ADAMAP: Learning Adaptive Multi-Action Prediction with Grounded Dreaming Guidance

Muyao Li[*] ByteDance Seed, Peking University
Zihao Wang[*] ByteDance Seed, Peking University
XuJing Li ByteDance Seed
Xiangyang Li ByteDance Seed, Peking University
Jiabo Ye East China Normal University
Zhiyong Wu ByteDance Seed
Yaodong Yang Peking University
Yujia Qin ByteDance Seed

Abstract

Large agentic models can now operate in complex digital games after pretraining on web-scale data, but gameplay trajectories often contain highly redundant actions. Single-action agents can exploit this redundancy spuriously by copying recent actions, leading to causal confusion. Fixed-length action chunks reduce this ambiguity by predicting multiple future actions at once, yet their open-loop execution can accumulate errors when the agent cannot react to new observations. Action horizons should therefore be learned as state-dependent temporal abstractions: the agent should commit to longer chunks when feedback is less needed and stop early when new observations are valuable. We introduce ADAMAP (Adaptive Multi-Action Prediction), which learns how many actions to generate at each decision point, grounds each chunk with interleaved visual dreams, and further improves the resulting policy through online reinforcement learning. Across Minecraft and VizDoom, ADAMAP improves multi-action control, with a 3.6-percentage-point gain in average Minecraft success over fixed-horizon RL. The learned horizons vary systematically across tasks and games, suggesting that adaptive multi-action prediction discovers useful temporal abstractions.

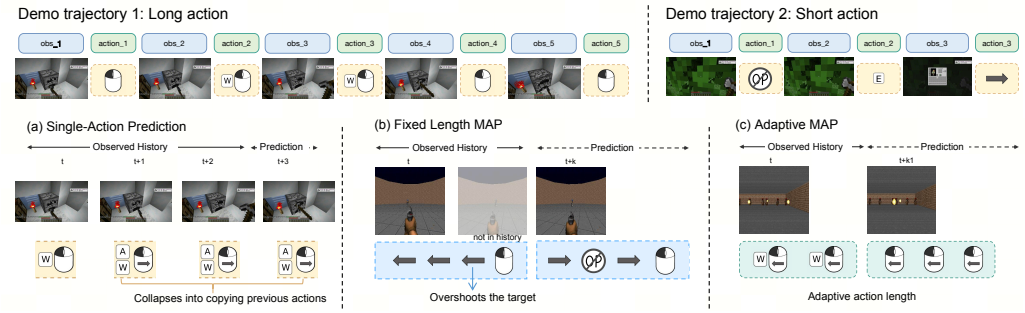


Figure 1: **From single-action prediction to Adaptive MAP. (a) Classical Single-Action Prediction** allows immediate correction but suffers from causal confusion, slow inference, and difficulty in learning multi-step behaviors. **(b) MAP with Fixed Length/Horizon** mitigates causal confusion via multi-step prediction, but its rigid, open-loop execution is prone to error accumulation. **(c) Adaptive MAP (ADAMAP)** dynamically scales the action horizon per state, effectively resolving both the causal confusion of (a) and the error accumulation of (b).

[*]Equal contribution.

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

1 Introduction

Foundation agentic models have made rapid progress in complex interactive environments. After large-scale pretraining on web data, recent agents can play digital games [8, 35, 54, 56, 59], manipulate physical objects [9, 18], and operate graphical interfaces [40, 44, 64] from visual observations and language instructions. Most of these agents are trained and deployed as single-action predictors: given the current observation and history, the model predicts the next atomic action and then waits for a new observation [63]. However, gameplay trajectories are highly redundant. Offline demonstrations often contain long stretches of repeated low-level actions, such as holding a movement key to walk forward or producing no operation while waiting [59, 68]. In such data, a single-action learner can achieve high next-action accuracy by copying recent action regularities rather than grounding its decision in the causal state of the game, a failure mode known as causal confusion [16, 51].

Multi-action prediction offers a natural way to mitigate this causal ambiguity. In single-step imitation, the model predicts one action at a time, so highly correlated actions can often be explained by copying recent action patterns rather than by modeling the temporal phase of the behavior. Action chunking changes the prediction target from an isolated next action to a short future action sequence [70], executed before the next observation is incorporated. This local sequence provides temporal context for otherwise ambiguous actions: repeated actions may indicate waiting, holding a key, sustained movement, or the continuation of a multi-frame maneuver, rather than a purely Markovian response to the current observation. Thus, the chunk serves as a compact temporal abstraction that captures short-horizon behavioral structure and reduces ambiguity in the imitation target. This principle underlies action chunking methods in robot learning [5–7, 15], where multi-action prediction both stabilizes behavior prediction and amortizes one model inference over multiple environment steps.

However, once an action chunk is emitted, it is executed open-loop: later actions in the chunk cannot condition on observations that arrive after earlier actions have changed the state. This exposes MAP again to the classical compounding-error problem in sequential decision making [45, 46]: small prediction errors or environment changes can shift the agent into states not anticipated by the chunk, while the agent cannot incorporate fresh observations until the chunk ends [49, 60]. This problem is especially severe in games, where semantic boundaries can appear abruptly: opening a chest reveals a new GUI, an enemy may move between frames, or an over-rotated camera can make all subsequent movement actions point in the wrong direction. Fixed-length action chunks force the same horizon onto all of these situations. Short horizons preserve feedback but weaken the benefit of MAP; long horizons reduce single-step ambiguity but amplify open-loop error accumulation. The central problem is how they should choose the horizon of each chunk so that MAP reduces causal confusion without losing control to accumulated errors.

We propose **ADAMAP** (Adaptive Multi-Action Prediction), an adaptive MAP framework in which the horizon of each action chunk is predicted rather than fixed. At every decision point, ADAMAP conditions on the current observation and history, generating an action chunk with a state-dependent horizon, committing to longer chunks in predictable states and returning to observation sooner when the situation becomes uncertain or fast-changing. We warm-start this behavior from offline demonstrations by constructing adaptive-horizon action chunks based on predictive confidence. Yet even with an adaptive horizon, the agent must still trust the actions it commits to before the next observation arrives. To address this, ADAMAP uses **Interleaved Grounded Visual Dreaming**, which interleaves actions with dreams inside each action chunk: lightweight and verifiable natural-language descriptions of the expected visual consequences after intermediate actions. These dreams provide chunk-internal guidance for subsequent action generation and expose whether the imagined future remains visually grounded before the next environment observation arrives.

Offline imitation provides a stable initialization, but the best horizon policy depends on the agent's own state distribution and task reward. Demonstration-derived horizons may be locally plausible, yet they are not necessarily optimal for task success, interaction efficiency, or recovery from accumulated errors. We therefore train ADAMAP through a multi-step stage paradigm: imitation learning first establishes a stable adaptive MAP prior, and **MAP-PPO** then optimizes the policy through environment interaction. During online optimization, **MAP-PPO** jointly refines action choices, dream reliability, and horizon decisions, while task rewards determine when longer open-loop commitments are actually beneficial.

Experiments on Minecraft and VizDoom show that ADAMAP improves over single-action prediction, fixed-horizon MAP, and alternative segmentation baselines. The gains are not explained by variable-length chunks alone: methods with similar average horizons can perform substantially worse, indicating that where the chunk boundaries are placed matters as much as how long the chunks are. The learned horizons also align with task demands, becoming shorter for reactive combat and first-person-shooter scenarios and longer for more predictable embodied tasks. After online optimization, ADAMAP further improves performance while retaining this horizon awareness, suggesting that adaptive MAP can discover useful task-dependent temporal abstractions without explicit skill labels.

Our key contributions are: **(1)** We propose **ADAMAP**, an adaptive multi-action prediction framework that learns the horizon of each action chunk. **(2)** We introduce **Interleaved Grounded Visual Dreaming**, which interleaves actions with grounded visual primitives as imagination observation inside each action chunk, providing verifiable future visual guidance that reduces accumulated errors and supports more reliable long-horizon prediction. **(3)** We develop a training paradigm for MAP, using imitation learning to initialize adaptive horizons and **MAP-PPO** to further online optimize through environment interaction, yielding substantial performance gains across various games.

2 Problem Formulation

We consider an agent interacting with a partially observable environment [31]. At time t, the policy conditions on a history representation

$$s_t \;=\; f(o_{t-H:t},\; a_{t-H:t}), \tag{1}$$

where o and a denote observations and atomic actions. Single-step policies predict one action and receive feedback after every environment step. Multi-action prediction (MAP) instead emits an action chunk

$$C_t^{(k)} \;=\; (a_t, a_{t+1}, \ldots, a_{t+k-1}), \qquad \pi(C_t^{(k)} \mid s_t), \tag{2}$$

and executes its k actions consecutively before receiving the next observation [5, 6, 15, 70]. This amortizes inference and reduces trivial step-to-step copying, but it also makes the agent open-loop inside the chunk.

