smaller ρ keeps the executed sequence closer to the target decoder.

KERV combines Kalman-filter action compensation with kinematics-based threshold adaptation (Zheng et al., 2026a). HeiSD combines kinematic routing between retrieved and learned drafts with feature-based verification skipping and sequence-wise relaxed acceptance (Zheng et al., 2026b). Both exploit visual information, but kinematic summaries alone do not determine object-relative contact risk: identical robot motion can have different consequences when object placement changes. WA-SpecDec enriches the shared scene representation for drafting and verification through action prediction and auxiliary next-frame supervision, while retaining the selected acceptance rule.

World Representations for Scene-Aware Decoding. The reliability of relaxed VLA verification depends not only on the acceptance rule, but also on the scene representation encoded into the shared prefill hidden states. Modern visual backbones provide strong semantic and spatial features (Oquab et al., 2024; Zhai et al., 2023), yet their pretraining objectives are not designed to preserve control-critical variables such as object pose, gripper–object distance, contact phase, and occlusion (Majumdar et al., 2023). Since token-based VLA policies use visual patch embeddings to form the multimodal prefill input for both drafting and verification, missing physical detail can affect the target reference and draft proposal together (Karamcheti et al., 2023).

World representations provide a complementary source of predictive physical structure. VAEs and latent world models learn compact representations through reconstruction and future prediction (Kingma & Welling, 2014; Ha & Schmidhuber, 2018). Latent dynamics models support predictive control from pixels (Hafner et al., 2019; Hansen et al., 2024), while recent video-predictive representation learning shows that latent compressors and feature-prediction objectives retain cues useful for physical change (Bardes et al., 2024; Team et al., 2025). WA-SpecDec uses this complementarity to make relaxed speculative decoding more reliable. Instead of adding a post-hoc acceptor, changing ρ, or building a larger draft model, it injects a world-model-derived physical scene bias into visual patch embeddings before prefill computation. The resulting world-aware prefill hidden states are shared between draft generation and target verification, preserving the original VLA action token interface and relaxed acceptance rule while giving both sides of speculative decoding access to the same predictive scene signal.

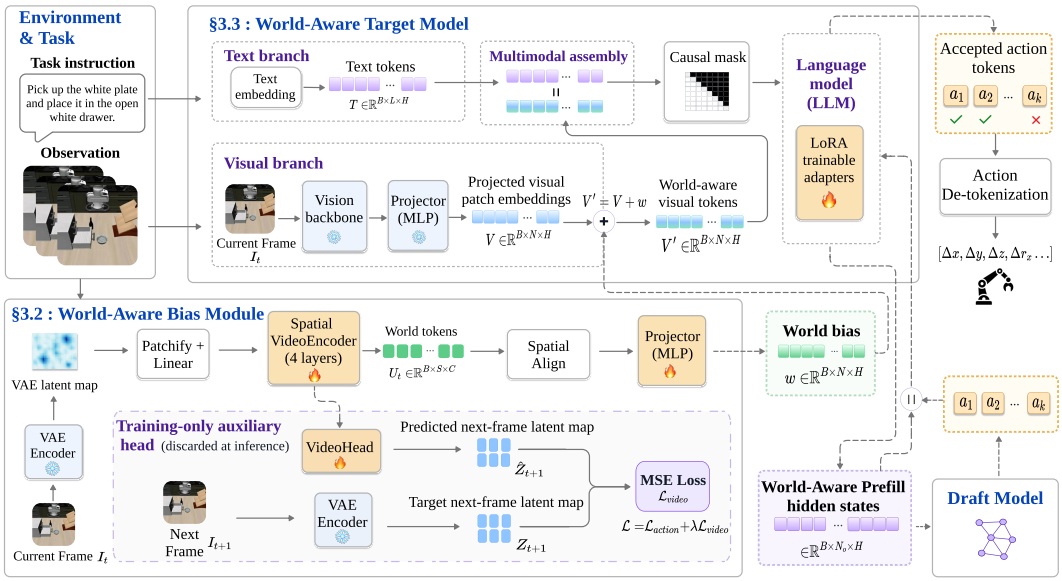


Figure 2: The overall WA-SpecDec architecture. World-aware bias module adds a spatially aligned world bias w_t to visual patch embeddings before prefill computation, yielding world-aware prefill hidden states shared between draft generation and target verification. The auxiliary next-frame prediction head is used only during training.

3 WA-SpecDec Framework

3.1 Overview

WA-SpecDec injects physical-world awareness into relaxed VLA speculative decoding by enriching visual inputs while keeping the relaxed acceptance rule unchanged. As shown in Figure 2, at each control step, a World-Aware Bias (WAB) module maps the current observation I_t to a spatially aligned bias w_t, which is added to visual patch embeddings before prefill. Together with the language instruction x, these augmented embeddings produce world-aware prefill states shared between draft generation and target verification, preserving the target-side sequence length and action token interface. A lightweight draft model then proposes action tokens, and the target VLA verifies them in parallel. Accepted prefixes are retained across speculative rounds, and the completed action token sequence is detokenized into continuous robot commands.

WAB and a target-policy LoRA adapter are jointly trained with action prediction and auxiliary next-frame latent supervision. After adaptation, the target is frozen and the draft is distilled from it. This adaptation changes the target reference used for relaxed verification, while the draft learns to propose tokens from the same world-aware prefill hidden states. At inference, the auxiliary prediction head is removed, and WAB computes the bias once per observation using only the current frame. We evaluate the resulting acceptance–success frontier and use NCF to characterize changes in failure composition. Section 3.2 describes WAB, and Section 3.3 details target adaptation, draft distillation, and verification.

3.2 World-Aware Bias Module

The World-Aware Bias (WAB) module provides the physical scene signal used to enrich the visual inputs that produce VLA prefill hidden states. Given the current RGB observation I_t, it produces a spatially aligned bias w_t that has the same shape as the VLA visual patch embeddings. Rather than appending extra world tokens to the multimodal sequence, WAB modulates existing visual tokens before prefill computation, allowing physical information to affect prefill hidden states without

increasing the target-side sequence length. WAB starts from a frozen VAE latent space:

$$Z_t = E_{\text{vae}}(I_t) \in \mathbb{R}^{B\times H_z\times W_z\times C_{\text{vae}}}. \tag{3}$$

The encoder E_{vae} maps I_t to a latent scene map Z_t, where B is the batch size, H_z and W_z are the latent spatial resolution, and C_{vae} is the VAE channel dimension. We keep this encoder frozen so that WAB learns only a lightweight adaptation path on top of a stable reconstructive representation. This latent representation is complementary to the semantic vision backbone because it retains spatial appearance and layout details that are useful for contact-sensitive control. The latent map is then adapted into world tokens:

$$U_t = F_\omega(\phi_{\text{tok}}(Z_t)) \in \mathbb{R}^{B\times S\times C}, \qquad S = H_z W_z. \tag{4}$$

The tokenization map ϕ_{tok} flattens the VAE spatial grid and applies a linear projection to produce S latent tokens of width C. The spatial VideoEncoder F_ω then refines these tokens into U_t. This step keeps the VAE latent grid as the source of spatial structure, while allowing the trainable WAB path to emphasize action-relevant factors such as object pose, occlusion, and local gripper–object geometry. To make these world tokens usable by the VLA visual branch, WAB aligns them to the visual patch grid and projects them to the VLA hidden width:

$$w_t = f_{\text{bias}}(\text{SpatialAlign}(U_t)) \in \mathbb{R}^{B\times N_v\times H}. \tag{5}$$

Here, N_v is the number of VLA visual patches and H is the hidden width of the projected visual embeddings. $\text{SpatialAlign}(\cdot)$ reshapes U_t to its two-dimensional latent grid, interpolates it to the visual patch resolution, and flattens it back to a token sequence. The projector f_{bias} maps each aligned world token to the VLA hidden width. As a result, w_t can be added directly to the visual patch embeddings $V_t \in \mathbb{R}^{B\times N_v\times H}$, giving each patch a location-specific physical scene bias.

