AtlasVA: Self-Evolving Visual Skill Memory for Teacher-Free VLM Agents

Pan Wang[*,1,2], Yihao Hu[*,1,3], Xiujin Liu[4], Jingchu Yang[1], Hang Wang[5], Zhihao Wen[✉,1]
[*] Equal Contribution ✉ Corresponding Author (z.wen@antgroup.com)
[1] Ant Group [2] University of Science and Technology of China [3] Westlake University
[4] University of Michigan - Ann Arbor [5] Sun Yat-sen University

Abstract

Vision-language model (VLM) agents increasingly rely on memory-augmented reinforcement learning to reuse experience across long-horizon tasks, yet most existing frameworks store memory as text and depend on proprietary teacher models to summarize or refine it. This design is poorly matched to spatial decision making: geometric priors are compressed into lossy language, and sparse interaction is often supervised through delayed textual feedback rather than dense visually grounded signals. We argue that reusable experience for VLM agents should remain visually grounded. Based on this insight, we propose **AtlasVA**, a teacher-free visual skill memory framework that organizes memory into three complementary layers: spatial heatmaps, visual exemplars, and symbolic text skills. AtlasVA further evolves danger and affinity atlases directly from trajectory statistics and lightweight grid heuristics, and reuses these self-evolving atlases as potential-based shaping rewards for reinforcement learning. This unifies perception, memory, and optimization without external LLM supervision. Experiments on SOKOBAN, FROZENLAKE, 3D embodied navigation, and 3D robotic manipulation benchmarks show that AtlasVA consistently outperforms text-centric memory baselines and competitive VLM agents, with especially strong gains on spatially intensive tasks.

Homepage: https://wangpan-ustc.github.io/AtlasvaWeb/

1 Introduction

Vision-language models (VLMs) are becoming a practical interface for interactive agents that must read instructions, parse screenshots, and execute grounded actions in sequential environments [14, 24, 26, 28, 47, 57]. This setting is especially challenging in long-horizon spatial tasks, where the agent must accumulate reusable experience rather than reason from scratch at every step. Recent memory-augmented reinforcement learning (RL) systems address this need by storing retrieved skills, retrospectives, and task heuristics across episodes [36, 37, 49, 54]. However, most of these systems still treat the VLM agent as a text-centric reasoner: the visual stream is consumed only as a transient observation, while reusable knowledge is stored almost entirely in language [11, 20].

Recent memory-centric frameworks show that explicit skill libraries can improve exploration and reuse past experience, with representative examples such as SkillRL [48] and XSkill [16]. Their common recipe is to summarize successful or failed trajectories into text, retrieve the relevant summaries for a new state, and then rely on a strong teacher model to refine the memory over time. This recipe is reasonable for language-only agents, but it is poorly matched to VLM agents that solve tasks through spatial perception [1, 23, 25]. Compressing a two-dimensional layout into one-dimensional textual rules discards geometric structure, turning visually grounded decision making into a lossy translation problem.

Under this text-centric design, VLM agents face unique challenges when interacting with complex environments, as illustrated in Figure 1 (top). Primarily, ❶ *encoding high-value spatial cues into text inevitably incurs severe information*

loss, as text cannot capture rich visual details.

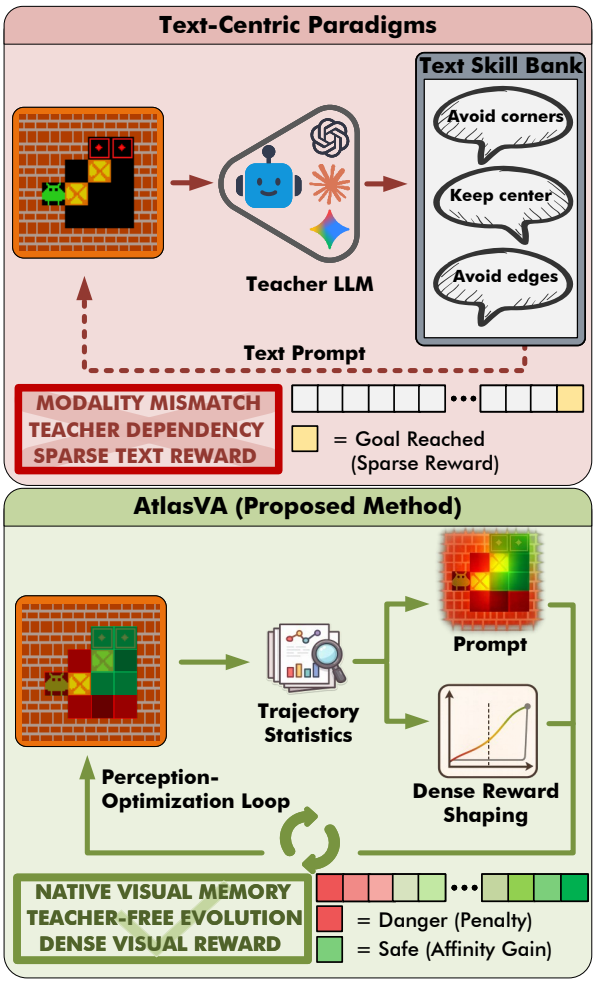

Figure 1 Comparison between text-centric memory paradigms and **AtlasVA**.

For example, text struggles to convey complex spatial topologies, such as dead ends, local hazards, and promising sub-goal regions. Furthermore, to manage these textual records, ❷ *many frameworks require a proprietary LLM to summarize failures, merge skills, or rewrite memory.* This dependence not only increases computational costs but also undermines the premise of fully autonomous self-improvement. Finally, using textual rewards to evaluate spatial interactions creates a ❸ *feedback mismatch for RL*. Because textual critiques are inherently abstract and lack precise spatial grounding, they deprive the agent of dense, coordinate-aware guidance. This exacerbates the credit assignment problem, underscoring the critical need to align the reward format with the spatial nature of the task.

In this paper, we propose **AtlasVA**, which transitions agent memory from a text-only repository to a *Visual Skill Memory (VSM)* hierarchy (Figure 1, bottom). **First**, VSM stores reusable experience across three complementary layers: spatial heatmaps, visual exemplars, and symbolic text to address ❶. During forward inference, these heatmaps act as visual prompts that project complex 3D environments into 2.5D spatial maps, effectively maintaining sensitive spatial knowledge within the VLM's native modality. **Second**, due to ❷, AtlasVA bootstraps memory directly from raw interaction history. Instead of using external models for textual summarization, it aggregates trajectory statistics to construct self-evolving spatial heatmaps, thereby reducing computational overhead and ensuring fully autonomous self-improvement. **Third**, to resolve ❸, these same heatmaps are mathematically formulated as a potential function for policy updates. By rewarding movement toward historically successful regions and penalizing identified deadlocks, this mechanism provides the dense, coordinate-aware guidance necessary to alleviate credit assignment difficulties.

Overall, our main contributions are as follows:

- We propose **AtlasVA**, a memory framework for VLM agents that replaces lossy text-only storage with a **Visual Skill Memory** hierarchy. By preserving spatial heatmaps, visual exemplars, and symbolic text, it natively aligns reusable experience with the agent's visual perception.
- We introduce a **teacher-free atlas evolution** mechanism to sustain this memory. This mechanism refines danger and affinity heatmaps directly from raw trajectory statistics, removing the need for proprietary LLM supervision.
- We develop a **dense visual reward shaping** strategy that repurposes the evolving visual priors as potential functions. This provides dense, coordinate-aligned gradients that mitigate the severe sparse-reward bottleneck in both 2D and 3D tasks, yielding substantial gains in both learning efficiency and final task performance.