This mismatch produces the accuracy–efficiency trade-off shown in Figure 1. Single-step control is accurate but slow; fixed-length chunks are fast but can be semantically misaligned. The key limitation is not multi-action prediction itself, but the assumption that one horizon can serve all states, tasks, and action types. A controlled study reported in §4.4 makes this concrete: holding the model fixed and varying only k on Minecraft, the optimal horizon differs by more than $4\times$ across task categories, and no single k avoids degradation on at least one category.

ADAMAP treats the temporal horizon as a learned decision variable. Instead of fixing k, the agent emits a variable-length chunk and terminates it with STOP, returning to observation when the current semantic unit is complete or when further actions require unobserved information. Appendix B gives the full formulation in terms of length policies, adaptive decision times, and stop decisions.

3 Learning Adaptive Multi-Action Prediction

This section describes how ADAMAP makes adaptive MAP trainable. The agent must learn a state-dependent action chunk that is useful as a temporal abstraction, reliable under open-loop execution, and improvable through online interaction. We warm-start adaptive chunk boundaries from demonstrations, ground longer chunks with interleaved visual dreams, and then optimize actions, dreams, and stopping decisions with MAP-PPO.

3.1 Adaptive MAP Output and Observation-Gap Warmup

For a state s_t, ADAMAP predicts an adaptive-length action sequence

$$\big(a_t, \ldots, a_{t+k_t-1}, \textsc{Stop}\big) \sim \pi_\theta(\cdot \mid s_t), \qquad k_t \le K. \tag{3}$$

The environment executes the k_t actions before STOP; the token only marks the chunk boundary in the output [53]. Unlike fixed-horizon MAP, k_t varies across states, so chunk boundaries can align with the reliability of open-loop execution.

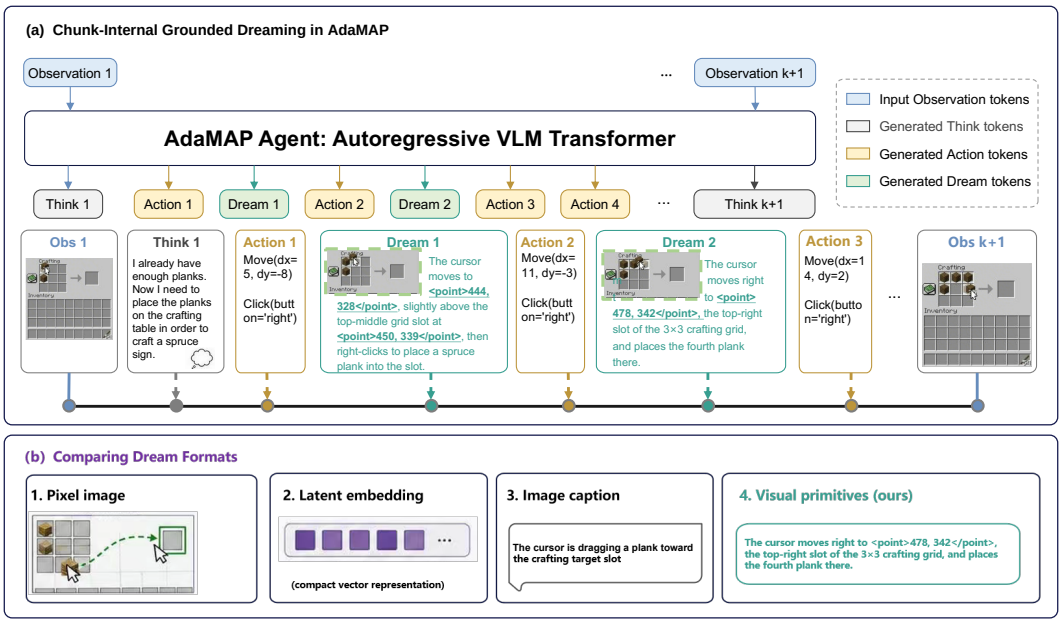


Figure 2: **ADAMAP output format and dream representation. (a) Chunk-internal grounded dreaming:** Given an observation, ADAMAP autoregressively generates thinking tokens followed by an adaptive-length chunk that interleaves environment actions with dream tokens before receiving the next observation (o_{k+1}). **(b) Comparing dream formats:** our dreams use *grounded visual primitives*, concise natural-language predictions anchored with explicit spatial coordinates (e.g., `<point>`).

We first construct warmup supervision from demonstrations using a base MAP model π_0 trained on multi-horizon expert trajectories, following prior evidence that uncertainty and change-point signals can reveal temporal boundaries [2, 17, 37]. For an expert continuation $\mathbf{a}^*_{t:t+K-1}$, each candidate boundary after k actions compares the next-action distribution from open-loop continuation with the distribution obtained after refreshing the state with the realized observation:

$$\begin{aligned} p_k^{\mathrm{ol}} &= \pi_0(\cdot \mid s_t, \mathbf{a}^*_{t:t+k-1}), \\ p_k^{\mathrm{cl}} &= \pi_0(\cdot \mid s_{t+k}), \\ g_k(s_t) &= D_{\mathrm{KL}}\big(p_k^{\mathrm{cl}} \,\|\, p_k^{\mathrm{ol}}\big)\,. \end{aligned} \tag{4}$$

The observation gap g_k measures how much the next-action belief changes when the model sees the intermediate visual state. A large gap indicates that further open-loop continuation from s_t is stale, so we place the boundary at the largest significant gap, or use the maximum horizon if no gap exceeds τ_{gap}:

$$k_t := \begin{cases} \arg\max_{1\le k<K} g_k(s_t), & \max_{1\le k<K} g_k(s_t) > \tau_{\mathrm{gap}}, \\ K, & \text{otherwise.} \end{cases} \tag{5}$$

The resulting targets $(a^*_t, \ldots, a^*_{t+k_t-1}, \textsc{Stop})$ form $\mathcal{D}_{\mathrm{adapt}}$ and initialize π_{adapt}. Appendix C gives the full state construction, thresholding, tie-breaking, and trajectory segmentation algorithm.

3.2 Interleaved Grounded Visual Dreaming

Adaptive horizons decide how long a chunk should remain open-loop, but the actions inside a chunk still depend on whether the model can anticipate their intermediate consequences. As shown in Figure 2, Interleaved Grounded Visual Dreaming augments action generation with grounded visual descriptors: concise natural-language predictions of verifiable state changes such as cursor locations, bounding boxes, object motions, inventory changes, and UI transitions, rather than pixel- or latent-level futures [3, 12, 33]. The model generates actions and dreams as a single augmented chunk:

$$\bar{C}_t = \big(a_t,\ d_{t+1},\ a_{t+1},\ d_{t+2},\ \ldots,\ a_{t+k_t-1},\ \textsc{Stop}\big). \tag{6}$$

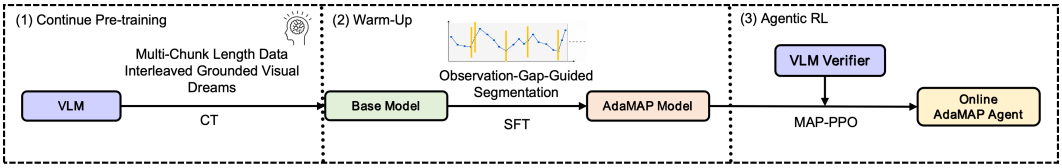


Figure 3: **Training paradigm of AdaMAP.** AdaMAP is trained in three stages: continual training teaches multi-horizon MAP and the action-dream format; warmup initializes adaptive chunks from demonstrations; MAP-PPO refines the warm-start model through online interaction.

Each dream d_{t+i} predicts the changed visual state caused by the preceding action span, and subsequent actions condition on earlier actions and dreams.

Dream supervision is constructed offline so that each d_{t+i} predicts a verifiable visual change rather than paraphrases a known future frame. The procedure follows three stages—a teacher extracts changed-state rubrics from the future frame, a dreamer predicts the change without ever seeing that frame, and a verifier retains only grounded predictions—and is detailed in Appendix D (Figure 6). We fine-tune π_{adapt} on the resulting action-dream sequences with a standard autoregressive loss, yielding π_{warm}. At deployment time, the policy emits dreams directly inside $\bar{C}_t$; no teacher, verifier, rubric extractor, or future frame is used.

3.3 Online Optimization with MAP-PPO

The warm-start policy is still grounded in offline evidence: its horizons imitate demonstration-derived boundaries and its dreams are supervised on observed transitions. MAP-PPO turns it into an online AdaMAP agent by optimizing over chunk-level interaction. At chunk boundary s_j, the policy samples $\bar{C}_j \sim \pi_\theta(\cdot \mid s_j)$; the environment executes only the action subsequence C_j, while dreams and STOP remain generated tokens.

The reward combines task success, dream consistency, and a KL penalty against the warm-start reference policy:

$$r_{t'} = \underbrace{r_{t'}^{\text{final}}}_{\text{task completion}} + \underbrace{\beta\, R_{\text{dream}}(s_{t'}, d_{t'})\, m_{t'}}_{\text{dream consistency}} - \underbrace{\beta_{\text{KL}}\, \text{KL}_{\text{step}}(t')}_{\text{drift penalty}}. \tag{7}$$

Here $m_{t'}$ indicates whether a dream is evaluated at atomic step t', and the dream reward is produced by a visual verifier model that checks the generated dream against the realized visual change.