WAB is trained not only through action prediction but also through a dense next-frame latent prediction objective. From the current-frame world tokens, a training-only VideoHead g_η predicts the next-frame VAE latent map:

$$\mathcal{L}_{\text{video}} = \frac{1}{H_z W_z C_{\text{vae}}} \left\| g_\eta(U_t) - E_{\text{vae}}(I_{t+1}) \right\|_2^2. \tag{6}$$

The next frame I_{t+1} is used only during training to form the target latent map, and η denotes the VideoHead parameters. Because the loss is computed over the full latent grid rather than a pooled representation, it encourages U_t to preserve spatially resolved cues predictive of short-term scene change. This supervision is particularly relevant for relaxed VLA decoding, where small token deviations can become unsafe near contact.

The overall training loss combines the action token objective with this auxiliary latent prediction term:

$$\mathcal{L} = \mathcal{L}_{\text{action}} + \lambda \mathcal{L}_{\text{video}}. \tag{7}$$

The scalar λ controls the strength of the auxiliary objective. The VideoHead is discarded after training, so inference uses only the current observation I_t and does not perform future-frame prediction. The learned WAB path therefore adds only the current-frame bias w_t at test time. This bias is later encoded into the world-aware prefill hidden states shared by both the target and draft, keeping proposal and verification conditioned on the same spatially grounded scene signal.

3.3 World-Aware Target Model

Figure 2 shows how the target model becomes world-aware before any action token is decoded. Let the frozen visual backbone and multimodal projector produce visual patch embeddings $V_t \in \mathbb{R}^{B\times N_v\times H}$. WAB produces the spatially aligned bias $w_t \in \mathbb{R}^{B\times N_v\times H}$ from Equation 5, which matches the shape of V_t, and the target injects it as a residual world-conditioned modulation.

$$V_t^{\text{wa}} = V_t + w_t. \tag{8}$$

The addition is element-wise across visual patches, so each patch receives its own location-specific bias. Although the operation is additive, w_t is not a free parameter because it is computed from VAE latent tokens that retain reconstructive scene detail and is shaped by the dense next-frame latent objective in Equation 6. The added direction is therefore trained to summarize scene factors predictive

of short-term physical change, such as pose, occlusion, and contact configuration, rather than only image–text semantics. Injecting it before prefill computation makes this physical scene direction participate in the transformer's query, key, value, and feed-forward projections, so it becomes part of the hidden states and key–value cache used to produce subsequent action token logits. The world-aware visual tokens are concatenated with text embeddings T_x and processed by the LoRA-adapted target:

$$P_t^{\mathrm{wa}} = [T_x; V_t^{\mathrm{wa}}], \qquad h_t^{\mathrm{wa}} = \mathrm{Prefill}_{\theta+\Delta\theta}(P_t^{\mathrm{wa}}), \tag{9}$$

where θ denotes the pretrained VLA parameters and $\Delta\theta$ denotes the trainable LoRA adapter. This placement is central because the world representation is not used as a post-hoc acceptor, but changes the target reference token itself. Because w_t is added to existing visual tokens rather than appended as extra world tokens, P_t^{wa} has the same sequence length as the original multimodal prefill input. The WAB path adds computation, but the target processes the same number of prefill tokens.

The action objective in Equation 7 trains this world-aware target to predict demonstration action tokens. For a training set $\mathcal{D}$ of instruction, observation, and action token sequences, we use

$$\mathcal{L}_{\mathrm{action}} = -\sum_{(x, I_t, a^\star_{1:K}) \in \mathcal{D}} \sum_{i=1}^{K} \log p_\theta^{\mathrm{wa}}(a_i^\star \mid x, I_t, a^\star_{<i}), \tag{10}$$

with

$$p_\theta^{\mathrm{wa}}(a_i \mid x, I_t, a_{<i}) \triangleq p_{\theta+\Delta\theta}(a_i \mid h_t^{\mathrm{wa}}, a_{<i}). \tag{11}$$

Together with $\mathcal{L}_{\mathrm{video}}$, this objective updates WAB and $\Delta\theta$ while keeping the VAE and base visual stack frozen. The VideoHead affects the target only through training-time gradients into WAB. This stage defines the target policy that WA-SpecDec accelerates. Relaxed verification uses this world-aware target as its reference and does not guarantee exact preservation of its output distribution. At inference, relaxed verification is unchanged except for the reference distribution. Given a draft token $\hat{a}_i$, the target produces

$$a_i^{\mathrm{wa}} = \arg\max_a p_\theta^{\mathrm{wa}}(a \mid x, I_t, a_{<i}), \tag{12}$$

and applies the same distance test as Equation 2,

$$\mathrm{Accept}_\rho^{\mathrm{wa}}(\hat{a}_i) = \mathbf{1}[d(\hat{a}_i, a_i^{\mathrm{wa}}) \le \rho]. \tag{13}$$

WA-SpecDec therefore changes the reference used by the relaxed acceptance rule. Any relaxed acceptance rule that compares draft proposals with target references can use the world-aware reference. This is where acceleration enters. Because the target and draft share the same physical scene-aware prefill hidden states, the draft is encouraged to match a better-conditioned reference, yielding longer accepted prefixes at fixed reliability and fewer target verification rounds. Since w_t adds no tokens, this gain is not offset by extra target-side attention cost.

The same world-aware prefill hidden states are used for drafting. After training the world-aware target, we freeze it and distill a lightweight draft model q_ϕ against its output distribution,

$$\mathcal{L}_{\mathrm{draft}} = \mathbb{E}_{(x, I_t, a_{<i})}\Big[\mathrm{KL}\big(p_\theta^{\mathrm{wa}}(\cdot \mid x, I_t, a_{<i}) \,\|\, q_\phi(\cdot \mid h_t^{\mathrm{wa}}, a_{<i})\big)\Big]. \tag{14}$$

Sharing h_t^{wa} conditions draft proposals and the target reference on the same physical scene information, rather than strengthening the draft in isolation. At inference, WAB computes w_t once per observation. The draft model proposes γ tokens from h_t^{wa} for parallel target verification using Equation 13. Each round retains the accepted prefix and replaces the first rejected draft token with the corresponding target token. Algorithm 1 in Appendix A summarizes training and inference.

4 Experiments

4.1 Experimental Setup

Benchmarks and target policies. We use fine-tuned OpenVLA (Kim et al., 2025) on the four LIBERO suites (Liu et al., 2023), with RynnVLA-002-Discrete (Cen et al., 2025) providing a stronger target on the same benchmark. OpenVLA and RLVLA-RL (Liu et al., 2025) extend the evaluation to SIMPLER-Env (Li et al., 2025) (see Appendix E.1). Physical experiments use MolmoAct (Lee et al., 2025) on four LeRobot SO-ARM101 tasks (Cadene et al., 2024). Figure 3 illustrates the evaluation tasks. Training and evaluation details are provided in Appendices B and C.

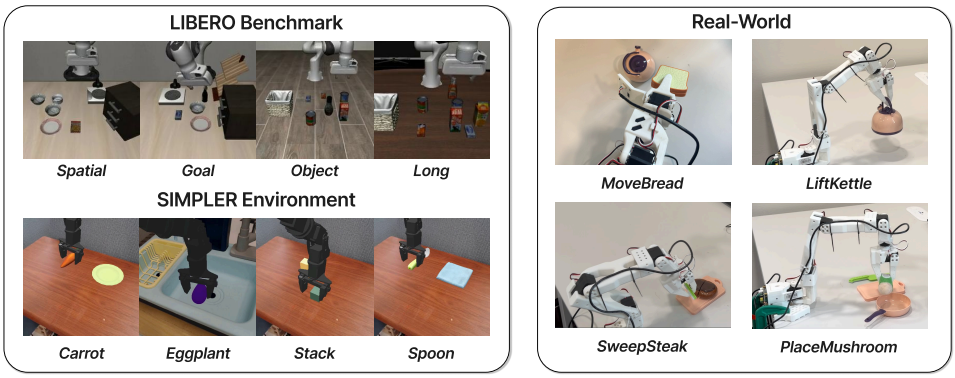


Figure 3: Evaluation tasks in simulation and on the physical robot. Left: LIBERO and SIMPLER-Env tasks. Right: manipulation tasks on the LeRobot SO-ARM101 platform.