2 Related Work

2.1 Vision-Language Models as Autonomous Agents

Large vision-language models (VLMs) have emerged as capable autonomous agents for sequential decision-making in interactive environments [19, 31]. By combining visual grounding with instruction following, VLM agents achieve

strong performance in domains such as web navigation, GUI control, and embodied planning [5, 20, 22, 44]. However, most current architectures treat visual observations merely as transient, step-level inputs [33]. To maintain historical context or long-horizon heuristics, these systems typically compress past experiences into textual representations, such as verbal summaries [36], scratchpads [53], or retrieved passages [4, 34]. This reliance on text creates an architectural asymmetry when operating in highly spatial environments, as verbal rules struggle to preserve fine-grained geometric layouts and topological constraints. In contrast, AtlasVA eliminates this text bottleneck by storing and updating spatial structures directly within the visual modality, aligning the agent's memory with its native perception.

2.2 Memory and Skill Evolution in RL Agents

To enable iterative self-improvement, recent reinforcement learning (RL) frameworks increasingly equip agents with external memory and skill evolution mechanisms. For instance, Reflexion [36] uses verbal self-critique to revise plans, while frameworks like ExpeL [55] and Mem0 [9] distill trajectories into retrievable linguistic records. Similarly, skill-augmented pipelines such as SkillRL [48] and XSkill [16] accelerate learning by constructing libraries of natural-language skills or Markdown workflows from interaction logs. These approaches, however, share two critical limitations: a strict reliance on text-based representations and strong teacher dependence. Even visually grounded frameworks like XSkill store experiences as Markdown or JSON, which inherently struggle to capture fine-grained spatial hazards, topological dead ends, and geometric traps [8, 10, 45, 50]. Furthermore, generating and refining these textual rules requires repeated calls to powerful, proprietary large language models, driving up API costs and limiting the agent's true autonomy [42]. AtlasVA circumvents these bottlenecks by shifting online adaptation entirely to *visual* atlases, which evolve directly from trajectory statistics without requiring external teacher supervision.

2.3 Reward Shaping and Visual vs. Textual Feedback

Sparse terminal rewards remain a fundamental bottleneck in RL, often leading to extreme sample inefficiency in spatial reasoning tasks [12, 18]. To address this in VLM agents, recent methods prompt external LLMs with action logs to generate "textual rewards" (e.g., "you should not have pushed the box to the corner") [32, 43, 46, 52, 56]. More formally, potential-based reward shaping (PBRS) [27] offers a rigorous solution by adding the difference of a potential function Φ to the reward. However, both paradigms face significant challenges in visual domains. Textual critiques are abstract and delayed, failing to translate into the dense, coordinate-aligned feedback necessary for spatial optimization, leaving agents to rely on blind trial-and-error when stuck in topological traps. Meanwhile, defining effective PBRS potentials typically requires heavy manual engineering (e.g., hardcoded distance heuristics or static penalty zones) that fails to scale to diverse or dynamic layouts. Standard exploration bonuses like count-based visitation [7] also struggle to scale directly to pixel-level visual inputs. AtlasVA bridges this gap by introducing a **self-evolving visual atlas** that acts as an automatically learned potential function. By combining grid cues with trajectory statistics, it converts spatial danger and affinity maps directly into dense, coordinate-aligned shaping rewards, automating reward engineering while maintaining precise spatial gradients at every step.

3 AtlasVA

3.1 Problem Formulation

We formulate the embodied decision-making process as a partially observable Markov decision process $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$. At each timestep t, the agent receives a multimodal observation $o_t \in \mathcal{O}$ comprising a rendered RGB frame and a natural language task description. The agent emits a discrete action $a_t \in \mathcal{A}$ sampled from a multimodal policy $\pi_\theta(a_t \mid o_{\leq t})$, parameterized by a vision-language model. The environment transitions to a new hidden state s_{t+1} according to the transition dynamics $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ and yields a sparse binary reward $\mathcal{R}(s_t, a_t) \in \{0, 1\}$ upon task success. The primary challenge in this setting is that the optimal policy requires profound spatial reasoning and long-horizon planning, yet the environment only provides sparse, delayed feedback. Standard VLM agents struggle to extract persistent topological priors from transient observations. To address this, we define a unified *GridState* abstraction g_t extracted directly from the simulator's internal state to represent semantic layers, including obstacles and interactive objects [17]. Let $\mathbf{p}_t \in \mathbb{Z}^2$ denote the 2D coordinate of the primary manipulated entity at timestep t. AtlasVA leverages this privileged representation exclusively during training to construct and evolve persistent spatial priors across episodes; the deployed VLM policy relies solely on visual inputs (see Appendix C.5 for API details).

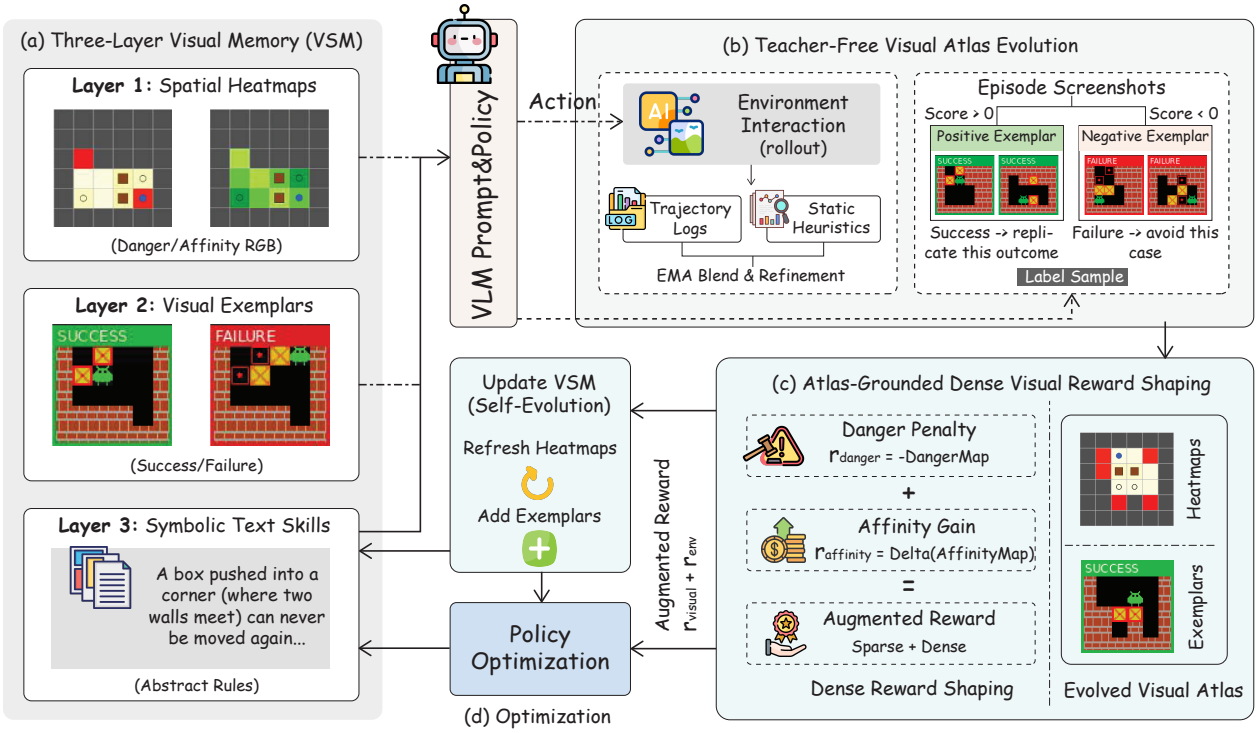

Figure 2 Overview of the AtlasVA architecture. (a) The Visual Skill Memory (VSM) prompts the policy using a three-layer hierarchy: spatial heatmaps, visual exemplars, and symbolic text rules. (b) Through teacher-free environment rollouts, trajectory logs and episode screenshots drive the self-evolution of the VSM via EMA blending and exemplar mining. (c) These evolved spatial priors are formulated into affinity gains and danger penalties, providing dense reward shaping. (d) In the optimization phase, the augmented rewards guide policy updates, while improved trajectories continuously refresh the VSM, forming a closed perception-optimization loop.