We optimize π_θ with a PPO-style objective over the entire augmented chunk [48]. The importance ratio is token-normalized [72] to avoid systematically larger log-ratio magnitudes for longer generations:

$$\rho_j(\theta) = \left(\frac{\pi_\theta(\bar{C}_j \mid s_j)}{\pi_{\theta_{\text{old}}}(\bar{C}_j \mid s_j)} \right)^{1/|\bar{C}_j|}, \tag{8}$$

and the clipped surrogate uses the chunk-level advantage $\bar{A}_j$:

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_j\Big[\min\big(\rho_j(\theta)\bar{A}_j,\ \text{clip}(\rho_j(\theta), 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})\bar{A}_j\big)\Big]. \tag{9}$$

Because STOP is part of the generated sequence, the termination decision receives credit through the same objective as action and dream tokens. Appendix F.1 gives the state-value critic, TD residual, and chunk-level GAE used to construct $\bar{A}_j$; Appendix F.2 lists hyperparameters.

3.4 Adaptive Horizon Annealing

Online optimization introduces a subtle bias in horizon selection. When rollouts are bounded by a fixed cap on chunk invocations, committing to longer chunks allows the agent to execute more atomic actions before episode termination. Consequently, unchecked horizon growth may merely reflect extra exploration steps rather than more effective temporal abstraction. We address this bias with **Adaptive Horizon Annealing (AHA)**, a success-conditioned curriculum that progressively tightens the episode-level interaction budget.

Let Y denote the maximum number of chunk invocations per episode and B the current atomic-step budget. An episode spanning $J \leq Y$ chunks consumes $L = \sum_{j=1}^{J} k_j$ atomic steps. AHA leaves B unchanged until task success stabilizes and a buffer $\mathcal{S}$ of at least l recent successful episode lengths has been accumulated, after which the budget is updated via

$$B \leftarrow \max\Big(B_{\min},\ \min\big(B,\ \mathrm{Quantile}_p(\mathcal{S})\big) \Big), \tag{10}$$

where $\mathrm{Quantile}_p(\mathcal{S})$ is the p-th quantile of $\mathcal{S}$ and $B_{\min}$ is a hard safety floor. This rule shrinks the budget monotonically down to $B_{\min}$ while naturally pausing annealing on sparse-reward tasks with insufficient successes. Any episode that exhausts B before task completion is terminated as a failure ($r^{\mathrm{final}} = 0$).

Crucially, AHA tightens the rollout constraint while leaving the reward formulation and PPO objective intact. Rather than prescribing a target horizon or penalizing long chunks directly, it requires the policy to solve the task within a shrinking atomic-step budget under the same chunk-invocation cap Y. This selective pressure prunes wasteful open-loop actions while preserving long commitments when they genuinely accelerate task completion.

4 Experiments

We design experiments around four questions: **(Q1)** Does adaptive MAP outperform both single-action prediction and fixed-horizon MAP? (§4.2) **(Q2)** Does Interleaved Grounded Visual Dreaming extend the reliable horizon of multi-action prediction? (§4.3) **(Q3)** Does MAP-PPO improve adaptive horizon selection beyond what generic online RL provides? (§4.3) **(Q4)** Do the learned horizons reflect task and environment dynamics without explicit supervision? (§4.4)

4.1 Experimental Setup

Model and Data. Our model builds upon GameTars-MoE-Mini [59], initialized from Seed1.5-VL [25] (2.5B activated parameters) with a 632M SigLIP visual encoder [69]. The training corpus comprises diverse human gameplay trajectories formatted into three splits: 50B tokens of fixed-horizon MAP prediction data (size-balanced across chunk lengths), 3B tokens of dreaming-augmented data (Appendix D), and 8B tokens of adaptive-horizon data generated via observation-gap-guided segmentation (§3.1). During MAP-PPO, we use Seed1.8 [10] to provide the dream-consistency reward. Crucially, to properly evaluate generalization, the training corpus strictly excludes all data from the VizDoom evaluation scenarios.

Evaluation Benchmarks. We evaluate our approach across two distinct game environments: (a) **Minecraft:** We use an open-world benchmark built on MineRL [26] and adopt the OpenHA protocol [58]. This includes over 800 hand-curated tasks across three categories: *Embodied* (physical 3D interactions), *GUI* (menu and crafting operations), and *Combat* (fast-paced engagements with hostile mobs). We report the best-of-3 success rate averaged across tasks per category. (b) **VizDoom:** We select three official scenarios [62] (DefendLine, TakeCover, and HealthGathering) and report the per-episode return. Full benchmark details are provided in Appendix E.

Baselines. To ensure a rigorous evaluation, all controlled baselines and our models are fine-tuned from the same base checkpoint using identical source trajectories, differing only in the action formulation strategy. We compare our approach against three paradigms: (1) **Prompted Generalist Agents:** GameTARS [59], Gemini-3-Flash [23], and GPT-5.2 [39], evaluated via standard single-action control without the MAP format. (2) **Supervised Fine-Tuning (SFT) Baselines:** This includes ReAct [63] (single-action prediction), MAP@n (SFT using MAP data with a fixed horizon of n), and BPE-MAP [50] (pre-segmenting and post-processing action sequences using BPE). (3) **Our Variants:** ADAMAP-SFT (adaptive-horizon warmup combined with Interleaved Grounded Visual Dreaming, §3), and ADAMAP* (which incorporates online optimization via MAP-PPO, §3.3).

Table 1: **Main results across open-world and FPS tasks.** We compare ReAct, MAP with fixed horizon(MAP@n), BPE-guided MAP, and ADAMAP variants under SFT and RL settings, together with frontier generalist baselines. Open-world columns report best-of-3 success rate (%); FPS columns report game scores. Horizon is the mean chunk length generated by the agent. * = after online RL. Red = best overall; blue = best before RL.

Method	Open-world Minecraft				FPS Vizdoom			
	Embodied	GUI	Combat	Horizon	DL	TC	HG	Horizon
Baseline								
GameTARS [59]	68.42	66.06	**73.77**	1.0	4.7	182.7	917.3	1.0
Gemini-3-Flash* [23]	70.83	0.00	0.00	1.0	8.3	227.0	602.7	1.0
GPT-5.2* [39]	36.67	0.00	50.00	1.0	**12.0**	166.7	538.7	1.0
Supervised Finetuning								
ReAct (no MAP) [63]	69.08	68.42	72.13	1.0	7.7	282.3	1077.3	1.0
MAP@4	65.79	73.14	62.3	3.9	7.7	**392.3**	816.0	4.0
MAP@8	68.42	73.14	63.93	7.6	9.7	265.7	810.7	7.9
MAP@16	65.79	66.79	49.18	15.6	7.7	243.3	538.7	15.7
BPE-MAP [50]	58.80	40.2	56.7	4.2	7.7	182.7	709.3	7.0
ADAMAP (SFT Only)	**71.71**	**74.55**	63.93	4.6	10.7	230.7	**1214.7**	3.8
Reinforcement Learning								
ReAct (no MAP)* [63]	82.89	**94.00**	81.97	1.0	11.7	218.3	**1600.0**	1.0
MAP@8*	83.55	90.36	77.05	7.5	11.0	142.7	**1600.0**	7.9
ADAMAP*	**84.21**	93.82	**83.61**	13.0	**12.7**	**497.7**	**1600.0**	4.3

4.2 Main Results

Table 1 summarizes the evaluation results across the Minecraft and VizDoom benchmarks.

Superiority of Adaptive Horizons in SFT. Under the imitation learning setting, ADAMAP-SFT emerges as the strongest method overall in Minecraft. It achieves the highest success rates in both Embodied (71.71%) and GUI (74.55%) tasks, while remaining highly competitive in Combat. Notably, it consistently outperforms the best-performing fixed-horizon variants (MAP@n). Furthermore, ADAMAP-SFT demonstrates robust zero-shot generalization on the out-of-distribution VizDoom scenarios without any environment-specific tuning, and transfers effectively to continuous robotic manipulation on Language-Table [38] (Appendix G). These results confirm that dynamically predicting the optimal action horizon per state is significantly more effective than committing to a static chunk length.

Adaptive vs. Heuristic Chunking. A natural follow-up question is whether the performance gain stems strictly from our adaptive mechanism, or merely from producing variable-length chunks. The BPE-MAP baseline provides a direct ablation: it generates variable-length chunks with a similar average horizon to ADAMAP (4.2 vs. 4.6) but determines boundaries based on surface-level action repetition and data frequencies. This heuristic segmentation performs substantially worse across all three Minecraft categories (e.g., 40.20% vs. 74.55% on GUI). This sharp contrast isolates our core mechanism: ADAMAP strategically places chunk boundaries where the model's open-loop prediction becomes unreliable (high observation gap), whereas heuristic methods may arbitrarily split coherent behaviors or merge unrelated actions.