Table 1: Main comparison on four LIBERO suites using OpenVLA, with $\rho = 10$ and default settings for KERV- and HeiSD-derived acceptance rules. SpecDec denotes Standard SpecDec. Succ. and NCF are percentages, with NCF defined in Section 4.1. Spd. is wall-clock speedup over the AR target, including WAB computation for WA-SpecDec.

Decoding Method		Spatial			Object			Goal			Long		
Rule	Method	Succ.	Spd.	NCF	Succ.	Spd.	NCF	Succ.	Spd.	NCF	Succ.	Spd.	NCF
N/A	OpenVLA (AR)	82.8	1.00×	73.3	69.8	1.00×	49.7	78.2	1.00×	46.8	57.8	1.00×	45.5
ρ	SpecDec	79.4	1.26×	78.6	63.0	1.13×	60.0	69.6	1.23×	64.5	49.4	1.18×	52.2
ρ	WA-SpecDec	**83.0**	**1.54×**	**72.9**	**68.8**	**1.24×**	**48.7**	**75.8**	**1.32×**	**46.3**	**56.2**	**1.21×**	**44.7**
KERV	SpecDec	80.2	1.52×	75.8	63.4	1.54×	63.9	70.8	1.61×	61.6	48.6	1.41×	54.1
KERV	WA-SpecDec	**83.4**	**1.59×**	**72.3**	**68.4**	**1.64×**	**48.1**	**77.6**	**1.67×**	**47.3**	**52.8**	**1.45×**	**42.8**
HeiSD	SpecDec	77.4	1.74×	69.0	68.0	2.26×	63.8	70.8	2.10×	66.4	50.4	1.71×	51.6
HeiSD	WA-SpecDec	**80.4**	**1.77×**	**65.3**	**70.6**	**2.31×**	**49.0**	**78.0**	**2.12×**	**43.6**	**57.0**	**1.79×**	**42.3**

Comparison protocol. Standard SpecDec uses Spec-VLA (Wang et al., 2025) without world-aware conditioning. We compare it with WA-SpecDec under three relaxed acceptance rules in this common implementation:Spec-VLA 's ρ-based token-distance rule and those adopted from KERV (Zheng et al., 2026a) and HeiSD (Zheng et al., 2026b). Both methods start from the same base policy fine-tuned with LoRA and use matched decoding settings and acceptance hyperparameters. Autoregressive decoding of this base policy serves as the common reference (AR target). The OpenVLA/LIBERO ablation also evaluates the world-aware target autoregressively to distinguish adaptation gains from changes in the AR-to-speculative success gap (Section 4.3).

Metrics. We report task success rate, accepted length (the mean number of draft action tokens accepted per target verification step), and wall-clock speedup relative to the AR target. Timing measurements use the same hardware and include draft proposal, target verification, WAB computation, and cache updates. On LIBERO, near-contact failure (NCF) is the percentage of failed episodes in which the gripper fingertips come within δ of either the manipulated object or the target receptacle at any point in the trajectory. We set $\delta = 2\,\mathrm{cm}$, approximately the width of one gripper finger. Appendix E.4 examines sensitivity to δ. Metric computation, including aggregation across evaluation settings, is detailed in Appendix B.5.

4.2 Task Success and Decoding Efficiency

Main results. Table 1 compares Standard SpecDec and WA-SpecDec under identical relaxed verifier families, including ρ, KERV, and HeiSD. Across all verifier families, WA-SpecDec stays closer to the AR target while retaining net wall-clock acceleration. Under the ρ verifier, WA-SpecDec im-

Table 2: Comparison using the stronger RynnVLA-002-Discrete policy, averaged over four LIBERO suites. Succ. and NCF are percentages. Spd. is wall-clock speedup over base-policy AR.

RynnVLA (AR) reference: 91.9% success, 1.00× speedup, 77.9% NCF									
Method	ρ			**KERV**			**HeiSD**		
	Succ.↑	**Spd.↑**	**NCF↓**	**Succ.↑**	**Spd.↑**	**NCF↓**	**Succ.↑**	**Spd.↑**	**NCF↓**
Standard SpecDec	89.5	1.26×	83.4	90.6	1.65×	82.0	90.3	2.11×	84.5
WA-SpecDec	93.7	1.38×	65.9	94.0	1.79×	69.2	94.0	2.78×	61.2

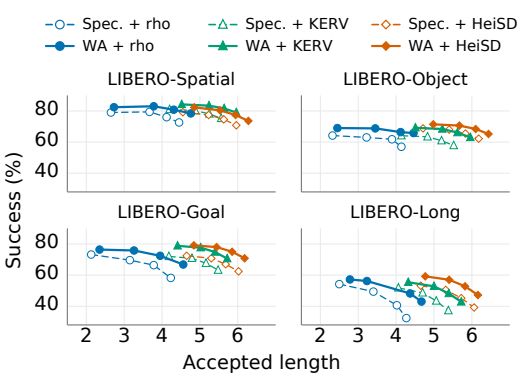


Figure 4: Task success versus accepted length for OpenVLA on four LIBERO suites under the ρ-based, KERV, and HeiSD acceptance rules. Each curve sweeps relaxation settings within a fixed rule. Solid lines denote WA-SpecDec and dashed lines denote Standard SpecDec.

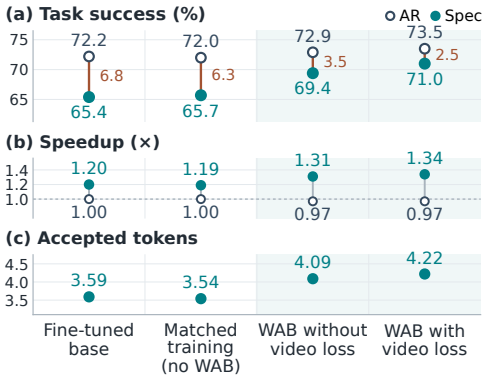


Figure 5: OpenVLA ablations averaged over four LIBERO suites ($\rho = 10$). (a) Success (%) and AR-to-Spec loss (percentage points). (b) Speedup over base-policy AR. (c) Mean accepted draft tokens per verification step. Open and filled markers denote AR and speculative decoding.

proves success over Standard SpecDec on Object, Goal, and Long, and remains comparable on Spatial, while increasing speedup from 1.26× to 1.54× on Spatial and from 1.23× to 1.32× on Goal. WA-SpecDec also maintains higher mean success than Standard SpecDec across different relaxation tolerances (Appendix E.3). The same trend holds with stronger verifiers: with HeiSD, WA-SpecDec improves success from 70.8 to 78.0 on Goal and from 50.4 to 57.0 on Long, with comparable or higher speedup. These results indicate that world-aware conditioning is complementary to existing relaxed verification rules rather than replacing them. The NCF results further support the contact-sensitive motivation. Averaged over the three acceptance rules, WA-SpecDec reduces suite-averaged NCF by 18.2% relative to Standard SpecDec (see Appendix E.2 for the corresponding NCF episode counts). Since NCF counts failures after the policy has entered the near-contact region, this reduction suggests that WA-SpecDec mainly improves reliability in states where small accepted action deviations are most likely to cause collision, missed grasp, or placement failure. Appendix D further illustrates WA-SpecDec's advantage in near-contact execution through a drawer-opening case study.

Stronger target policy. Table 2 reports results with RynnVLA-002-Discrete, a stronger target policy with an AR baseline success rate of 91.9% on LIBERO. Across the three acceptance rules, WA-SpecDec achieves 93.7%–94.0% mean success, compared with 89.5%–90.6% for Standard SpecDec. Under HeiSD, WA-SpecDec increases speedup relative to the AR baseline from 2.11× to 2.78× and reduces NCF from 84.5% to 61.2%. These gains show that WA-SpecDec remains effective beyond OpenVLA, extending its benefits to a stronger target policy.

Tradeoff between accepted length and task success. Figure 4 shows the acceptance–success frontier across a wider tolerance sweep. Within each relaxed verifier family, WA-SpecDec curves shift upward and rightward relative to the corresponding Standard SpecDec curves. The upward shift indicates higher task success at comparable accepted length, while the rightward shift indicates longer

Table 3: Real-robot comparison of Standard SpecDec (SpecDec) and WA-SpecDec using MolmoAct across four tasks, with 50 trials per task and decoding configuration. Succ. is task success rate (%). Spd. measures wall-clock speedup in action generation relative to autoregressive MolmoAct.