3.2 Three-Layer Visual Skill Memory (VSM)

(**Fig. 2a**) To rectify the modality mismatch of text-centric rules, we introduce a multimodal memory architecture comprising three complementary layers. The augmented multimodal prompt provided to the VLM is constructed as $\tilde{o}_t = [M_{danger}, M_{affinity}, \mathcal{E}_{vis}, \mathcal{S}_{text}, o_t]$.

Layer 1: Spatial Heatmaps. At the foundational level, we provide continuous spatial fields over the environment: a *danger map* indicating deadlock risks and an *affinity map* indicating proximity to task completion. Rather than encoding these as text arrays, we render them as RGB heatmaps $M \in \mathbb{R}^{H\times W\times 3}$, which natively align with the VLM visual encoder. These are injected as separate visual tokens or alpha-blended over the original observation o_t, allowing the agent to intuitively perceive hazardous and promising regions without modality translation.

Layer 2: Visual Exemplars. To complement these abstract spatial priors with concrete visual references, the second layer introduces visual exemplars ($\mathcal{E}_{vis}$) by mining a small bank of representative screenshots from historical rollouts, explicitly annotated with success or failure tags.

Layer 3: Symbolic Text Skills. Finally, to preserve high-level strategic reasoning, the third layer integrates symbolic text skills ($\mathcal{S}_{text}$), retaining a compact set of textual heuristics. Together, these components establish a complete knowledge gradient from perceptual spatial maps and concrete layout examples to symbolic logic, providing holistic, in-context guidance without the lossy translation inherent in purely text-based memory.

3.3 Teacher-Free Visual Atlas Evolution

(**Fig. 2b**) To eliminate reliance on external LLM teachers, we propose a purely data-driven forward process that bootstraps spatial priors directly from the agent's own interaction data. The spatial heatmaps in Layer 1 are synthesized by fusing static grid heuristics $M_{heuristic}$ with accumulated trajectory statistics M_{stat}.

Trajectory accumulation. First, for the static heuristics branch, we extract topological features from the current layout, such as corner-like deadlock regions and Breadth-First Search distances to target objects. Second, for the trajectory statistics branch, we analyze the structured rollout logs of the current training batch. Let $\mathcal{T}_{fail}$ and $\mathcal{T}_{succ}$ denote the set of failed and successful trajectories. We compute the batch-level danger map by accumulating the terminal failure positions $\mathbf{p}_T$:

$$M_{batch}^{danger}(\mathbf{p}) = \frac{1}{|\mathcal{T}_{fail}|} \sum_{\tau \in \mathcal{T}_{fail}} \mathbb{I}(\mathbf{p}_T = \mathbf{p}) \tag{1}$$

Similarly, the batch-level affinity map $M_{batch}^{affinity}$ is derived by recording the normalized visit frequency of coordinates along the successful paths in $\mathcal{T}_{succ}$.

EMA blending. Next, we update the historical trajectory statistics using an Exponential Moving Average (EMA) with decay rate α: $M_{stat} \leftarrow \alpha M_{stat} + (1-\alpha) M_{batch}$. Finally, the output heatmap presented to the VLM is a dynamic blend of the static heuristics and the EMA statistics:

$$M_{final} = (1-\beta_k) M_{heuristic} + \beta_k M_{stat} \tag{2}$$

where $\beta_k \in [0,1]$ is a scheduling coefficient that progressively anneals from 0 to 1 over training epochs k. This scheduling coefficient provides safe, cold-start exploration guided by static geometry in early stages, while smoothly transitioning to experience-driven refinement in later stages without risking catastrophic forgetting.

3.4 Atlas-Grounded Dense Visual Reward Shaping

(**Fig. 2c**) To alleviate extreme sample inefficiency in sparse reward settings, we design an automated reward shaping mechanism that utilizes our self-evolving spatial atlas as a dynamic potential function. Given a transition from $\mathbf{p}_t$ to $\mathbf{p}_{t+1}$, we compute a bounded auxiliary visual reward $r_{visual} = \lambda_{danger} \cdot r_{danger} + \lambda_{affinity} \cdot r_{affinity}$.

Danger penalty. First, the danger penalty r_{danger} discourages the agent from exploring regions that the atlas has historically identified as risky. We apply a negative penalty proportional to the danger value of the coordinate the primary entity enters: $r_{danger} = -M_{final}^{danger}(\mathbf{p}_{t+1})$.

Affinity gain. Second, the affinity gain $r_{affinity}$ incentivizes the agent to follow paths that move closer to the goal manifold. We formulate it as the per-step difference of the affinity potential:

$$r_{affinity} = M_{final}^{affinity}(\mathbf{p}_{t+1}) - M_{final}^{affinity}(\mathbf{p}_t) \tag{3}$$

Because $M_{final}^{affinity}$ inherits the BFS distance gradient from $M_{heuristic}^{affinity}$, this difference produces a strictly positive signal whenever the agent moves one step closer to the goal, and a strictly negative signal when it moves away.

3.5 Optimization and Closed Loop

(**Fig. 2d**) The final reward used for RL optimization is $r_{visual} + r_{env}$. While the affinity gain is formulated as a potential difference [27], the danger penalty acts as a heuristic safety constraint that intentionally alters the optimal policy to prioritize safe navigation over hazardous shortcuts. Furthermore, this mechanism closes the perception-optimization loop autonomously: improved policies generate higher-quality trajectories, which refine the visual atlas via EMA, which in turn provides more accurate dense rewards for further optimization (see Figure 3 and Appendix D.1 for details on projecting 3D continuous spaces into 2.5D visual priors).

To prevent modality mismatch when injecting RGB heatmaps, visual exemplars, and symbolic text back into the policy during this loop, AtlasVA establishes a unified, interleaved vision-language prompt interface. As summarized in Table 1, each VSM layer is assigned a distinct modality, update rule, and prompt anchor. We natively inject the spatial heatmaps (**Layer 1**) and episodic visual exemplars (**Layer 2**) as standalone visual tokens (⟨image⟩). These visual priors are hierarchically anchored alongside the symbolic text rules (**Layer 3**) as shown in the prompt skeleton (Appendix B). This native multimodal injection empowers the VLM to perform zero-shot spatial pattern matching directly in the pixel space. It cleanly bridges the gap between abstract strategic planning and visually-grounded execution, ensuring the VLM leverages its pre-trained visual encoders efficiently.