Unlocking Online RL Potential. After online optimization via MAP-PPO (§3.3), ADAMAP* dominates the benchmarks, achieving the highest or tied-highest scores in 5 out of 6 task categories. The critical comparison here is against ReAct*, which applies the exact same RL procedure to single-action predictions. ADAMAP* matches or exceeds ReAct*'s performance (e.g., 84.21% vs. 82.89% on Embodied). Furthermore, ADAMAP* outpaces its optimized fixed-horizon counterpart (MAP@8*) given the identical RL budget. This demonstrates that adaptive MAP is not merely a

better initialization for SFT; it provides a richer, more expressive action representation that RL can exploit to jointly optimize *what* actions to execute and *how many* to commit to per decision.

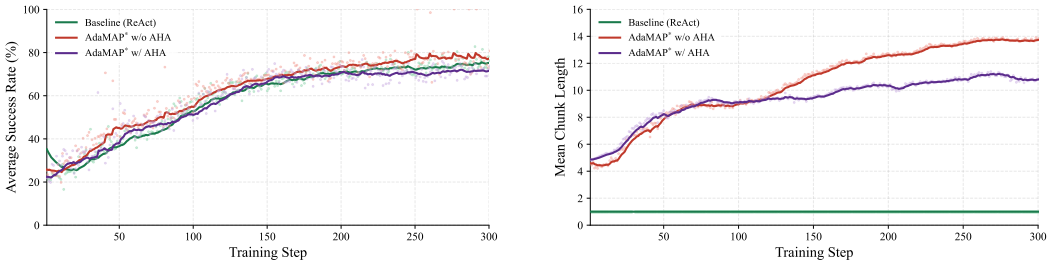
Inference Efficiency. Beyond task success, we compare the end-to-end environment interaction throughput of SFT models on Minecraft. Single-action ReAct is bottlenecked by per-step model invocation at 0.59 FPS, whereas fixed-horizon MAP@8 reaches 2.90 FPS, albeit at the cost of open-loop error accumulation. Despite generating additional intra-chunk dream tokens, ADAMAP-SFT achieves 2.57 FPS, a $4.36\times$ speedup over ReAct, with only a 5.3% throughput reduction relative to its no-dream counterpart. Thus, ADAMAP-SFT retains the throughput advantage of multi-action prediction while achieving the highest task success rate.

4.3 Ablation Studies

Dreaming extends reliable multi-action prediction. Adaptive horizons decide how many actions to predict, but the agent must still trust those actions once committed. Interleaved Grounded Visual Dreaming addresses this by maintaining a lightweight internal state estimate, so the model can condition later actions on anticipated consequences rather than generating them blindly. We isolate this effect by removing dream tokens from ADAMAP-SFT while keeping the adaptive-horizon pipeline unchanged (Table 2). We evaluate both variants on a 300-task Minecraft benchmark using three independent seeds and report mean ASR with its across-seed sample standard deviation. Dreaming improves overall ASR from 51.06% to 54.95%, a gain of 3.89 percentage points. The clearest signal is in the entropy: mean per-chunk action entropy drops from 0.134 to 0.099 with standard deviation shrinking from 0.047 to 0.017. Lower entropy means the model is more certain about what to predict next within a chunk, and the average horizon extends from 4.00 to 4.66. The model learns to predict more actions per chunk precisely because the interleaved dreams make each additional action less uncertain.

Table 2: **Ablation of Interleaved Grounded Visual Dreaming on Minecraft.** We compare ADAMAP-SFT trained with and without intra-chunk visual dreams while keeping the adaptive-horizon warmup pipeline identical. ASR (%) reports the mean success rate and sample standard deviation across three independent seeds on the 300-task benchmark, Entropy measures the mean per-chunk action prediction uncertainty, and Horizon is the average executed chunk length.

Variant	ASR (%)	Entropy	Horizon
w/o Dream	$51.06^{\pm 3.49}$	$0.134^{\pm 0.047}$	$4.00^{\pm 1.87}$
w/ Dream	$\mathbf{54.95}^{\pm 1.58}$	$\mathbf{0.099}^{\pm 0.017}$	$\mathbf{4.66}^{\pm 1.59}$



(a) Average success rate.

(b) Mean horizon.

Figure 4: **Online RL training dynamics on Minecraft.** (a) All three methods converge to similar success rates ($\sim$75%); curves begin after critic warm-up. (b) Without AHA, the mean horizon grows unchecked to $\sim$14.6; AHA caps it at $\sim$13 while preserving comparable performance.

MAP-PPO improves horizon selection beyond generic online RL. Figure 4 tracks three online RL runs on Minecraft: ReAct (k=1), ADAMAP* without AHA, and ADAMAP* with full AHA annealing (§3.4). All three converge to comparable success rates ($\sim$75%, Figure 4a), but the horizon trajectories (Figure 4b) reveal a subtlety specific to adaptive MAP under RL. Without AHA, the mean horizon drifts from $\sim$5 to $\sim$14.6: because each rollout has a fixed cap on chunk invocations,

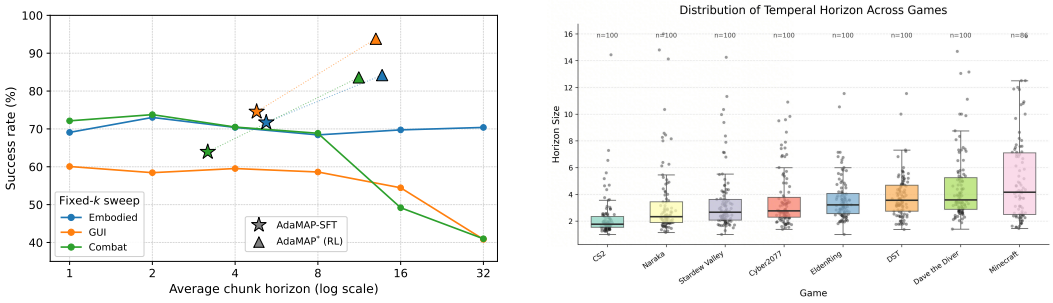
the agent learns that longer chunks cover more temporal ground per rollout and exploits this as a free exploration bonus. AHA removes this bias by progressively tightening the atomic-step budget once the success rate stabilizes, capping the mean horizon at $\sim$13.0.

Table 3 confirms that this regularization generalizes across both long- and short-horizon domains. Across Minecraft and VizDoom HealthGathering (HG), AHA consistently trims the average chunk length by $\sim$11.3% (14.65 $\rightarrow$ 13.00 on Minecraft and 4.87 $\rightarrow$ 4.32 on VizDoom HG) while preserving or slightly improving task performance (73.39% $\rightarrow$ 74.05% ASR on Minecraft; 1600 saturated score on HG). Rather than collapsing the policy toward single-step control, AHA prunes only low-utility open-loop actions, forcing the agent to commit to long chunks selectively when the state transition is predictable.

Table 3: **Ablation of Adaptive Horizon Annealing (AHA) across Minecraft and VizDoom.** We compare online RL policies (AdaMAP*) trained with and without AHA under identical MAP-PPO settings, reporting average success rate (ASR, %) on Minecraft, episode return on VizDoom HealthGathering (HG), and the converged mean chunk length (Horizon). AHA trims redundant horizon inflation by $\sim$11% across both domains while preserving or slightly improving task performance.

Strategy	Minecraft		VizDoom	
	ASR (%)	Horizon	HG Score	Horizon
AdaMAP* w/o AHA	73.39	14.65	**1600**	4.87
AdaMAP* w/ AHA	**74.05**	13.00	**1600**	4.32

Backbone capacity. We also ablate backbone capacity by scaling the activated parameters from 2.5B to 7B while holding the AdaMAP pipeline fixed (Appendix H). The 7B variant consistently improves zero-shot returns across all three VizDoom scenarios, confirming that AdaMAP scales favorably with stronger visual-language representations.



(a) Controlled horizon study on Minecraft.

(b) Horizon distributions across games.

Figure 5: **Horizon analysis across games and tasks.** (a) We fix the base model and training data, varying only the inference-time horizon $k \in \{1, 2, 4, 8, 16, 32\}$ (solid lines). Per-category optima diverge by more than $4\times$. Stars mark AdaMAP-SFT and triangles mark AdaMAP* (RL), each at its per-category mean horizon; both sit above the fixed-k envelope. (b) Horizon distributions produced by AdaMAP-SFT across games; each point is one trajectory's mean horizon, boxes show quartiles. Fast-paced games concentrate on short horizons, while open-world games exhibit longer and more dispersed horizons, without game-specific tuning.

4.4 Learned Adaptive Horizons Reflect Task and Environment Dynamics

We now revisit the premise behind adaptive MAP: useful action horizons are task-dependent rather than global constants. We test this directly with a controlled fixed-horizon sweep on Minecraft, holding the checkpoint and data fixed while varying only the inference-time chunk length $k \in \{1, 2, 4, 8, 16, 32\}$ (Figure 5a). If a single global horizon were sufficient, the three task categories should peak at similar values of k. Instead, Figure 5a shows separated optima: feedback-sensitive tasks favor shorter horizons, while more continuous tasks tolerate longer commitments. The best

fixed horizons differ by more than $4\times$ across categories, confirming that horizon choice is intrinsically task-dependent.