Decoding method		MoveBread		LiftKettle		SweepSteak		PlaceMushroom	
Rule	**Method**	**Succ.↑**	**Spd.↑**	**Succ.↑**	**Spd.↑**	**Succ.↑**	**Spd.↑**	**Succ.↑**	**Spd.↑**
N/A	MolmoAct (AR)	82	1.00×	92	1.00×	74	1.00×	90	1.00×
ρ	SpecDec	72	1.44×	94	1.35×	64	1.50×	90	1.48×
	WA-SpecDec	86	1.61×	94	1.56×	82	1.74×	92	1.66×
KERV	SpecDec	78	1.73×	88	1.77×	68	1.69×	76	1.88×
	WA-SpecDec	88	1.92×	**96**	1.89×	80	1.78×	90	2.01×
HeiSD	SpecDec	82	2.56×	90	2.61×	78	2.30×	90	2.21×
	WA-SpecDec	**90**	**2.88×**	94	**2.95×**	**86**	**2.83×**	**94**	**2.44×**

accepted prefixes at comparable success. Because longer accepted prefixes reduce the number of target verification rounds, this frontier shift shows that WA-SpecDec improves the trade-off between aggressive acceptance and reliable task execution, supporting more effective acceleration rather than simply accepting riskier draft tokens.

4.3 Ablation Study

Figure 5(a) reports component ablations using OpenVLA on LIBERO ($\rho = 10$). Matched continued training without WAB improves speculative success by only 0.3 percentage points, suggesting that additional training alone does little to mitigate the drop in VLA task success under relaxed speculative decoding. By contrast, WAB improves success even without next-frame supervision, and combining both components yields a 5.6-point gain over the base policy. Next-frame supervision further improves the success–speed trade-off by raising task success and narrowing the AR-to-Spec gap from 3.5 to 2.5 percentage points. As Figure 5(b,c) shows, it also increases mean accepted length from 4.09 to 4.22 tokens per verification step and wall-clock speedup from $1.31\times$ to $1.34\times$. The inference overhead of world-aware conditioning remains small, with both WAB variants retaining $0.97\times$ the base policy's AR decoding speed. Together, these ablations indicate that WAB provides the main gain in speculative success beyond additional training, while next-frame supervision further improves performance preservation under relaxed verification and decoding efficiency.

4.4 Real-Robot Evaluation

Table 3 shows that WA-SpecDec accelerates MolmoAct inference while improving overall task success over Standard SpecDec on four real-robot tasks under three relaxed acceptance rules. On SweepSteak, WA-SpecDec raises success from 64% to 82% under the ρ-based rule and increases inference speedup over AR decoding from $1.50\times$ to $1.74\times$. These gains show that the benefit extends to physical tool–object interaction, supporting faster action generation with reliable task execution. Appendix C provides details of the real-robot setup, policy training, and evaluation protocol.

5 Conclusion

In this study, we presented WA-SpecDec, a world-aware speculative decoding framework for VLA models. By injecting world-model-derived physical scene awareness before prefill computation, WA-SpecDec produces world-aware prefill hidden states shared between draft generation and target verification, improving relaxed decoding without changing the acceptance rule. Experiments on LIBERO, SIMPLER-Env, and four real-robot tasks show improved task success and inference speed over Standard SpecDec. On LIBERO, NCF decreases across both evaluated policies, while relaxation sweeps with OpenVLA show higher success under looser acceptance and longer accepted prefixes. These results suggest that physically informed shared conditioning is an effective approach for improving the reliability of accelerated VLA action decoding.

AI USE STATEMENT

In this work, we have not used generative AI tools for tasks with required disclosure, and the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for language editing, grammar checking, and assisting with software code editing. We have reviewed all AI-assisted work. Specifically, the authors manually reviewed the AI-assisted revisions to ensure that they preserved the intended technical meaning, and verified and tested AI-assisted code modifications for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies efficient VLA decoding in simulated manipulation environments, including LIBERO and SIMPLER-Env, and in physical robot experiments on tabletop manipulation tasks. We use publicly available benchmarks and model components, together with demonstrations collected through robot teleoperation. The study does not collect personal data or involve human-subject evaluation. Any released code and evaluation scripts will respect the licenses and terms of use of the underlying models, datasets, and simulators.

REPRODUCIBILITY STATEMENT

Section 3 describes the WA-SpecDec architecture and training objectives, while Algorithm 1 in Appendix A summarizes training and inference. Section 4.1 describes the benchmarks, baselines, comparison protocol, and evaluation metrics. Appendix B provides policy construction procedures, model, training, and decoding configurations, evaluation budgets, and hardware and timing details. Appendix B.5 specifies the random seeds used in simulation and explains how task success, speedup, and NCF are aggregated across seeds and evaluation settings. Appendix C documents the real-robot setup, demonstration collection, policy adaptation, task success criteria, and evaluation protocol. Code and evaluation scripts will be released upon acceptance.

REFERENCES

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=QaCCuDfBk2. Featured Certification.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander 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 S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H 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 *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.025.

Remi Cadene, Simon Alibert, Alexander Soare, Quentin Gallouedec, Adil Zouitine, Steven Palma, Pepijn Kooijmans, Michel Aractingi, Mustafa Shukor, Dana Aubakirova, Martino Russi, Francesco Capuano, Caroline Pascal, Jade Choghari, Khalil Meftah, Maxime Ellerbach, Jess

Moss, and Thomas Wolf. Lerobot: State-of-the-art machine learning for real-world robotics in pytorch. https://github.com/huggingface/lerobot, 2024.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5209–5235. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/cai24b.html.

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

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-e: An embodied multimodal language model. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Zhenyu He, Zexuan Zhong, Tianle Cai, Jason Lee, and Di He. Rest: Retrieval-based speculative decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1582–1595, 2024.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as i can, not as i say: Grounding language in robotic affordances. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-Driven Representation Learning for Robotics. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.032.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W Mahoney, Amir Gholami, and Kurt Keutzer. Speculative decoding with big little decoder. *Advances in Neural Information Processing Systems*, 36:39236–39256, 2023.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, 2014. URL https://arxiv.org/abs/1312.6114.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact: Action reasoning models that can reason in space, 2025. URL https://arxiv.org/abs/2508.07917.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, 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. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3705–3728. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/li25c.html.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28935–28948. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/li24bt.html.

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.

Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can RL bring to VLA generalization? An empirical study. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 97121–97151. Curran Associates, Inc., 2025. doi: 10.52202/085713-3248. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8c0fabe372177d2aded596be2d3b4544-Paper-Conference.pdf.

Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Tingfan Wu, Jay Vakil, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? *Advances in Neural Information Processing Systems*, 36:655–677, 2023.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pp. 932–949, 2024.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 892–909. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/nair23a.html.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

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. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.012.

Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pp. 416–426. PMLR, 2023.

Wan Team, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2(3):4, 2025.

Songsheng Wang, Rucheng Yu, Zhihang Yuan, Chao Yu, Feng Gao, Yu Wang, and Derek F Wong. Spec-vla: speculative decoding for vision-language-action models with relaxed acceptance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 26928–26940, 2025.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023.

Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. Efficientvla: Training-free acceleration and compression for vision-language-action models. *Advances in Neural Information Processing Systems*, 38:40891–40914, 2026.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. Draft& verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11263–11282, 2024.

Zihao Zheng, Zhihao Mao, Maoliang Li, Jiayu Chen, Xinhao Sun, Zhaobo Zhang, Donggang Cao, Hong Mei, and Xiang Chen. KERV: Kinematic-rectified speculative decoding for embodied VLA models. In *Proceedings of the 63rd ACM/IEEE Design Automation Conference (DAC)*, 2026a. To appear.

Zihao Zheng, Zhihao Mao, Sicheng Tian, Maoliang Li, Jiayu Chen, Xinhao Sun, Zhaobo Zhang, Xuanzhe Liu, Donggang Cao, Hong Mei, et al. Heisd: Hybrid speculative decoding for embodied vision-language-action models with kinematic awareness. *arXiv preprint arXiv:2603.17573*, 2026b.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL `https://proceedings.mlr.press/v229/zitkovich23a.html`.