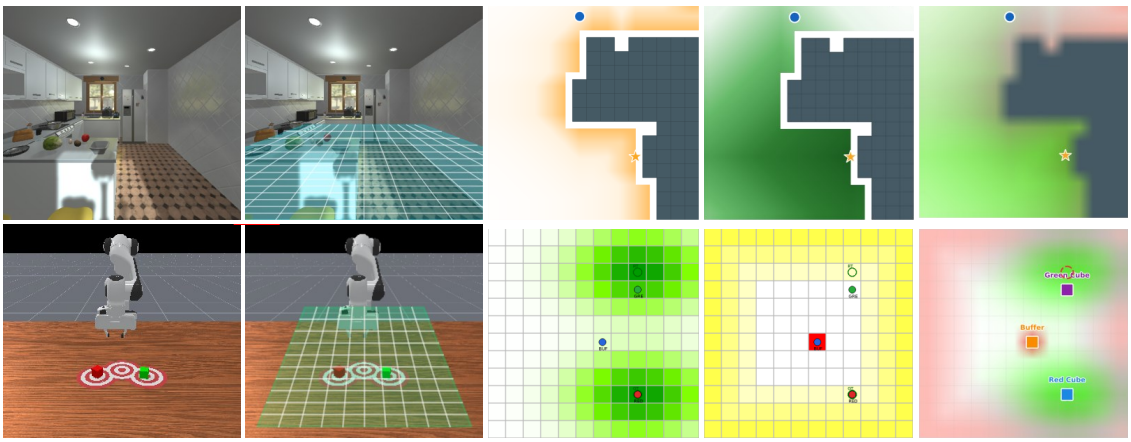

Figure 3 Projection of 3D continuous spaces into 2.5D visual priors. Top row: In Navigation, 3D rooms are mapped to 2D floor plans with obstacle-aware heatmaps. **Bottom row:** In PrimitiveSkill, tabletop workspaces are discretized into 2.5D grids, separating x-y planar guidance from continuous z-axis constraints.

Table 1 VSM injection specification. Each layer is assigned a distinct modality, update rule, and anchor position in the VLM prompt, forming a unified visual-textual memory interface.

Layer	Modality	Content	Update Rule	Prompt Anchor
L1 Heatmap	RGB image	Danger / Affinity map	EMA over trajectories	`## Spatial Skill Maps`
L2 Exemplar	RGB image(s)	Top-k success / failure frames	Retrieval + eviction (cap=6)	Before current observation
L3 Text	Text	Principles & mistakes	Per-episode summarization	`## Learned Principles`

4 Experiments

In this section, we evaluate AtlasVA across diverse 2D and 3D spatial benchmarks and investigate the following research questions:

RQ1: Does visual skill memory enable a compact 3B-parameter VLM agent to surpass significantly larger proprietary models on spatially intensive tasks? (Sec. 4.2)

RQ2: Does atlas-grounded dense reward shaping mitigate the sparse-reward bottleneck and accelerate policy convergence in long-horizon spatial tasks? (Sec. 4.2, Sec. 4.3)

RQ3: Can spatial heatmaps be effectively bootstrapped from raw trajectory statistics alone, without reliance on external teacher LLMs? (Sec. 4.3)

RQ4: How critical is each layer of the three-layer visual skill memory, and does native visual grounding provide clear advantages over text-only representations? (Sec. 4.3)

4.1 Experimental Setup

Base Model and Optimization. We adopt Qwen2.5-VL-3B-Instruct as the base vision-language model. AtlasVA is optimized via Proximal Policy Optimization (PPO) with Generalized Advantage Estimation (GAE). During the rollout phase, the agent executes multi-turn interactions with the environment. We employ a training batch size of 128, setting the learning rate to 1×10^{-6} and 1×10^{-5} for the actor and critic networks, respectively. Full optimization hyperparameters are listed in Appendix C.6.

Visual Skill System Configuration. The Three-Layer Visual Skill Memory provides continuous spatial grounding. To prevent information leakage, all memory components (heatmaps, exemplars, and text skills) are evolved exclusively using trajectories from the *training* environments, keeping validation sets strictly separated for zero-shot evaluation. Detailed configurations, including EMA decay rates, exemplar pool capacities, and memory pruning mechanics, are provided in Appendix C.6.

Datasets. We evaluate AtlasVA across four diverse agentic benchmarks: Classic Grid Puzzles (SOKOBAN [41] and

Model/Method	Sokoban	FrozenLake	Navigation			PrimitiveSkill					Overall
			Base	Common	Average	Place	Stack	Drawer	Align	Swap	
Proprietary Models											
GPT-5 [29]	0.70	0.77	0.75	0.81	0.78	1.00	0.63	0.00	1.00	0.55	0.69
o3 [30]	0.60	0.78	0.81	0.75	0.78	1.00	0.63	0.00	1.00	0.82	0.71
o4-mini [30]	0.44	0.82	0.75	0.75	0.75	1.00	0.50	0.00	0.75	0.33	0.60
GPT-4o [15]	0.43	0.54	0.75	0.69	0.72	0.50	0.63	0.00	0.88	0.94	0.60
Gemini 2.5 flash [13]	0.52	0.57	0.58	0.56	0.57	0.75	0.50	0.00	0.88	0.82	0.58
Gemini 2.5 Pro [13]	0.58	0.78	0.63	0.63	0.63	0.63	0.63	0.00	0.75	0.00	0.51
Gemini 2.0 [39]	0.28	0.61	0.50	0.63	0.56	0.75	0.13	0.00	0.25	0.33	0.39
Claude Sonnet 4.5 [3]	0.31	0.80	0.67	0.67	0.67	0.63	0.50	0.00	1.00	1.00	0.62
Claude Sonnet 3.7 [2]	0.25	0.69	0.48	0.47	0.47	0.63	0.13	0.00	1.00	0.90	0.51
Open-Source Models											
Qwen2.5-VL-72B [6]	0.18	0.44	0.72	0.75	0.73	1.00	0.50	0.00	1.00	0.33	0.55
Qwen2.5-VL-7B [6]	0.13	0.14	0.28	0.39	0.34	0.00	0.00	0.00	0.75	0.03	0.19
Qwen2.5-VL-3B [6]	0.14	0.14	0.22	0.27	0.24	0.00	0.00	0.00	0.00	0.00	0.09
VLM-R1-3B [35]	0.13	0.13	0.31	0.34	0.33	0.00	0.00	0.00	0.00	0.00	0.10
VAGEN [41]	0.61	0.71	0.78	0.80	0.79	1.00	0.88	0.88	0.88	0.50	0.78
Ours											
AtlasVA	**0.79**	**0.83**	**0.85**	**0.87**	**0.86**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.93**

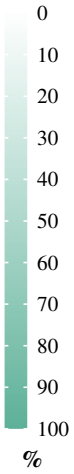

Table 2 Performance comparison across different tasks. Cell shading indicates relative performance within each column, with deeper color denoting higher success rates.

FROZENLAKE [40]), Embodied 3D NAVIGATION [21, 51], and Detailed Object Manipulation (PRIMITIVESKILL [38]). PRIMITIVESKILL, first introduced by VAGEN [41], evaluates multi-turn robotic control. In addition to its standard tasks (Place, Stack, Drawer, and Align), we introduce a new *Swap* task, which requires the agent to exchange the positions of two cubes.