The star markers then ask whether ADAMAP discovers this structure on its own. ADAMAP-SFT assigns the shortest horizons to Combat (3.18), an intermediate horizon to GUI (4.79), and the longest horizon to Embodied tasks (5.19), without task labels or explicit horizon supervision. Thus the learned horizon recovers the task ordering revealed by the controlled sweep.

This pattern is not confined to Minecraft. Figure 5b applies the same ADAMAP-SFT model to trajectories from diverse games without game-specific fine-tuning. Fast-paced competitive games concentrate at short horizons, while open-world games shift longer and exhibit wider spread, reflecting mixtures of sustained movement, inventory manipulation, and abrupt reactive events. This behavior is reminiscent of unsupervised skill discovery [19, 24]: the model segments behavior into coherent temporal units whose granularity varies with task dynamics.

Online optimization preserves this separation. The triangular markers in Figure 5a correspond to larger horizons, but remain task-dependent rather than collapsing to a single operating point.

5 Related Work

Error Accumulation in Action Chunking. Action chunking is now the dominant paradigm for flow-matching [5, 42] and autoregressive [21, 32] models, with parallel lines of work on chunk execution [6, 7, 14, 49] and representation [34, 41, 57, 74]. Action chunking improves execution efficiency and trajectory smoothness by predicting multiple future actions at once, but it can also cause error accumulation because it does not receive updated state information during execution and therefore cannot correct deviations in its actions. A handful of recent works begin to address error accumulation from different perspectives. AAC [37] truncates a chunk when per-step entropy exceeds a hand-tuned threshold. VLASH [55] fixes state mismatch in asynchronous VLA inference by predicting the future state using previous actions, improving real-time response without changing the model architecture. ADAMAP complements these approaches by extending the reliable prediction range via intra-chunk dreaming, self-determining where each chunk should end through observation-gap-guided boundary discovery, and refining this strategy through online chunk-level RL.

World Models for Stabilizing Long-Horizon Prediction. Learned dynamics models have long been used to improve policies [27–29], and a growing body of work brings this idea to large-scale agent policies [1, 12, 13, 30, 61, 65]. Explicit inference-time imagination [22, 43, 71, 73] helps internalize world knowledge into the model's representations, improving action quality [4, 67]. Yet nearly all these world-model signals live between action steps [43, 71, 73]. Inside a committed chunk, the policy still flies blind. WorldVLA [12] interleaves action and image decoding in one autoregressive model, but pixel-level prediction leaves the action modality under-trained and intra-chunk conditioning *degrades* performance, forcing the authors to mask prior actions entirely. ADAMAP's Interleaved Grounded Visual Dreaming (§3.2) brings world-model feedback *inside* the chunk for the first time: the prediction is internalized into the model's own autoregressive pass as adaptive, concise natural-language descriptions of the expected environmental consequence rather than costly pixel-level futures.

6 Conclusion

We presented ADAMAP, a framework that turns temporal horizon from a fixed hyperparameter into a learned decision variable. The agent learns to predict farther through interleaved dreaming, to stop when open-loop predictions diverge from closed-loop beliefs, and to refine this strategy through online RL. The resulting horizon choices align with task semantics without any explicit supervision, echoing the temporal abstraction long sought by the Options framework [53] but emerging here end-to-end.

Limitations. Our evaluation focuses on simulated environments; deploying ADAMAP on physical robot hardware would further strengthen generality claims. ADAMAP also builds on an autoregressive VLM; extending adaptive horizon to diffusion models [5, 15] is a natural next step.

Acknowledgments and Disclosure of Funding

We thank the Seed Multimodal Interaction and World Model team for their infrastructure support, and our colleagues at ByteDance for their support of the ADAMAP project.

References

[1] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

[2] Hemanath Arumugam, Falong Fan, and Bo Liu. Enhancing hierarchical reinforcement learning through change point detection in time series. *arXiv preprint arXiv:2510.24988*, 2025.

[3] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

[4] BeingBeyond Team. Being-H0.7: A latent world-action model from egocentric videos. `https://research.beingbeyond.com/being-h07`, 2026.

[5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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

[7] Kevin Black, Allen Z. Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

[8] Adrian Bolton, Alexander Lerchner, Alexandra Cordell, Alexandre Moufarek, Andrew Bolt, Andrew Lampinen, Anna Mitenkova, Arne Olav Hallingstad, Bojan Vujatovic, Bonnie Li, et al. SIMA 2: A generalist embodied agent for virtual worlds. *arXiv preprint arXiv:2512.04797*, 2025.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, 2023.

[10] ByteDance Seed. Seed1.8 model card: Towards generalized real-world agency. *arXiv preprint arXiv:2603.20633*, 2026.

[11] ByteDance Seed. Seed 2.0. `https://seed.bytedance.com/en/seed2`, 2026.

[12] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[13] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[14] Tongqing Chen, Hang Wu, Jiasen Wang, Xiaotao Li, and Lu Fang. StreamVLA: Breaking the reason-act cycle via completion-state gating. *arXiv preprint arXiv:2602.01100*, 2026.

[15] 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*, pages 1684–1704, 2025.

[16] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems*, 2019.

[17] Jingwen Deng, Zihao Wang, Shaofei Cai, Anji Liu, and Yitao Liang. Open-world skill discovery from unsegmented demonstrations. *arXiv preprint arXiv:2503.10684*, 2025.

[18] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[19] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019.

[20] Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayanaraju, Xinze Guan, and Xin Eric Wang. GRIT: Teaching MLLMs to think with images. In *Advances in Neural Information Processing Systems*, 2025.

[21] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini Robotics: Bringing AI into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[22] GigaBrain Team, Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Haoyun Li, Jie Li, Jiagang Zhu, Lv Feng, et al. GigaBrain-0: A world model-powered vision-language-action model. *arXiv preprint arXiv:2510.19430*, 2025.

[23] Google DeepMind. Gemini 3. `https://deepmind.google/technologies/gemini/`, 2025.

[24] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. In *International Conference on Learning Representations*, 2017.

[25] Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, et al. Seed1.5-VL technical report. *arXiv preprint arXiv:2505.07062*, 2025.

[26] William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. MineRL: A large-scale dataset of Minecraft demonstrations. *arXiv preprint arXiv:1907.13440*, 2019.

[27] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[28] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[29] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[30] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

[31] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

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

[33] Yann LeCun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

[34] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.

[35] Muyao Li, Zihao Wang, Kaichen He, Xiaojian Ma, and Yitao Liang. JARVIS-VLA: Post-training large-scale vision language models to play visual games with keyboards and mouse. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

[36] Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=XUks1Y96NR`.

[37] Yuanchang Liang, Xiaobo Wang, Kai Wang, Shuo Wang, Xiaojiang Peng, Haoyu Chen, David Kim Huat Chua, and Prahlad Vadakkepat. Adaptive action chunking at inference-time for vision-language-action models. *arXiv preprint arXiv:2604.04161*, 2026.

[38] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

[39] OpenAI. GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[40] OpenAI. Operator System Card. `https://cdn.openai.com/operator_system_card.pdf`, 2025.

[41] 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.

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

[43] Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: A steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[44] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. UI-TARS: Pioneering automated GUI interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

[45] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, 2010.

[46] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, 2011.

[47] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[48] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[49] Kohei Sendai, Maxime Alvarez, Tatsuya Matsushima, Yutaka Matsuo, and Yusuke Iwasawa. Leave no observation behind: Real-time correction for VLA action chunks. *arXiv preprint arXiv:2509.23224*, 2025.

[50] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

[51] Seokin Seo, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. Regularized behavior cloning for blocking the leakage of past actions. In *Advances in Neural Information Processing Systems*, 2023.

[52] Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. VLM-R1: A stable and generalizable R1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

[53] Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

[54] Weihao Tan, Xiangyang Li, Yunhao Fang, Heyuan Yao, Shi Yan, Hao Luo, Tenglong Ao, Huihui Li, Hongbin Ren, Bairen Yi, et al. Lumine: An open recipe for building generalist agents in 3D open worlds. *arXiv preprint arXiv:2511.08892*, 2025.

[55] Jiaming Tang, Yufei Sun, Yilong Zhao, Shang Yang, Yujun Lin, Zhuoyang Zhang, James Hou, Yao Lu, Zhijian Liu, and Song Han. VLASH: Real-time VLAs via future-state-aware asynchronous inference. *arXiv preprint arXiv:2512.01031*, 2025.

[56] Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, et al. JARVIS-1: Open-world multi-task agents with memory-augmented multimodal language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[57] Zihao Wang, Shaofei Cai, Zhancun Mu, Haowei Lin, Ceyao Zhang, Xuejie Liu, Qing Li, Anji Liu, Xiaojian Ma, and Yitao Liang. OmniJARVIS: Unified vision-language-action tokenization enables open-world instruction following agents. *Advances in Neural Information Processing Systems*, 2024.

[58] Zihao Wang, Muyao Li, Kaichen He, Xiangyu Wang, Zhancun Mu, Anji Liu, and Yitao Liang. OpenHA: A series of open-source hierarchical agentic models in Minecraft. *arXiv preprint arXiv:2509.13347*, 2025.