APPENDIX

A PSEUDOCODE

Algorithm 1 summarizes the training procedure and presents a simplified sequence-level abstraction of WA-SpecDec inference. The experiments use speculative tree decoding with the configuration described in Appendix B.3. Tree construction, tree-attention masking, and accepted-path selection are omitted from the pseudocode for clarity. The input block collects the notation used throughout the algorithm, and the output block specifies the trained models and the continuous control sequence returned for a current query. At inference, the training-only auxiliary branch is removed, and WA-SpecDec uses only the current instruction–observation pair while applying the original relaxed acceptance rule without modification.

Algorithm 1 WA-SpecDec training and inference

Require: Training set $\mathcal{D} = \{(x, I_t, I_{t+1}, a^\star_{1:K})\}$, where x is the instruction, I_t the current image, I_{t+1} the training-only next image, $a^\star_{1:K}$ the demonstration action tokens, and K the number of action tokens decoded for one control query. Frozen VAE encoder E_{vae}, visual embeddings V_t, text embeddings T_x, and frozen base VLA parameters θ. Trainable modules: token projector ϕ_{tok}, spatial VideoEncoder F_ω over VAE latent tokens, bias projector f_{bias}, training-only latent prediction head g_η, and LoRA adapter $\Delta\theta$. Auxiliary weight λ. Draft model q_ϕ, proposal operator Draft_ϕ, block length γ, distance d, tolerance ρ, candidate action token a, and detokenizer Detok.

Ensure: Trained target p^{wa}_θ, trained draft q_ϕ, and continuous control sequence $u_{1:K}$ for a current query (x, I_t).

Train the world-aware target

for each minibatch of samples $(x, I_t, I_{t+1}, a^\star_{1:K})$ from $\mathcal{D}$ **do**
  $Z_t \leftarrow E_{\text{vae}}(I_t)$ ▷ Eq. 3
  $U_t \leftarrow F_\omega(\phi_{\text{tok}}(Z_t))$ ▷ Eq. 4
  $w_t \leftarrow f_{\text{bias}}(\text{SpatialAlign}(U_t))$ ▷ Eq. 5
  $V^{\text{wa}}_t \leftarrow V_t + w_t$ and $P^{\text{wa}}_t \leftarrow [T_x; V^{\text{wa}}_t]$ ▷ Eqs. 8–9
  $h^{\text{wa}}_t \leftarrow \text{Prefill}_{\theta+\Delta\theta}(P^{\text{wa}}_t)$ ▷ Eq. 9
  Compute $\mathcal{L}_{\text{action}}$ from p^{wa}_θ and $a^\star_{1:K}$. ▷ Eq. 10
  Compute $\mathcal{L}_{\text{video}}$ from $g_\eta(U_t)$ and $E_{\text{vae}}(I_{t+1})$. ▷ Eq. 6
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{action}} + \lambda\mathcal{L}_{\text{video}}$ ▷ Eq. 7
  Update ϕ_{tok}, F_ω, f_{bias}, g_η, and $\Delta\theta$ by minimizing $\mathcal{L}$, keeping the frozen modules fixed.
end for
Freeze p^{wa}_θ and train q_ϕ by minimizing $\mathcal{L}_{\text{draft}}$. ▷ Eqs. 11, 14

Run one inference control query

Discard g_η and do not use I_{t+1} or $\mathcal{L}_{\text{video}}$ at inference.
Compute Z_t, U_t, w_t, V^{wa}_t, P^{wa}_t, and h^{wa}_t from the current (x, I_t). ▷ Eqs. 3, 4, 5, 8–9
$A \leftarrow ()$ and $i \leftarrow 1$
while $i \le K$ **do**
  $r \leftarrow \min(\gamma, K - i + 1)$ and $J \leftarrow \{i, \dots, i + r - 1\}$
  $\hat{a}_{i:i+r-1} \leftarrow \text{Draft}_\phi(h^{\text{wa}}_t, A, r)$
  In one target verification pass, compute $a^{\text{wa}}_j \leftarrow \arg\max_a p^{\text{wa}}_\theta(a \mid h^{\text{wa}}_t, a_{<j})$ for all $j \in J$. ▷ Eq. 12
  Use prefix $a_{<j} = [A; \hat{a}_{i:j-1}]$ for each verified position j.
  $m \leftarrow 0$
  while $m < r$ **and** $d(\hat{a}_{i+m}, a^{\text{wa}}_{i+m}) \le \rho$ **do** ▷ Eq. 13
    Append $\hat{a}_{i+m}$ to A and set $m \leftarrow m + 1$
  end while
  if $m < r$ **then**
    Append a^{wa}_{i+m} to A ▷ replace the first rejected draft token
    $i \leftarrow i + m + 1$
  else
    $i \leftarrow i + m$
  end if
end while
$u_{1:K} \leftarrow \text{Detok}(A)$
return $u_{1:K}$

B Implementation Details

We evaluate five policy–benchmark settings: OpenVLA and RynnVLA-002-Discrete on LIBERO, OpenVLA and RLVLA-RL on SIMPLER-Env, and MolmoAct on the physical robot. Each setting has its own base-policy AR reference. The following sections distinguish the policy construction procedure from the benchmark-specific evaluation settings.

B.1 Policy Construction

Within each setting, the AR reference and Standard SpecDec use the same base VLA policy. For the simulation experiments, this reference is the corresponding fine-tuned model. For the physical robot, we obtain the reference by fine-tuning a MolmoAct base checkpoint with LoRA on the collected demonstrations. WA-SpecDec starts from the corresponding reference policy and jointly trains WAB and the target-policy LoRA adapter with the action token and auxiliary next-frame latent objectives in Section 3.3. The VAE encoder and base visual stack remain frozen. After this adaptation stage, we freeze the world-aware target and distill the draft model from its output distribution using Equation 14.

At inference, WAB provides world-aware conditioning through prefill hidden states shared between draft generation and target verification. The VideoHead is discarded, and world-aware conditioning uses only the current observation. Within each paired comparison, Standard SpecDec and WA-SpecDec use the same draft configuration and acceptance rule settings. The resulting comparison includes both target adaptation and speculative decoding, with the component ablation in Section 4.3 evaluating their separate effects for OpenVLA on LIBERO.

B.2 Benchmark-Specific Settings

OpenVLA on LIBERO. We use a fine-tuned OpenVLA-7B base policy on LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long. Each method is evaluated over 50 rollouts per task, yielding 500 episodes per suite, with matched task splits, rollout budgets, and simulator settings. Section 4.2 discusses the main comparisons in Table 1 and the acceptance–success sweep in Figure 4. Section 4.3 reports the component ablation, and Appendix E.3 reports the separate tolerance sweep.

RynnVLA on LIBERO. We additionally evaluate RynnVLA-002-Discrete on the same four LIBERO suites. Its AR reference is the model fine-tuned on LIBERO data, and WAB/LoRA adaptation and draft distillation use the LIBERO fine-tuning data. Each method is evaluated over 50 rollouts per task, yielding 500 episodes per suite. Table 2 reports averages over the four suites, using this fine-tuned model as the reference for both success and latency.

OpenVLA and RLVLA-RL on SIMPLER-Env. We evaluate OpenVLA and RLVLA-RL in the WidowX setup of SIMPLER-Env on Carrot, Stack, Spoon, and Eggplant, using 192 episodes per task. For each policy, the AR reference is its corresponding fine-tuned model, and WAB/LoRA adaptation and draft distillation use the fine-tuning data for this WidowX setting. Tables 5 and 6 report the results for OpenVLA and RLVLA-RL, respectively. Standard SpecDec and WA-SpecDec are compared under the ρ-based, KERV, and HeiSD relaxed acceptance rules with matched settings within each pair. Appendix E.1 discusses the task-level results.

MolmoAct on the physical robot. We evaluate MolmoAct on MoveBread, LiftKettle, SweepSteak, and PlaceMushroom. The base checkpoint is fine-tuned with LoRA on 250 demonstrations per task, and each method is evaluated over 50 trials per task. Appendix C specifies the demonstration collection, MolmoAct adaptation procedure, robot platform, and physical evaluation protocol. Table 3 reports the corresponding results.