4.2 Main Results

We evaluate AtlasVA across four diverse environments, ranging from 2D discrete grid-worlds (SOKOBAN, FROZENLAKE) to continuous 3D embodied navigation and 3D robotic manipulation (PrimitiveSkill via ManiSkill). Quantitative comparisons are detailed in Table 2.

Overall Performance. AtlasVA achieves state-of-the-art performance across all evaluated benchmarks with an average success rate of 0.93 (Table 2). Despite using a compact 3B-parameter base model, AtlasVA significantly outperforms substantially larger proprietary models, including GPT-5 (0.69) and o3 (0.71) (**RQ1**). Furthermore, it establishes a considerable margin over the strongest open-source baseline, VAGEN (0.78). These results demonstrate that integrating visual skill memory with dense reward shaping effectively overcomes the performance ceiling of traditional text-centric VLM agents.

Efficacy in Spatial Reasoning. AtlasVA demonstrates strong spatial reasoning capabilities in environments that require intensive geometric planning, such as Sokoban and FrozenLake. Baseline open-source models exhibit limited spatial comprehension; for instance, zero-shot Qwen2.5-VL-3B obtains a success rate of merely 0.14 in Sokoban. By contrast, AtlasVA elevates this performance to 0.79, outperforming even the proprietary GPT-5 (0.70). For zero-shot transfer to unseen validation layouts with novel topologies, AtlasVA relies on the dynamic heuristic branch ($M_{heuristic}$) of the heatmaps, which computes topological features (e.g., BFS fields and corner traps) on-the-fly for the new layout, ensuring robust generalization despite shifted entities. This substantial gain empirically validates that progressively evolving spatial heatmaps successfully embed critical geometric priors, overcoming the inherent limitations of pure text-based representations in spatial tasks.

Robustness in 3D Navigation and 3D Robotic Manipulation. AtlasVA maintains robust execution across complex, continuous 3D tasks such as embodied Navigation and PrimitiveSkill robotic manipulation. On the 3D Navigation benchmark, AtlasVA yields an average success rate of 0.86 across all sub-tasks. In the 3D PrimitiveSkill manipulation domain (Place, Stack, Drawer, Align, Swap via ManiSkill), the method achieves a perfect 1.00 success rate across all five categories. This starkly contrasts with existing baselines, which frequently fail on complex manipulations; for instance, on the Swap task, Qwen2.5-VL-72B and GPT-5 achieve success rates of only 0.33 and 0.55, respectively. We

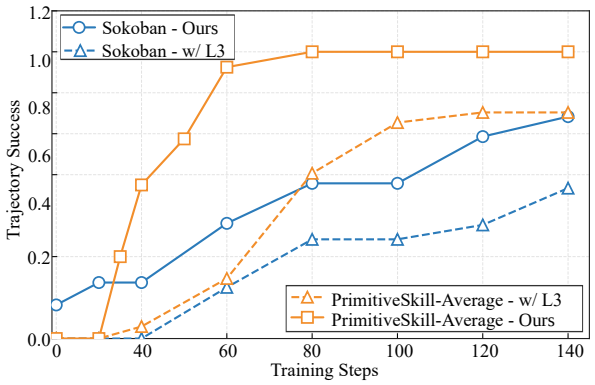

Figure 4 Learning efficiency. AtlasVA (Ours) achieves faster convergence and higher task success than the text-only baseline (w/ L3), proving the efficiency of visual feedback.

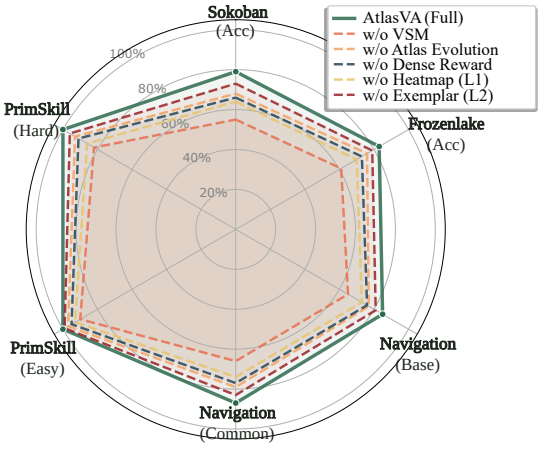

Figure 5 Component ablation. Removing the visual skill memory (VSM), atlas evolution, or dense reward shaping consistently degrades performance across all spatial benchmarks.

attribute this robustness to the integration of contrasting visual exemplars, which effectively regularizes the policy and prevents the agent from revisiting historical failure modes during extended execution.

Learning Efficiency and Convergence. Beyond final task success, AtlasVA significantly accelerates training convergence compared to a text-centric baseline (Figure 4). The baseline, relying solely on textual rules (Layer 3), struggles to exceed a 0.25 success rate in the 2D Sokoban environment due to severe modality mismatch and sparse feedback. In contrast, AtlasVA rapidly climbs to approximately 0.80 success within 140 training steps. Similarly, on the 3D PrimitiveSkill benchmark, AtlasVA quickly converges to a perfect success rate, whereas the baseline plateaus at roughly 0.60. This accelerated learning curve confirms that our teacher-free visual skill memory and dense visual reward shaping provide immediate, spatially grounded feedback, effectively solving the credit assignment problem and improving sample efficiency (**RQ2**).

4.3 Ablation Studies

Diagnostic Analysis. To evaluate AtlasVA's core components (Figure 5), we analyze five ablations to address our research questions. *Visual Skill Memory:* Removing the entire visual hierarchy (w/o VSM) severely degrades performance on spatial puzzles (Sokoban, FrozenLake), confirming that compressing geometric constraints into text causes fatal information loss and validating the necessity of native visual grounding (**RQ4**). Furthermore, omitting either topological maps (w/o Heatmap) or episodic retrieval (w/o Exemplar) hurts performance, proving both layers are required for optimal execution. *Atlas Evolution:* Disabling off-policy updates (w/o Atlas Evolution) restricts the agent to static rules. The resulting major regression across all tasks proves that spatial heatmaps can be effectively bootstrapped from raw statistics alone, without external teacher LLMs (**RQ3**). *Dense Reward Shaping:* A variant trained with only sparse feedback (w/o Dense Reward) struggles to escape local optima in long-horizon tasks. This confirms that translating the visual atlas into dense shaping signals is essential to mitigate the sparse-reward bottleneck and accelerate convergence (**RQ2**).

Evolution of Spatial Heatmaps. Visualizing the progression of spatial heatmaps confirms that AtlasVA effectively extracts geometric priors through pure environment interaction (Figure 6). At Step 0, the initial heatmaps contain no spatial information. As training progresses via EMA updates, the heatmaps rapidly capture meaningful topological structures. By Step 200, the *Danger* map (top row) explicitly highlights structural hazards and dead-ends, while the *Affinity* map (bottom row) traces traversable sub-goal paths. This qualitative progression directly illustrates how the framework distills 2D layouts into visual priors without relying on external language model supervision.