[59] Zihao Wang, Xujing Li, Yining Ye, Junjie Fang, Haoming Wang, Longxiang Liu, Shihao Liang, Junting Lu, Zhiyong Wu, Jiazhan Feng, et al. Game-TARS: Pretrained foundation models for scalable generalist multimodal game agents. *arXiv preprint arXiv:2510.23691*, 2025.

[60] Zihua Wang, Zhitao Lin, Ruibo Li, Yu Zhang, Xu Yang, Siya Mi, and Xiu-Shen Wei. Open-loop planning, closed-loop verification: Speculative verification for VLA. *arXiv preprint arXiv:2604.02965*, 2026.

[61] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

[62] Marek Wydmuch, Michał Kempka, and Wojciech Jaśkowski. ViZDoom competitions: Playing Doom from pixels. *IEEE Transactions on Games*, 11(3):248–259, 2018.

[63] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[64] Jiabo Ye, Xi Zhang, Haiyang Xu, Haowei Liu, Junyang Wang, Zhaoqing Zhu, Ziwei Zheng, Feiyu Gao, Junjie Cao, Zhengxi Lu, et al. Mobile-Agent-v3: Fundamental agents for GUI automation. *arXiv preprint arXiv:2508.15144*, 2025.

[65] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[66] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

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

[68] Yuguang Yue, Irakli Salia, Samuel Hunt, Chris Green, Wenzhe Shi, and Jonathan J. Hunt. P2P: Scaling behavior cloning improves causal reasoning: An open model for real-time video game playing. *arXiv preprint arXiv:2601.04575*, 2026.

[69] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[70] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[71] Zhonghan Zhao, Wenwei Zhang, Haian Huang, Kuikun Liu, Jianfei Gao, Gaoang Wang, and Kai Chen. RIG: Synergizing reasoning and imagination in end-to-end generalist policy. *arXiv preprint arXiv:2503.24388*, 2025.

[72] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

[73] Enshen Zhou, Yiran Qin, Zhenfei Yin, Yuzhou Huang, Ruimao Zhang, Lu Sheng, Yu Qiao, and Jing Shao. MineDreamer: Learning to follow instructions via chain-of-imagination for simulated-world control. *arXiv preprint arXiv:2403.12037*, 2024.

[74] 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. *arXiv preprint arXiv:2506.06072*, 2025.

[75] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Agent Context Representation

Create green terracotta



Action: mouseMove(31,-2) and keyPress('left.shift')
Action: mouseMove(5,-3) and keyPress('left.shift')
Dream: The cursor moves to hover over the first icon in the craft bar <point>260 413</point>, and the \"Green Terracotta\" tooltip displays next to the cursor at <point>326 389</point>.
Action: mouseMove(8,-2) and keyPress('left.shift')
Action: mouseMove(12,4) and keyPress('left.shift')
Action: mouseMove(1,0) and keyPress('left.shift')
Action: mouseMove(1,0) and keyPress('left.shift')
Action: mouseMove(2,-1) and keyPress('left.shift')

Action: mouseClick('left') and keyPress('left.shift')
Dream: The cursor remains hovering over the first icon in the craft bar at <point>283 410</point>, and a green_terracotta appears in the result slot at <point>689,372</point>.
Action: keyPress('left.shift')
Action: no_op

Action: mouseMove(30, 0)
Action: mouseMove(166, 0) and keyPress('left.shift')

Mine spruce_leaves blocks



Action: mouseMove(-29, 10)
Action: mouseMove(-18, 10)
Action: mouseMove(-22, 9)
Dream: The camera shifts left and slightly downward, moving the crosshair toward the left-side oak leaves <point>498 495</point>.
Action: mouseMove(-4, 2)
Action: no_op
Action: mouseClick('left')

Action: mouseClick('left')

Action: mouseClick('left')
Action: mouseClick('left')
Action: mouseClick('left')
Action: mouseClick('left')
Action: mouseClick('left')
Dream: The stone axe completes the chopping motion, and the oak leaves around <point>500 500</point> disappear.
Action: mouseClick('left')

Hunt arctic animals



Action: mouseMove(24, 29)
Action: mouseMove(64, 41)
Action: mouseMove(24, 17)
Dream: The view shifts right and down, bringing the polar bear closer to the center of the screen at <point>453 461</point>.
Action: mouseMove(8, 11)
Action: mouseMove(11, 20)
Action: no_op
Action: no_op
Action: mouseMove(-11, 12)
Action: mouseMove(10, 9)
Action: keyPress('w')

Action: keyPress('w')

Action: keyPress('w')
Action: keyPress('w')
Action: mouseMove(-3, 1) and keyPress('w')
Action: mouseMove(-24, 5) and keyPress('w')
Action: mouseMove(-3, 0) and keyPress('w')

B Problem Formulation Details

We provide the fuller formulation behind Section 2. Single-step control predicts one atomic action from the history state,

$$\pi(a_t \mid s_t), \qquad o_1, a_1, o_2, a_2, \ldots, o_T, a_T. \tag{11}$$

Because the agent receives a fresh observation after every action, this closed-loop formulation gives frequent feedback but requires one model inference per atomic step and can encourage imitation models to copy recent action regularities rather than infer the causal state [16, 51].

Fixed-horizon multi-action prediction replaces the one-step target with a chunk

$$C_t^{(k)} \;=\; (a_t, a_{t+1}, \ldots, a_{t+k-1}), \qquad \pi(C_t^{(k)} \mid s_t), \tag{12}$$

which yields an execution sequence

$$o_1, C_1, o_{k+1}, C_2, \ldots, \qquad C_i = (a_{(i-1)k+1}, \ldots, a_{ik}). \tag{13}$$

The k actions are generated in one inference call and executed consecutively before the next observation. This amortizes inference and reduces step-to-step copying, but the agent is open-loop inside the chunk.

Adaptive MAP makes this temporal horizon state-dependent. One equivalent view is to predict a length at each chunk boundary,

$$k_i = g_\theta(s_{t_i}) \qquad \text{or} \qquad k_i \sim \pi_\theta^k(\cdot \mid s_{t_i}), \quad k_i \in \{1, \ldots, K\}, \tag{14}$$

then emit

$$C_i = (a_{t_i}, a_{t_i+1}, \ldots, a_{t_i+k_i-1}) \sim \pi_\theta^C(\cdot \mid s_{t_i}, k_i), \tag{15}$$

so that decision times follow

$$t_{i+1} = t_i + k_i, \qquad o_{t_1}, C_1, o_{t_2}, C_2, \ldots. \tag{16}$$

For an autoregressive agent, the same idea can be implemented as a stop decision inside the generated chunk [53]:

$$p_{i,j}^{\text{stop}} = P_\theta(\text{EOS} = 1 \mid s_{t_i}, a_{t_i:j}), \qquad k_i = 1 + \min\{j - t_i \;:\; p_{i,j}^{\text{stop}} \geq \tau\}. \tag{17}$$

The target horizon is the longest reliable commitment whose actions remain coherent under the current observation:

$$k_i^\star = \arg\max_{k \in \{1,\ldots,K\}} \mathcal{U}(s_{t_i}, k), \tag{18}$$

where $\mathcal{U}$ captures task progress, inference amortization, prediction risk, and semantic mismatch.

C Observation-Gap Warmup Details

Algorithm 1 gives the implementation-level procedure used to construct the adaptive-horizon warmup dataset from expert trajectories [2, 17, 37]. The procedure supplies initial observation-gap chunk boundaries; Interleaved Grounded Visual Dreaming and MAP-PPO later refine the policy beyond this offline initialization. At each boundary t, we search over candidate horizons $k \in \{1, \ldots, 16\}$, truncated by the remaining trajectory length. For each candidate, we compare the next-action distribution induced by open-loop continuation from s_t with the closed-loop one-step distribution after refreshing the state with the realized observation o_{t+k}. The open-loop distribution is

$$p_k^{\text{ol}}(a \mid s_t) := \pi_0(a \mid s_t, \mathbf{a}^*_{t:t+k-1}),$$

and the closed-loop distribution is

$$p_k^{\text{cl}}(a \mid s_{t+k}) := \pi_0(a \mid s_{t+k}), \qquad s_{t+k} = f(o_{t+k-L:t+k}, a^*_{t+k-L:t+k-1}).$$

We define the observation gap as

$$g_k(s_t) := D_{\text{KL}}\big(p_k^{\text{cl}}(\cdot \mid s_{t+k}) \;\big\|\; p_k^{\text{ol}}(\cdot \mid s_t)\big)\,.$$

A large g_k means that seeing the intermediate observation would substantially change the next-action belief, so the open-loop chunk should stop before relying on that stale continuation. If no candidate exceeds the threshold τ_{gap}, we use the longest available horizon. If multiple horizons share the same largest gap, we choose the earliest boundary.