B.3 Model, Training, and Decoding Configuration

Across all five settings, the auxiliary next-frame latent prediction loss is weighted by $\lambda = 0.1$, the spatial VideoEncoder has four layers, and the same frozen Wan 2.2 VAE encoder serves as the latent

compressor. Given the current RGB observation I_t, the encoder produces the spatial latent map Z_t defined in Equation 3. The map is tokenized, linearly projected, and processed by the VideoEncoder before being aligned to the VLA visual patch grid. The auxiliary VideoHead is used only during training, so inference does not run next-frame prediction, VAE decoding, or video generation.

Target-policy adaptation. We jointly train the target-policy LoRA adapters and all trainable WAB modules with learning rates of 10^{-4} and 2×10^{-4}, respectively. We zero-initialize the weights and bias of the final linear layer in the bias projector, so that $w_t = 0$ at initialization and the pretrained visual embeddings are initially preserved. LoRA uses rank 16, scaling parameter 32, and dropout 0.05, applied to the query, key, value, and output projections in the language model attention layers. The original VLA parameters and VAE encoder remain frozen. Adaptation runs for five additional epochs with an effective batch size of 32 and 1,000 warm-up steps, minimizing $\mathcal{L}_{\text{action}} + 0.1\mathcal{L}_{\text{video}}$. The continued-training control in the OpenVLA/LIBERO ablation uses the same data, training budget, and LoRA settings without WAB or next-frame supervision.

Draft distillation. After target adaptation, the world-aware target is frozen, and the draft is distilled using the forward KL objective in Equation 14, averaged over valid action token positions at temperature $T = 1$. The draft uses a single decoder layer with a linear fusion layer and frozen token embeddings from the corresponding target policy. We use AdamW with a learning rate of 5×10^{-5}, zero weight decay, and an effective batch size of 32. Training runs for 50,000 optimizer updates in BF16, with 1,000 warm-up steps followed by cosine decay to 5×10^{-6}.

Decoding configuration. Speculative tree decoding uses at most 50 tree nodes and the top-8 tokens to construct the draft tree. The draft-tree depth parameter is set to 4 by default. For Figure 4, Standard SpecDec and WA-SpecDec both use a depth parameter of 6 to explore the acceptance–success trade-off over a wider range of accepted lengths. These tree settings are matched between Standard SpecDec and WA-SpecDec within each experiment.

For the main LIBERO experiments, we set $\rho = 10$ for the ρ-based relaxed acceptance rule and use the default settings of the KERV and HeiSD acceptance rules. The separate tolerance-sensitivity experiment in Table 8 evaluates OpenVLA at $\rho \in \{5, 10, 15, 20\}$.

B.4 Metrics and Timing

For the LIBERO evaluations, NCF is computed from simulator-state geometry. For each failed episode, we measure the minimum distance over the full trajectory between the gripper fingertip sites and the nearest points on the surfaces of the task-relevant object or target receptacle. An episode contributes to NCF if this minimum distance falls below $\delta = 2\,\text{cm}$ before termination. The same threshold is used across both LIBERO target policies, task suites, and acceptance rules. Appendix E.4 varies this threshold for OpenVLA.

All inference timing measurements use NVIDIA RTX A6000 GPUs. Within each setting, wall-clock speedup is measured relative to autoregressive decoding of the corresponding base policy on the same hardware setup. The reported speedup includes inference-time computation for draft proposal, target verification, WAB, and key–value cache updates. For the physical robot, observation preprocessing, action token decoding, and action postprocessing are also included, while robot actuation and scene-reset time are excluded, as specified in Appendix C.

All experiments are conducted using four NVIDIA RTX A6000 GPUs. Training, distillation, and evaluation require approximately 960 GPU-hours in total, corresponding to roughly 240 hours of wall-clock time when using all four GPUs in parallel.

B.5 Aggregation across Evaluation Settings

All reported metrics from simulation experiments, including those in the main comparisons, ablations, and tolerance sweeps, are arithmetic means over four random seeds: 1111, 2222, 3333, and 4444. The simulation evaluation episode counts specified throughout the paper are per seed.

The overall success and speedup results combine five settings: OpenVLA and RynnVLA-002-Discrete on LIBERO, OpenVLA and RLVLA-RL on SIMPLER-Env, and MolmoAct on the real

robot. For each setting, we average paired success-rate differences and speedup factors equally over the four suites or tasks and three relaxed acceptance rules, then average equally across the five settings, using seed-averaged metrics for the simulation settings. Success gains are measured in percentage points relative to Standard SpecDec. Speedup is the ratio of base-policy AR inference latency to WA-SpecDec inference latency, and we take the arithmetic mean of these ratios. These aggregates use the main comparisons in Tables 1, 2, 3, 5, and 6, excluding tolerance sweeps and ablations.

For each LIBERO suite and seed, NCF is the percentage of failed episodes that are near-contact failures. The reported suite-level NCF is the arithmetic mean over the four seeds. To compute mean relative NCF reductions, we first average these seed-averaged NCF values equally across the four suites for each method, target policy, and acceptance rule. We then compute the relative reduction from Standard SpecDec using the suite means. For a single target policy, we average these reductions equally over the three acceptance rules. For the overall LIBERO result, we average equally over the six combinations of two target policies and three rules.

For the threshold-sensitivity results in Table 9, we pool failed episodes across the four LIBERO suites within each seed and evaluation configuration. At each distance threshold, NCF is the total number of near-contact failures across these suites divided by the total number of failed episodes across the same suites, expressed as a percentage. We then average these percentages equally over the four seeds.

C Real-Robot Experiments

Objective. Simulation provides a controlled setting for evaluating decoding efficiency, but it does not fully capture the consequences of prediction errors once generated actions are coupled to physical dynamics. In particular, small deviations that remain recoverable during free-space motion may lead to irreversible failures near contact. We therefore conduct a real-robot evaluation to determine whether WA-SpecDec can reduce action-generation latency while maintaining physical task success relative to autoregressive decoding of the base policy.

Physical platform. Our real-robot platform is shown in Figure 6. The system consists of a paired SO-101 leader–follower setup and a fixed overhead RGB camera observing the tabletop workspace. The leader arm is used exclusively for teleoperated demonstration collection, whereas the follower arm autonomously executes the learned policy during evaluation. Low-level joint commands are executed at 30 Hz. Before each data collection or evaluation session, we verify the robot calibration, camera placement, workspace configuration, and leader–follower correspondence. All methods use the same observation modalities (visual observations and robot proprioception), language instructions, action representation, and low-level controller.

Demonstration dataset. For each of the four tasks, we collect 250 teleoperated demonstrations, resulting in 1,000 real-robot trajectories in total. During demonstration collection, an operator controls the SO-101 leader arm while the follower arm reproduces the commanded motion in the physical workspace. Each demonstration pairs the language instruction with the synchronized visual observations, robot states, and executed actions used for policy fine-tuning. All evaluation episodes are collected after training and are disjoint from the demonstration data.

Policy adaptation. We instantiate the training procedure in Appendix B.1 with MolmoAct (Lee et al., 2025), which generates discrete action tokens autoregressively. We first fine-tune a MolmoAct base checkpoint with LoRA on the four-task demonstration dataset to obtain the base policy used by the AR reference and Standard SpecDec. For WA-SpecDec, we start from this fine-tuned reference and jointly train WAB and the target-policy LoRA adapter on the same demonstration data, following Section 3.3. This stage uses the action token objective and auxiliary next-frame latent prediction loss while keeping the VAE and base visual stack frozen. We then freeze the resulting world-aware target and distill the draft model from its output distribution using Equation 14. During evaluation, WA-SpecDec uses this adapted target for verification. The auxiliary prediction head is discarded, and only the current observation is used for world-aware conditioning. This procedure preserves the target policy's action space and the underlying robot controller.