Dynamics of the Visual Exemplar Pool. The visual exemplar pool dynamically updates to maintain highly relevant context throughout training, while keeping the prompt size strictly bounded (Figures 7 and 8). Our implementation caps the pool at 6 exemplars, which it populates within the first 40 training steps (Figure 7). The rapid establishment of this pool directly correlates with an initial spike in the validation success rate from near zero to over 70%, demonstrating the immediate benefit of visual references. Furthermore, Figure 8 illustrates the continuous eviction mechanism: early exemplars (e.g., POS#001) are gradually replaced by newer observations (e.g., POS#020) as the agent explores novel

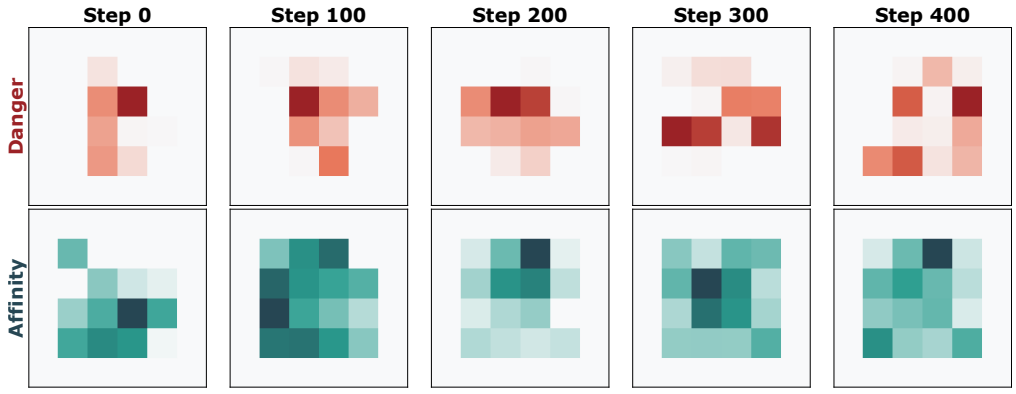

Figure 6 Evolution of Spatial Heatmaps (Layer 1). From Step 0 to Step 400, through pure environment interaction, the heatmaps learn to encode structural hazards (Danger, red) and sub-goal paths (Affinity, green).

states. The high retrieval frequency of active exemplars (indicated by shading intensity) confirms that the agent actively consults this rolling buffer, ensuring that the provided visual context matches its current exploration frontier.

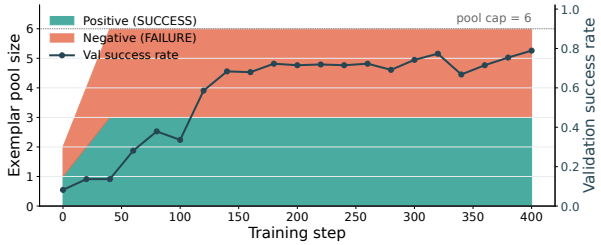

Figure 7 Exemplar pool capacity. The accumulation of visual exemplars coincides directly with a rapid increase in validation success rate.

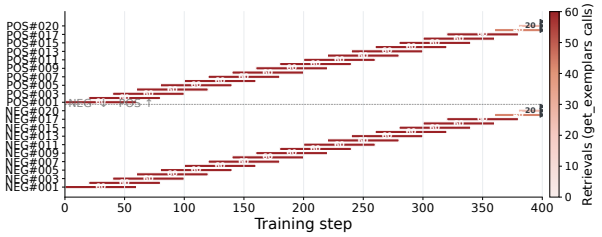

Figure 8 Lifecycle of individual exemplars. The pool continuously evicts older observations in favor of newer ones, providing dynamically updated visual context.

Impact of Dense Visual Reward Shaping. Our Atlas-Grounded Dense Reward Shaping successfully converts sparse environment feedback ($r \in \{0, 1\}$), which provides inadequate gradients for long-horizon tasks, into a continuous optimization signal. As shown in Figure 9, our visual potential function explicitly supplements the sparse rewards: a **Danger Penalty** heavily discourages approaching learned hazards, while an **Affinity Gain** rewards progress toward sub-goals. This waterfall analysis verifies that translating 2D spatial heatmaps into a continuous potential field provides the dense guidance necessary to mitigate the credit assignment problem in sparse-reward settings.

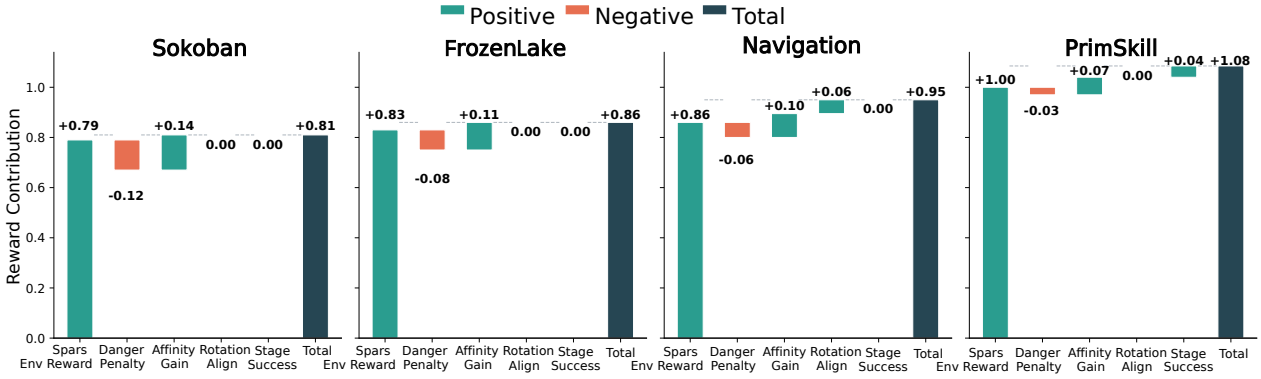

Figure 9 Reward waterfall analysis. The composition of the total reward demonstrates that the Danger Penalty (orange) and Affinity Gain (green) provide continuous step-level signals, effectively supplementing the sparse environment rewards.

5 Conclusion and Limitations

We presented **AtlasVA**, a native Visual Skill Memory that resolves spatial blindness and modality mismatch in VLM agents. By bootstrapping geometric priors directly from trajectories, AtlasVA enables continuous visual reward shaping

without external teacher supervision. A compact 3B-parameter AtlasVA significantly outperforms larger models like GPT-5 on diverse spatial benchmarks. A primary limitation is the projection of table-top 3D manipulation into 2.5D visual priors; scaling this to highly occluded, ego-centric 3D robotics remains critical future work.

References

[1] An, D., Wang, H., Wang, W., Wang, Z., Huang, Y., He, K., Wang, L.: Etpnav: Evolving topological planning for vision-language navigation in continuous environments. IEEE Transactions on Pattern Analysis and Machine Intelligence **47**(7), 5130–5145 (2025). https://doi.org/10.1109/TPAMI.2024.3386695

[2] Anthropic: The claude 3 model family: Opus, sonnet, haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf (2024), model card

[3] Anthropic: Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5 (2025)

[4] Asai, A., Wu, Z., Wang, Y., Sil, A., Hajishirzi, H.: Self-rag: Learning to retrieve, generate, and critique through self-reflection. In: The Twelfth International Conference on Learning Representations (2023)

[5] Bai, H., Zhou, Y., Cemri, M., Pan, J., Suhr, A., Levine, S., Kumar, A.: Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. Advances in Neural Information Processing Systems **37**, 12461–12495 (2024)

[6] Bai, J., Bai, S., Yang, S., Wang, S., Tan, S., Wang, P., Lin, J., Zhou, C., Zhou, J.: Qwen-vl: A versatile vision-language model for understanding, localization. Text Reading, and Beyond **2**(1), 1 (2023)

[7] Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., Munos, R.: Unifying count-based exploration and intrinsic motivation. Advances in neural information processing systems **29** (2016)