Algorithm 1 Observation-Gap Segmentation of an Expert Trajectory

Require: Expert trajectory τ, model π_0, maximum horizon $K = 16$, threshold τ_{gap}
Ensure: Adaptive-horizon chunk dataset $\mathcal{D}_{\text{adapt}}$

$t \leftarrow 0$
while $t < |\tau|$ **do**
  $K_t \leftarrow \min(K, |\tau| - t)$
  for $k \leftarrow 1$ **to** $K_t - 1$ **do**
    $p_k^{\text{ol}} \leftarrow \pi_0(\cdot \mid s_t, \mathbf{a}^*_{t:t+k-1})$
    $s_{t+k} \leftarrow f(o_{t+k-L:t+k}, a^*_{t+k-L:t+k-1})$
    $p_k^{\text{cl}} \leftarrow \pi_0(\cdot \mid s_{t+k})$
    $g_k \leftarrow D_{\text{KL}}(p_k^{\text{cl}} \,\|\, p_k^{\text{ol}})$
  end for
  if $K_t = 1$ **then**
    $k \leftarrow 1$
  else
    $k^* \leftarrow \min \arg\max_{1 \le j < K_t} g_j$
    if $g_{k^*} > \tau_{\text{gap}}$ **then**
      $k \leftarrow k^*$
    else
      $k \leftarrow K_t$
    end if
  end if
  Add target $(s_t, (a_t^*, \ldots, a^*_{t+k-1}, \textsc{Stop}))$ to $\mathcal{D}_{\text{adapt}}$
  $t \leftarrow t + k$
end while

D Rubric-Based Dream Data Construction

The dream training data used in §3.2 is constructed through an offline, three-stage pipeline that ensures each dream reflects genuine causal prediction rather than paraphrasing of the future observation. The key design principle is **information isolation**: future frames are allowed only for supervision extraction and verification. The dream generator never sees the ground-truth future state, future-frame caption, or extracted answer; it must predict from the current observation, the executed action span, and a rubric describing which visual changes are worth predicting.

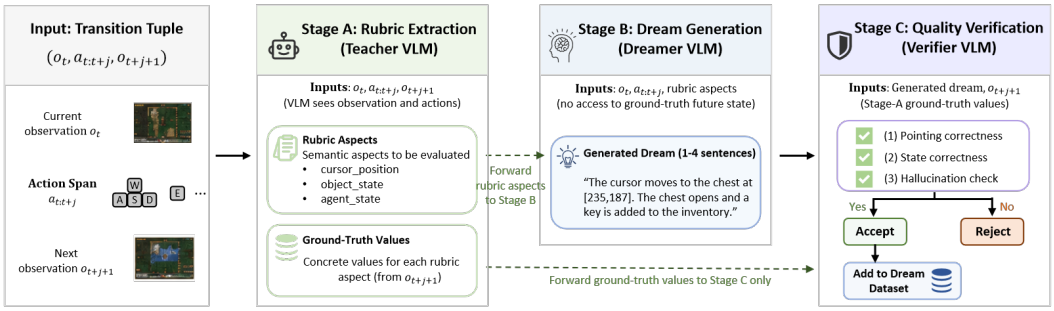


Figure 6: **Rubric-based dream data construction.** A teacher VLM observes both the current and future frames and extracts verifiable changed-state rubrics together with their ground-truth values. A dreamer model receives only the current observation, the action span, and the rubric aspects, and predicts the resulting visual change without access to the future frame. A verifier compares the candidate dream against the held-out ground truth along grounded criteria (pointing, state, hallucination, naturalness) and retains only verified predictions for training.

D.1 Pipeline Overview

Given a transition tuple $(o_t, \mathbf{a}_{t:t+j}, o_{t+j+1})$ from general game trajectory data, the pipeline proceeds as follows (Figure 6):

Stage A: Rubric Extraction. A teacher VLM [11] observes both o_t and o_{t+j+1}, computes the state delta, and produces two outputs: (1) a list of **rubric aspects**—the dimensions of environmental change most relevant to the executed actions (e.g., "cursor position change", "target object state transition")—and (2) the corresponding **ground-truth values** extracted from o_{t+j+1}. Only the rubric aspects are forwarded to Stage B as hints. The ground-truth values and the future observation are held out from the dream generator and forwarded *only* to Stage C for verification.

Stage B: Dream Generation. A dreamer model receives o_t, the action span $\mathbf{a}_{t:t+j}$, and the rubric aspects. It does not receive o_{t+j+1}, the Stage-A ground-truth values, or any textual description produced from o_{t+j+1}. It generates a candidate dream: 1–4 sentences of anchored natural language describing the expected environmental consequence, optionally embedding spatial anchors such as $[x, y]$ coordinates. The model is performing genuine causal imagination, not reformulating a known answer.

Stage C: Quality Filtering. A verification stage compares the candidate dream against the Stage-A ground truth along four grounded criteria [20, 52]:

1. **Pointing correctness**: any spatial anchors in the dream are compared against actual object positions in o_{t+j+1} (within a tolerance).
2. **State correctness**: player/object state descriptions are checked against the true state delta.
3. **Hallucination check**: the dream must not mention objects or state changes absent from both o_t and o_{t+j+1}.
4. **Naturalness check**: the dream must read as fluent natural language, not as a structured field dump.

Only samples passing all four checks enter the final training set. The retained training sequence contains the current context, actions, and accepted dream text, but not the future observation or Stage-A answer. At deployment, the trained model generates dreams autonomously; the rubric, teacher, verifier, and future frame are never used at inference time.

D.2 Prompt Templates

We provide the core prompts for each stage below. All prompts are instantiated with task-specific context (game name, current task instruction) at runtime.

Prompt D.1: Stage A: Rubric Extraction

```
You are given two consecutive game observations (Frame A and Frame B) and the actions
    executed between them.
1. Describe the state delta between Frame A and Frame B.
2. Identify 1-3 rubric aspects most relevant to the executed actions (e.g.,
    cursor_position, object_state, agent_state, ui_change).
3. For each aspect, extract the ground-truth value from Frame B, including spatial
    coordinates where applicable.
Output JSON: {"delta_state": "...", "rubric_aspects": [...], "ground_truth": {"aspect":
    "value", ...}}
```

Prompt D.2: Stage B prompt (Dream Generation).

```
You are a game agent that has just performed the following actions after observing the
    current frame.
Consider predicting the following aspects: [ASPECTS].
```

```
In 1-4 concise sentences, describe what you expect to see in the environment as a
    result of your actions.
Embed spatial anchors [x, y] where relevant.
Do NOT describe irrelevant background details.
```

Prompt D.3: Stage C prompt (Quality Verification).

```
Compare the following dream prediction against the actual next observation and
    ground-truth annotations.
Check: (1) pointing accuracy, (2) state correctness, (3) hallucination, (4) naturalness.
Output JSON: {"accept": true/false, "analysis": "...", "dream_text": "<corrected dream
    if minor fix needed>"}
```

D.3 Dream Selection Criteria

Not every intra-chunk step requires a dream. We prioritize steps where the action produces a visually verifiable state change (e.g., clicking an item, rotating the camera toward a target, opening a UI panel) and omit steps with negligible environmental effect (e.g., repeated forward movement with no scene change). The first step of each chunk is excluded (the agent has a real observation) and so is the step after the chunk ends (covered by the next observation).

E Evaluation and Implementation Details

Minecraft evaluation follows the OpenHA protocol [58] on MineRL-based environments [26]. We group tasks into Embodied, GUI, and Combat categories following the semantic structure used in the benchmark, and evaluate each task on novel procedurally generated seeds. For each method, we run three independent rollouts per task and report best-of-3 success averaged within each category.

VizDoom evaluation uses the official DefendLine, TakeCover, and HealthGathering scenarios [62]. For DefendLine, we use `frame_skip=4` and limit each episode to 400 agent steps. Imitation-learning results are evaluated zero-shot because data from these evaluation scenarios is excluded from the pretraining and SFT corpus; RL results use environment-specific online fine-tuning with the same MAP-PPO formulation described in Section 3.3. For all baselines, the underlying model checkpoint and observation/action interface are held fixed, and only the horizon strategy differs.

F Chunk-Level GAE Derivation and RL Hyperparameters

F.1 Telescoping Derivation

We provide the full derivation of the chunk-level GAE used in Section 3.3. Throughout, we index chunks by j, write s_j for the state at the j-th chunk boundary, C_j for the chunk emitted there [36], k_j for its horizon, and adopt the shorthands

$$R_j = \sum_{i=0}^{k_j-1} \gamma^i \, r_{t_j+i}, \qquad \Gamma_j = \gamma^{k_j}.$$

Chunk-boundary TD residual. The state-value critic $V_\phi(s_j)$ estimates the expected return before generating the current chunk. The chunk-boundary TD residual is

$$\delta_j \;:=\; R_j \;+\; \Gamma_j \, V_\phi(s_{j+1}) \;-\; V_\phi(s_j). \tag{19}$$

n-chunk advantage. The n-chunk advantage estimate is the return over chunks $j, \ldots, j+n-1$, bootstrapped with $V_\phi(s_{j+n})$, minus the pre-chunk baseline $V_\phi(s_j)$:

$$\hat{A}_j^{(n)} = \sum_{m=0}^{n-1} \left(\prod_{i=0}^{m-1} \Gamma_{j+i} \right) R_{j+m} \;+\; \left(\prod_{i=0}^{n-1} \Gamma_{j+i} \right) V_\phi(s_{j+n}) \;-\; V_\phi(s_j). \tag{20}$$

When $n=1$ this reduces to $\hat{A}_j^{(1)} = \delta_j$.