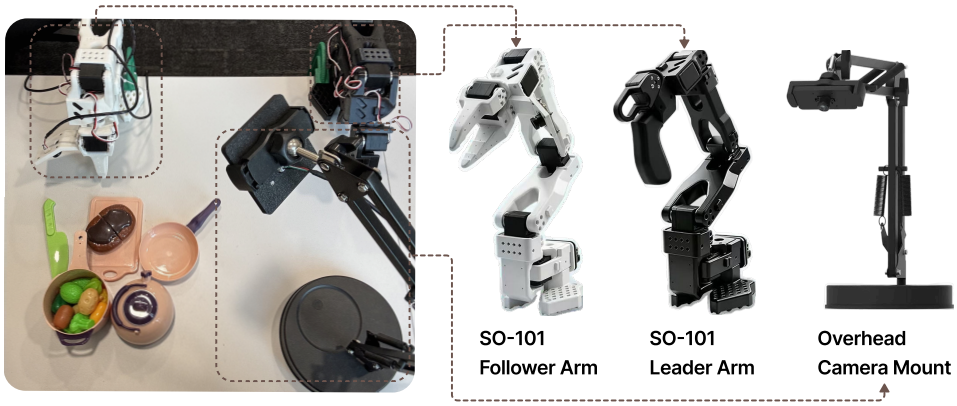


Figure 6: **Real-robot experimental platform. Left:** Overhead view of the tabletop workspace used for teleoperated demonstration collection and physical evaluation. **Right:** The SO-101 follower arm used for autonomous policy execution, the SO-101 leader arm used for demonstration collection, and the fixed overhead camera mount.

Table 4: Real-robot task instructions and success criteria. All tasks have a 15 s time limit. The terminal conditions below must hold continuously for at least 2 s.

Task and instruction	**Success criteria**
MoveBread Move the bread from the right side of the kettle to the left side.	The bread's center is at least 5 cm left of the kettle's center. The bread is released, fully supported by the table, and stable. The kettle remains upright and moves no more than 3 cm from its initial position.
LiftKettle Pick up the kettle by its handle.	The gripper grasps the handle, not the kettle body. The kettle's lowest point is at least 5 cm above the table without external support. The kettle is not dropped and remains within 30° of upright.
SweepSteak Use the knife to sweep the steak off the cutting board.	The knife is already secured in the gripper at the start. Knife–steak contact moves the steak completely off the cutting board's top surface, leaving it supported by the table and stable. Direct gripper contact with or grasping of the steak is a failure.
PlaceMushroom Move the mushroom from the cutting board into the frying pan.	After release, the mushroom lies entirely within the pan's inner rim. Contact with or straddling of the rim is a failure. The pan remains upright and moves no more than 3 cm from its initial position.

Task suite. We evaluate four tabletop manipulation tasks spanning complementary interaction regimes. MoveBread requires spatial rearrangement relative to a reference object. LiftKettle requires a handle-constrained grasp followed by stable lifting. SweepSteak requires tool-mediated object displacement: the episode begins with a knife already secured in the robot gripper, and the policy must use the knife to sweep the steak off the cutting board. Finally, PlaceMushroom requires precise placement within a receptacle. Table 4 specifies the language instruction and operational success criterion for each task.

Evaluation protocol. We evaluate MolmoAct (AR), Standard SpecDec (SpecDec in Table 3), and WA-SpecDec, with 50 independent episodes per task and configuration, totaling 200 rollouts per configuration. The speculative methods use the ρ-based, KERV, and HeiSD relaxed acceptance rules, with matched draft configurations and acceptance settings. Standard SpecDec verifies against the base MolmoAct policy, whereas WA-SpecDec uses the adapted world-aware target. The comparison therefore captures the combined effect of target adaptation and speculative decoding.

Before each episode, the robot and objects are reset to standardized task-specific configurations using the same scene templates across methods. Policy state, observation history, decoding caches, and action-generation state are also reset. Task and decoding conditions are interleaved to reduce

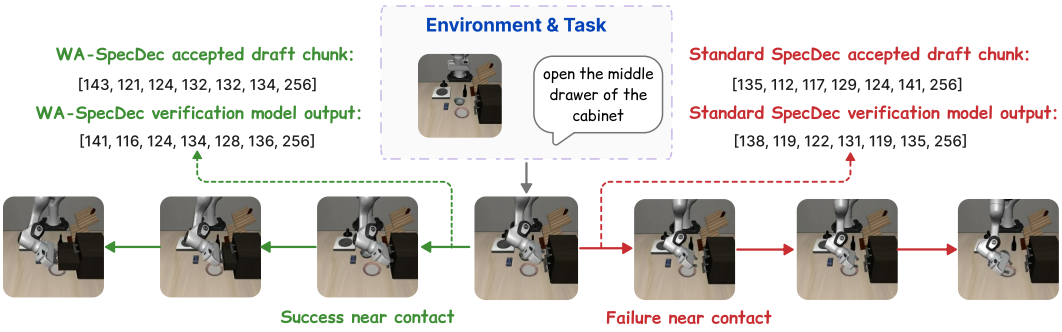


Figure 7: Case study of accepted action chunks under relaxed verification with $\rho = 10$ for the LIBERO-Goal task "open the middle drawer of the cabinet." Green denotes the WA-SpecDec accepted chunk and successful near-contact execution, while red denotes the Standard SpecDec accepted chunk and near-contact failure.

temporal confounding. Episodes have a 15 s limit and succeed when the criteria in Table 4 hold continuously for at least 2 s. Timeouts, object drops, manipulation of an incorrect object, prohibited strategies, unsafe policy-induced collisions, and human intervention after the first policy action count as failures and are retained without replacement. Trials invalidated by external technical or initialization errors before task execution are logged separately and repeated.

Outcome adjudication. Every rollout is recorded from two viewpoints. Three evaluators independently assess task success from these recordings using the predefined criteria in Table 4. The decoding configuration is hidden during outcome assessment. A rollout is counted as successful only when all three evaluators judge it successful.

Metrics. Table 3 reports physical task success and inference speedup for each task and evaluated configuration. Succ. is the percentage of the 50 evaluation episodes that satisfy the task's success criteria. Spd. is the ratio of the action-generation latency of the base MolmoAct policy under native autoregressive decoding to that of the evaluated configuration. MolmoAct (AR) therefore has a speedup of $1.00\times$ for each task. Latency measurements include all inference-time computation, including observation preprocessing, WAB computation for WA-SpecDec, draft generation, target verification, action token decoding, key–value cache updates, and action postprocessing.

Robot actuation time, human scene-reset time, and video-recording overhead are excluded from action-generation latency. All latency measurements are collected on NVIDIA RTX A6000 GPUs using the same deployment hardware setup across methods after model warm-up. Together, these metrics evaluate whether WA-SpecDec reduces the inference cost of MolmoAct without compromising physical task success.

D Case Study

Figure 7 shows a representative near-contact case for the task "open the middle drawer of the cabinet." Both Standard SpecDec and WA-SpecDec produce an accepted action chunk under relaxed verification with $\rho = 10$. Although both chunks satisfy the same token-distance relaxed acceptance rule, their execution outcomes differ substantially. The Standard SpecDec chunk is accepted in token space, but the resulting trajectory fails near the drawer-contact region. In contrast, WA-SpecDec completes the drawer-opening motion successfully.

This example illustrates a failure mode of token-distance-based relaxed acceptance in contact-sensitive states. A draft chunk that remains close to the target output in token space can still lead to task-ineffective motion when execution depends on fine-grained geometry near an object surface or handle. WA-SpecDec mitigates this issue by injecting world-aware information into prefill hidden states shared between draft generation and target verification, conditioning both processes on additional physical scene structure. In this case, the accepted WA-SpecDec chunk is better aligned with

Table 5: OpenVLA results on four SIMPLER-Env tasks. Succ. is task success (%). Spd. is wall-clock speedup relative to autoregressive decoding of the base OpenVLA policy. Rule denotes the relaxed acceptance rule. Each pair uses the same rule, settings, and decoding configuration.