[8] Cheng, A.C., Yin, H., Fu, Y., Guo, Q., Yang, R., Kautz, J., Wang, X., Liu, S.: Spatialrgpt: Grounded spatial reasoning in vision-language models. Advances in Neural Information Processing Systems **37**, 135062–135093 (2024)

[9] Chhikara, P., Khant, D., Aryan, S., Singh, T., Yadav, D.: Mem0: Building production-ready ai agents with scalable long-term memory. arXiv preprint arXiv:2504.19413 (2025)

[10] Cui, X., Liu, Q., Liu, Z., Wang, H.: Frontier-enhanced topological memory with improved exploration awareness for embodied visual navigation. In: European Conference on Computer Vision. pp. 296–313. Springer (2024)

[11] Deng, X., Gu, Y., Zheng, B., Chen, S., Stevens, S., Wang, B., Sun, H., Su, Y.: Mind2web: Towards a generalist agent for the web. Advances in Neural Information Processing Systems **36**, 28091–28114 (2023)

[12] Feng, S., Tuo, K., Wang, S., Kong, L., Zhu, J., Wang, H.: Rewardmap: Tackling sparse rewards in fine-grained visual reasoning via multi-stage reinforcement learning. arXiv preprint arXiv:2510.02240 (2025)

[13] Google: Gemini 2.5: Our most intelligent ai model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/ (2025)

[14] Gou, B., Wang, R., Zheng, B., Xie, Y., Chang, C., Shu, Y., Sun, H., Su, Y.: Navigating the digital world as humans do: Universal visual grounding for GUI agents. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=kxnoqaisCT

[15] Hurst, A., Lerer, A., Goucher, A.P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al.: Gpt-4o system card. arXiv preprint arXiv:2410.21276 (2024)

[16] Jiang, G., Su, Z., Qu, X., et al.: Xskill: Continual learning from experience and skills in multimodal agents. arXiv preprint arXiv:2603.12056 (2026)

[17] Jiang, H., Lu, Z.: Visual grounding for object-level generalization in reinforcement learning. In: European Conference on Computer Vision. pp. 55–72. Springer (2024)

[18] Jiang, Y., Liu, Q., Yang, Y., Ma, X., Zhong, D., Hu, H., Yang, J., Liang, B., Xu, B., Zhang, C., et al.: Episodic novelty through temporal distance. arXiv preprint arXiv:2501.15418 (2025)

[19] Kang, H., Sachdeva, E., Gupta, P., Bae, S., Lee, K.: Gflowvlm: Enhancing multi-step reasoning in vision-language models with generative flow networks. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 3815–3825 (2025)

[20] Koh, J.Y., Lo, R., Jang, L., Duvvur, V., Lim, M., Huang, P.Y., Neubig, G., Zhou, S., Salakhutdinov, R., Fried, D.: Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 881–905 (2024)

[21] Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Deitke, M., Ehsani, K., Gordon, D., Zhu, Y., et al.: Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474 (2017)

[22] Li, D., Zhang, Y., Cao, M., Liu, D., Xie, W., Hui, T., Lin, L., Xie, Z., Li, Y.: Towards long-horizon vision-language-action system: Reasoning, acting and memory. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6839–6848 (2025)

[23] Lin, B., Nie, Y., Wei, Z., Chen, J., Ma, S., Han, J., Xu, H., Chang, X., Liang, X.: Navcot: Boosting llm-based vision-and-language navigation via learning disentangled reasoning. IEEE Transactions on Pattern Analysis and Machine Intelligence **47**(7), 5945–5957 (2025). https://doi.org/10.1109/TPAMI.2025.3554559

[24] Lin, K.Q., Li, L., Gao, D., Yang, Z., Wu, S., Bai, Z., Lei, S.W., Wang, L., Shou, M.Z.: Showui: One vision-language-action model for gui visual agent. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 19498–19508 (2025)

[25] Liu, R., Wang, W., Yang, Y.: Volumetric environment representation for vision-language navigation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16317–16328 (2024)

[26] Luo, T., Logeswaran, L., Johnson, J., Lee, H.: Visual test-time scaling for gui agent grounding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19989–19998 (2025)

[27] Ng, A.Y., Harada, D., Russell, S.: Policy invariance under reward transformations: Theory and application to reward shaping. In: Icml. vol. 99, pp. 278–287. Citeseer (1999)

[28] Niu, R., Li, J., Wang, S., Fu, Y., Hu, X., Leng, X., Kong, H., Chang, Y., Wang, Q.: Screenagent: A vision language model-driven computer control agent. arXiv preprint arXiv:2402.07945 (2024)

[29] OpenAI: Introducing gpt-5. https://openai.com/index/introducing-gpt-5/ (2025)

[30] OpenAI: Introducing openai o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/ (2025)

[31] Paglieri, D., Cupiał, B., Coward, S., Piterbarg, U., Wolczyk, M., Khan, A., Pignatelli, E., Kuciński, Ł., Pinto, L., Fergus, R., et al.: Balrog: Benchmarking agentic llm and vlm reasoning on games. arXiv preprint arXiv:2411.13543 (2024)

[32] Rocamonde, J., Montesinos, V., Nava, E., Perez, E., Lindner, D.: Vision-language models are zero-shot reward models for reinforcement learning. arXiv preprint arXiv:2310.12921 (2023)

[33] Sarch, G., Saha, S., Khandelwal, N., Jain, A., Tarr, M.J., Kumar, A., Fragkiadaki, K.: Grounded reinforcement learning for visual reasoning. arXiv preprint arXiv:2505.23678 (2025)

[34] Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., Manning, C.D.: Raptor: Recursive abstractive processing for tree-organized retrieval. In: The Twelfth International Conference on Learning Representations (2024)

[35] Shen, H., Liu, P., Li, J., Fang, C., Ma, Y., Liao, J., Shen, Q., Zhang, Z., Zhao, K., Zhang, Q., et al.: Vlm-r1: A stable and generalizable r1-style large vision-language model. arXiv preprint arXiv:2504.07615 (2025)

[36] Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: Language agents with verbal reinforcement learning. Advances in neural information processing systems **36**, 8634–8652 (2023)

[37] Sridhar, K., Dutta, S., Jayaraman, D., Lee, I.: REGENT: A retrieval-augmented generalist agent that can act in-context in new environments. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=NxyfSW6mLK

[38] Tao, S., Xiang, F., Shukla, A., Qin, Y., Hinrichsen, X., Yuan, X., Bao, C., Lin, X., Liu, Y., Chan, T.k., et al.: Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. arXiv preprint arXiv:2410.00425 (2024)

[39] Team, G., Anil, R., Borgeaud, S., Alayrac, J.B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., et al.: Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 (2023)

[40] Towers, M., Kwiatkowski, A., Terry, J., Balis, J.U., De Cola, G., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., et al.: Gymnasium: A standard interface for reinforcement learning environments. arXiv preprint arXiv:2407.17032 (2024)

[41] Wang, K., Zhang, P., Wang, Z., Gao, Y., Li, L., Wang, Q., Chen, H., Wan, C., Lu, Y., Yang, Z., et al.: Vagen: Reinforcing world model reasoning for multi-turn vlm agents. arXiv preprint arXiv:2510.16907 (2025)

[42] Wang, N., Hu, X., Liu, P., Zhu, H., Hou, Y., Huang, H., Zhang, S., Yang, J., Liu, J., Zhang, G., et al.: Efficient agents: Building effective agents while reducing cost. arXiv preprint arXiv:2508.02694 (2025)