Telescoping. We show that the V_ϕ-terms cancel telescopically, leaving only TD residuals. From the definition of δ_{j+m} (Eq. 19), we can write

$$R_{j+m} = \delta_{j+m} - \Gamma_{j+m}\, V_\phi(s_{j+m+1}) + V_\phi(s_{j+m}).$$

Substituting into Eq. (20) and collecting terms, the intermediate V_ϕ values cancel pairwise, yielding

$$\hat{A}_j^{(n)} \;=\; \sum_{m=0}^{n-1} \left(\prod_{i=0}^{m-1} \Gamma_{j+i} \right) \delta_{j+m}. \tag{21}$$

(1–λ)-weighted mixture. The chunk-level GAE is the exponentially-weighted average of n-chunk advantages:

$$\hat{A}_j(\gamma, \lambda) \;=\; (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1}\, \hat{A}_j^{(n)}.$$

Substituting Eq. (21) and exchanging the order of summation (for a fixed m, the inner sum over n from $m+1$ to ∞ evaluates to $(1-\lambda)\lambda^{m-1} \cdot \frac{\lambda}{1-\lambda} = \lambda^m$), we obtain the closed form

$$\boxed{\hat{A}_j(\gamma, \lambda) \;=\; \sum_{m=0}^{\infty} \lambda^m \left(\prod_{i=0}^{m-1} \Gamma_{j+i} \right) \delta_{j+m}.} \tag{22}$$

Relation to classical GAE. In the special case where every chunk has unit length ($k_j = 1$ for all j), we have $\Gamma_j = \gamma$ for all j, and Eq. (22) reduces to $\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}$, recovering the original GAE of Schulman et al. [47]. For variable-length chunks the per-step discount γ is replaced by the chunk-dependent $\Gamma_{j+i} = \gamma^{k_{j+i}}$, so the horizon chosen by the agent modulates credit assignment automatically through the discount product.

Per-chunk normalization. Because longer chunks aggregate more atomic-step rewards into R_j, the raw advantage $\hat{A}_j$ tends to grow with k_j. To prevent the PPO objective from systematically favoring long horizons (whose raw advantages are larger in magnitude), we normalize by the horizon before computing the policy gradient: $\bar{A}_j := \hat{A}_j / k_j$.

Practical truncation. In practice, trajectories are finite. At a terminal state we set the bootstrap $V_\phi(s_{j+1}) = 0$; at a non-terminal trajectory cutoff we bootstrap with the critic's current estimate, following the standard infinite-horizon RL convention [48]. The infinite sum in Eq. (22) is truncated at the trajectory boundary.

F.2 RL Hyperparameters

Table 4 lists all hyperparameters used by the MAP-PPO [48] loop in Section 3.3, grouped by the role they play in the formalism. We deliberately set $\gamma = 1.0$. KL penalty regularization [75] is applied only as a per-token reward shaping term ($\beta_{\text{KL}} = 10^{-5}$, with reference policy π_{warm}), and not as a separate loss (Section 3.3). The asymmetric clip $(\epsilon_{\text{low}}, \epsilon_{\text{high}}) = (0.2, 0.3)$ follows DAPO [66] and gives the chunk-level ratio extra room to expand in the improving direction.

The bottom group shows our semi-asynchronous rollout scheduler, which overlaps rollout collection with optimization to avoid stalls caused by heterogeneous chunk lengths. An update is triggered once `complete_ratio` = 0.7 of the target rollout batch has finished, rather than waiting for the slowest environments. To limit the policy lag introduced by this asynchronous schedule, `rollout_agent_offpolicy_step_th` = 2 filters out samples collected by actors that are more than two optimization steps behind the current policy. This keeps the chunk-level importance ratio $\rho_j(\theta)$ in a range where PPO clipping remains effective. During best-of-N collection, `bon_strategy` = `all` retains all sampled trajectories for PPO, preserving the empirical distribution of agent-selected horizons instead of training only on a selected top-k subset.

Table 4: **Chunk-level PPO and AHA hyperparameters.** See Sections 3.3 and 3.4.

Group	Symbol	Config key	Value
Reward shaping	β	dream_reward_coef	0.1
	β_{KL}	kl_coef	1×10^{-5}
Chunk-level GAE	γ	gae_gamma	1.0
	λ	gae_lam	0.95
PPO objective	ϵ_{low}	clip_ratio_low	0.2
	ϵ_{high}	clip_ratio_high	0.3
	–	kl_loss_weight	0.0
AHA	y	max_iterations	111
	l	window_size	100
	p	quantile	0.9
	a	action_chunk_len_lower_bound	4
Semi-async rollout	–	complete_ratio	0.7
	–	rollout_agent_offpolicy_step_th	2
	–	bon_strategy	`all`

G Cross-Domain Evaluation

To evaluate whether adaptive temporal horizons transfer beyond gameplay to continuous robotic manipulation, we conduct a cross-domain evaluation on **Language-Table** [38], a language-conditioned tabletop manipulation benchmark in which a robot arm pushes blocks to target configurations from visual observations and natural-language instructions. Under an identical compute budget, we fine-tune both single-action ReAct and AdaMAP on the same publicly released 5B-token dataset and evaluate them across all five official task categories.

As shown in Table 5, AdaMAP improves success rates in four out of five categories and raises overall success from 55.00% to 62.00%. The largest gain appears on *Block-to-Relative-Location*, where the required pushing horizon varies substantially with the initial spatial arrangement of the blocks. Although AdaMAP is slightly lower on the already near-saturated *Separate* category, its consistent improvements on precision-demanding spatial tasks confirm that learned state-dependent horizons generalize effectively to embodied manipulation dynamics.

Table 5: **Cross-domain evaluation on Language-Table.** Success rates (%) are reported as mean $\pm$ standard deviation across evaluation rollouts for all five official task categories.

Method	Block-to-Block	Block-to-Abs.	Block-to-Block-Rel.	Block-to-Rel.	Separate	Overall
ReAct	$45.00^{\pm21.21}$	$35.00^{\pm7.07}$	$30.00^{\pm28.28}$	$45.00^{\pm7.07}$	$\mathbf{100.00}^{\pm0.00}$	$55.00^{\pm2.08}$
AdaMAP	$\mathbf{55.00}^{\pm7.07}$	$\mathbf{45.00}^{\pm7.07}$	$\mathbf{40.00}^{\pm0.00}$	$\mathbf{80.00}^{\pm14.14}$	$90.00^{\pm0.00}$	$\mathbf{62.00}^{\pm2.83}$

H Backbone-Capacity Ablation

Interleaved Grounded Visual Dreaming and adaptive stopping rely on the underlying VLM's multimodal perception and spatial grounding capabilities. To examine how AdaMAP scales with model capacity, we compare our default 2.5B-activated-parameter MoE backbone against a larger 7B-activated-parameter variant while keeping the AdaMAP training pipeline and evaluation protocol unchanged.

Table 6 reports zero-shot generalization results across the three VizDoom scenarios. Scaling the backbone from 2.5B to 7B activated parameters consistently improves episode returns across all scenarios: DefendLine increases from 10.67 to 14.00, TakeCover from 230.67 to 244.33, and HealthGathering from 1214.67 to 1514.67, accompanied by reduced run-to-run variance on HealthGathering. These results indicate that AdaMAP benefits directly from stronger visual-language representations without requiring architectural modifications.

Table 6: **Backbone-capacity ablation on VizDoom.** Zero-shot episode returns (mean $\pm$ standard deviation) when scaling the activated parameters of the VLM backbone from 2.5B to 7B.

Backbone Capacity	**DefendLine (DL)**	**TakeCover (TC)**	**HealthGathering (HG)**
ADAMAP (2.5B activated)	$10.67^{\pm 6.43}$	$230.67^{\pm 15.14}$	$1214.67^{\pm 393.04}$
ADAMAP (7B activated)	$\mathbf{14.00}^{\pm 7.00}$	$\mathbf{244.33}^{\pm 17.85}$	$\mathbf{1514.67}^{\pm 224.76}$

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the proposed adaptive MAP framework, its components, and the empirical scope of the claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The conclusion includes a limitations paragraph discussing the current evaluation scope and model-family assumptions.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The problem formulation states the policy and horizon assumptions, and Appendix F.1 provides the derivation used by MAP-PPO.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Experiments.

   Guidelines:

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

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

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

Answer: [No]

Justification: No.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer:[Yes]

   Justification: Experiments.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Experiments.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [No]

   Justification: No.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research presented in this paper conforms to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer:[No]

    Justification: No.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [Yes]

    Justification: Experiments.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Citation.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Formulation.

    Guidelines:

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

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

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

    Answer: [Yes]

    Justification: N/A.

    Guidelines:

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

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

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

    Answer: [Yes]

    Justification: N/A.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer:[Yes]

Justification: N/A.

Guidelines:

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