Decoding Method		**Carrot**		**Stack**		**Spoon**		**Eggplant**	
Rule	**Policy**	**Succ.**	**Spd.**	**Succ.**	**Spd.**	**Succ.**	**Spd.**	**Succ.**	**Spd.**
N/A	AR: OpenVLA	51.56	1.00×	27.08	1.00×	60.42	1.00×	16.15	1.00×
ρ	Standard SpecDec	43.75	1.26×	23.44	1.19×	60.94	1.23×	14.06	1.20×
ρ	WA-SpecDec	53.65	1.43×	26.56	1.56×	63.54	1.38×	17.71	1.48×
KERV	Standard SpecDec	44.79	1.44×	25.00	1.50×	57.29	1.43×	13.54	1.71×
KERV	WA-SpecDec	52.60	1.67×	30.73	1.74×	61.98	1.78×	18.23	1.92×
HeiSD	Standard SpecDec	46.88	2.31×	24.48	2.04×	58.33	1.95×	15.10	2.23×
HeiSD	WA-SpecDec	54.17	2.78×	27.60	2.55×	67.19	2.67×	17.19	2.36×

Table 6: RLVLA-RL results on the same four SIMPLER-Env tasks. Succ. is task success (%). Spd. is wall-clock speedup relative to autoregressive decoding of the base RLVLA-RL policy. Rule denotes the relaxed acceptance rule. Each pair uses the same rule, settings, and decoding configuration.

Decoding Method		**Carrot**		**Stack**		**Spoon**		**Eggplant**	
Rule	**Policy**	**Succ.**	**Spd.**	**Succ.**	**Spd.**	**Succ.**	**Spd.**	**Succ.**	**Spd.**
N/A	AR: RLVLA-RL	77.1	1.00×	57.8	1.00×	83.3	1.00×	40.6	1.00×
ρ	Standard SpecDec	73.4	1.28×	58.3	1.31×	82.3	1.18×	36.5	1.33×
ρ	WA-SpecDec	79.2	1.44×	60.4	1.83×	85.4	1.59×	41.1	1.62×
KERV	Standard SpecDec	76.0	1.57×	55.7	1.65×	83.9	1.42×	37.0	1.76×
KERV	WA-SpecDec	80.2	1.89×	58.3	1.88×	86.5	1.80×	42.2	1.91×
HeiSD	Standard SpecDec	77.6	2.45×	54.2	2.01×	81.8	2.12×	39.1	2.31×
HeiSD	WA-SpecDec	80.7	2.74×	58.3	2.46×	85.4	2.63×	43.2	2.49×

the near-contact execution requirement, improving reliability without changing the relaxed acceptance rule itself.

E Additional Experiments

E.1 Evaluation on SIMPLER-Env

We evaluate WA-SpecDec on SIMPLER-Env to determine whether the improvements observed on LIBERO persist in a different simulation environment. For OpenVLA, WA-SpecDec improves both task success and wall-clock speedup over Standard SpecDec across all four tasks and three acceptance rules (Table 5). On Carrot with ρ-based verification, success increases from 43.75% to 53.65%, while speedup relative to base-policy AR increases from $1.26\times$ to $1.43\times$. The simultaneous improvement in both metrics demonstrates a more favorable success–efficiency trade-off under matched acceptance settings.

We also evaluate RLVLA-RL, whose higher AR success across all four tasks provides a stronger target policy within SIMPLER-Env. WA-SpecDec retains its advantage in both success and speedup across all task–rule combinations (Table 6). On Stack with ρ-based verification, it achieves a $1.83\times$ speedup compared with $1.31\times$ for Standard SpecDec, while improving success from 58.3% to 60.4%. These results support the applicability of WA-SpecDec across simulation environments and show that its gains persist with a stronger target policy.

Table 7: Near-contact failure percentages and episode counts on LIBERO with OpenVLA, using the same evaluation settings as Table 1. SpecDec denotes Standard SpecDec.

		NCF ↓		**NCF episodes count ↓**	
Suite	**Rule**	SpecDec	WA-SpecDec	SpecDec	WA-SpecDec
Spatial	ρ	78.6%	72.9%	81	62
	KERV	75.8%	72.3%	75	60
	HeiSD	69.0%	65.3%	78	64
Object	ρ	60.0%	48.7%	111	76
	KERV	63.9%	48.1%	117	76
	HeiSD	63.8%	49.0%	102	72
Goal	ρ	64.5%	46.3%	98	56
	KERV	61.6%	47.3%	90	53
	HeiSD	66.4%	43.6%	97	48
Long	ρ	52.2%	44.7%	132	98
	KERV	54.1%	42.8%	139	101
	HeiSD	51.6%	42.3%	128	91

Table 8: Effect of progressively increasing the relaxation tolerance ρ on task success across four LIBERO task suites with OpenVLA as the target policy. SpecDec and Ours denote Standard SpecDec and WA-SpecDec, respectively. Both use the same ρ-based relaxed acceptance rule at each tolerance value.

	–	$\rho = 5$		$\rho = 10$		$\rho = 15$		$\rho = 20$	
Suite	AR target	SpecDec	**Ours**	SpecDec	**Ours**	SpecDec	**Ours**	SpecDec	**Ours**
Spatial	82.8%	79.0%	**82.4%**	79.4%	**83.0%**	76.0%	**80.8%**	72.6%	**78.4%**
Object	69.8%	64.2%	**69.0%**	63.0%	**68.8%**	61.8%	**66.4%**	57.0%	**65.8%**
Goal	78.2%	73.2%	**76.4%**	69.6%	**75.8%**	66.4%	**72.4%**	58.2%	**66.8%**
Long	57.8%	54.2%	**57.2%**	49.4%	**56.2%**	40.6%	**48.2%**	32.4%	**43.0%**
Mean	72.2%	67.7%	**71.3%**	65.4%	**71.0%**	61.2%	**67.0%**	55.1%	**63.5%**

E.2 Near-Contact Failure Episode Counts

Table 7 supplements Table 1 with the corresponding NCF episode counts. Since the reported NCF metric is normalized by the number of failed episodes (see Section 4.1), these counts show whether improvements in the conditional measure also correspond to fewer NCF episodes across the full evaluation set. WA-SpecDec yields consistent reductions in both measures across all four LIBERO suites and all three acceptance rules. Compared with Standard SpecDec on LIBERO-Goal with HeiSD, WA-SpecDec reduces the conditional NCF from 66.4% to 43.6% and the number of NCF episodes from 97 to 48. This agreement between these two views shows that the improvement is evident both within the failure set and across the full evaluation budget.

E.3 Sensitivity to the Relaxation Tolerance

Table 8 reports sensitivity to the relaxation tolerance ρ for OpenVLA across four LIBERO suites. The ρ-based acceptance rule and tolerance are held fixed within each comparison. As ρ increases from 5 to 20, Standard SpecDec's mean success drops from 67.7% to 55.1%, while WA-SpecDec retains 71.3%, 71.0%, 67.0%, and 63.5% at $\rho = 5, 10, 15, 20$, respectively. The smaller decline demonstrates greater robustness to permissive verification without changing the acceptance criterion. These gains are consistent with the benefit of conditioning draft proposal and the target reference on shared world-aware prefill hidden states.

Table 9: Sensitivity of near-contact failure (NCF, %) to the distance threshold δ for OpenVLA across four LIBERO suites. Rule denotes the relaxed acceptance rule.

Rule	Policy	1 cm	2 cm	3 cm	4 cm
N/A	AR	36.83	51.17	59.72	62.87
ρ	Standard SpecDec	44.64	60.89	67.93	72.35
ρ	WA-SpecDec	35.20	50.26	58.07	63.12
KERV	Standard SpecDec	46.85	61.46	68.15	73.34
KERV	WA-SpecDec	32.82	49.24	56.75	62.78
HeiSD	Standard SpecDec	45.31	60.72	67.43	70.21
HeiSD	WA-SpecDec	31.17	48.25	56.07	65.90

E.4 Sensitivity to the Near-Contact Threshold

We examine whether the NCF improvement is robust to the distance threshold used to define near contact. Our default threshold of $\delta = 2\,\mathrm{cm}$ approximates the width of one gripper finger in the LIBERO simulator, providing a physically motivated scale for identifying near-contact episodes. Table 9 reports pooled NCF for OpenVLA across the four LIBERO suites as δ varies from 1 to $4\,\mathrm{cm}$. WA-SpecDec achieves lower NCF than Standard SpecDec under every verifier family at every evaluated threshold. Averaged across the three families, the absolute reductions are 12.54, 11.77, 10.87, and 8.03 percentage points at 1, 2, 3, and $4\,\mathrm{cm}$, respectively. The larger reductions at 1–$2\,\mathrm{cm}$ indicate that the NCF advantage is more pronounced under stricter definitions of near contact. This advantage persists throughout the evaluated range, supporting the robustness of the comparison to the threshold choice.