[43] Wang, Y., Sun, Z., Zhang, J., Xian, Z., Biyik, E., Held, D., Erickson, Z.: Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. arXiv preprint arXiv:2402.03681 (2024)

[44] Wang, Z., Cai, S., Mu, Z., Lin, H., Zhang, C., Liu, X., Li, Q., Liu, A., Ma, X., Liang, Y.: Omnijarvis: Unified vision-language-action tokenization enables open-world instruction following agents. Advances in Neural Information Processing Systems **37**, 73278–73308 (2024)

[45] Wang, Z., Chen, W., Yang, L., Zhou, S., Zhao, S., Zhan, H., Jin, J., Li, L., Shao, Z., Bu, J.: Mp-gui: Modality perception with mllms for gui understanding. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 29711–29721 (2025)

[46] Wei, T., Yang, Y., Xing, J., Shi, Y., Lu, Z., Ye, D.: Gtr: Guided thought reinforcement prevents thought collapse in rl-based vlm agent training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18855–18865 (2025)

[47] Wu, Q., Cheng, K., Yang, R., Zhang, C., Yang, J., Jiang, H., Mu, J., Peng, B., Qiao, B., Tan, R., et al.: Gui-actor: Coordinate-free visual grounding for gui agents. arXiv preprint arXiv:2506.03143 (2025)

[48] Xia, P., Chen, J., Wang, H., Liu, J., Zeng, K., Wang, Y., Han, S., Zhou, Y., Zhao, X., Chen, H., et al.: Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. arXiv preprint arXiv:2602.08234 (2026)

[49] Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., Zhang, Y.: A-mem: Agentic memory for llm agents. arXiv preprint arXiv:2502.12110 (2025)

[50] Yang, J., Yang, S., Gupta, A.W., Han, R., Fei-Fei, L., Xie, S.: Thinking in space: How multimodal large language models see, remember, and recall spaces. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 10632–10643 (2025)

[51] Yang, R., Chen, H., Zhang, J., Zhao, M., Qian, C., Wang, K., Wang, Q., Koripella, T.V., Movahedi, M., Li, M., et al.: Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. arXiv preprint arXiv:2502.09560 (2025)

[52] Yang, Y.: Text2reward: Reward shaping with language models for reinforcement learning. In: International Conference on Learning Representations (ICLR), 2024 (07/05/2024-11/05/2024, Vienna, Austria) (2024)

[53] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K.R., Cao, Y.: React: Synergizing reasoning and acting in language models. In: The eleventh international conference on learning representations (2022)

[54] Zhang, D., Chen, L., Zhang, S., Xu, H., Zhao, Z., Yu, K.: Large language models are semi-parametric reinforcement learning agents. Advances in Neural Information Processing Systems **36**, 78227–78239 (2023)

[55] Zhao, A., Huang, D., Xu, Q., Lin, M., Liu, Y.J., Huang, G.: Expel: Llm agents are experiential learners. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 19632–19642 (2024)

[56] Zhong, V., Misra, D., Yuan, X., Côté, M.A.: Policy improvement using language feedback models. Advances in Neural Information Processing Systems **37**, 43730–43758 (2024)

[57] Zhou, Y., Dai, S., Wang, S., Zhou, K., Jia, Q., Xu, J.: Gui-g1: Understanding r1-zero-like training for visual grounding in gui agents. arXiv preprint arXiv:2505.15810 (2025)

Appendix

A Algorithmic and Mathematical Framework

While Section 3 introduced the conceptual framework of AtlasVA, including the Visual Skill Memory and dense reward shaping, this section provides the formal mathematical definitions and algorithmic flow of how these components fundamentally alter the underlying decision-making process. Specifically, we detail how the standard Partially Observable Markov Decision Process (POMDP) is augmented with an evolving observation space and a dynamic potential-based reward function.

A.1 Standard POMDP vs. Augmented Observation Space

A standard multi-turn VLM agentic task is typically defined as a POMDP tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{E}, \mathcal{R}, \gamma \rangle$, where:

- $\mathcal{S}$ is the set of true unobservable environment states.
- $\mathcal{O}$ is the raw observation space (e.g., a first-person or top-down RGB image o_t).
- $\mathcal{R}(s_t, a_t) \to \{0, 1\}$ is the sparse environment reward.

In standard approaches (e.g., VAGEN), the VLM policy $\pi_\theta(a_t | o_{\leq t})$ conditions only on the raw visual input o_t. To mitigate the spatial blindness discussed in Section 3.2, AtlasVA explicitly expands the standard observation space. At any time step t in episode k, the policy is conditioned on an augmented observation tuple $\tilde{o}_t$:

$$\tilde{o}_t = \langle o_t, \mathcal{M}_k^{(heatmap)}, \mathcal{M}_k^{(exemplar)}, \mathcal{M}_k^{(text)} \rangle \tag{4}$$

where $\mathcal{M}_k$ denotes the state of the three-layer Visual Skill Memory. This formalization ensures that non-local spatial priors (e.g., distant deadlock zones) become a native part of the state representation rather than relying on error-prone textual coordinate inference.

A.2 Formalization of Potential-Based Dense Reward

As introduced in Section 3.4, AtlasVA converts the highly sparse environment reward r_t^{env} into a dense, per-step reward $\tilde{r}_t$. Here we provide the formal definition of the shaping term, inspired by potential-based reward shaping [27] on the affinity branch and a heuristic safety constraint on the danger branch.

We define a visual potential field $\Phi(o_t)$ derived directly from the spatial heatmap $\mathcal{M}_k^{(heatmap)}$. The shaped reward is formulated as:

$$\tilde{r}_t = r_t^{env} + F(o_t, a_t, o_{t+1}) \tag{5}$$

where the shaping term F combines an affinity potential difference and a danger penalty:

$$F(o_t, a_t, o_{t+1}) = \left[\Phi_{\text{affinity}}(o_{t+1}) - \Phi_{\text{affinity}}(o_t)\right] - \beta \cdot \mathbb{I}(\text{enters\_danger}(o_{t+1})) \tag{6}$$

where:

- $\Phi_{\text{affinity}}(o_t)$ evaluates the agent's current position on the affinity heatmap, whose base layer is a BFS distance field over reachable cells. The per-step difference $\Phi(o_{t+1}) - \Phi(o_t)$ therefore yields a non-trivial gradient at every state rather than only at terminals.
- $\mathbb{I}(\text{enters\_danger})$ triggers a danger penalty when the next observation falls into a region identified as a deadlock or fatal trap. We treat this term as a heuristic safety constraint that is intentionally non-potential-based: it deliberately reshapes the optimal policy to favor safer trajectories over hazardous shortcuts.
- β is a scaling coefficient for the penalty.

The potential Φ_{affinity} blends static layout heuristics (BFS distance field) with EMA-accumulated trajectory statistics, providing a continuous visual gradient that transforms the sparse MDP into a dense optimization landscape.

A.3 Off-Policy Memory Evolution Dynamics

Traditional RL environments maintain static transition dynamics and reward functions. However, as conceptually described in Section 3.3, AtlasVA operates as a dynamically evolving system.

We mathematically define this evolution as an off-policy update loop. At the end of a rollout batch (comprising trajectories τ), the spatial priors are updated via an Exponential Moving Average (EMA) mechanism:

$$\mathcal{M}_{k+1}^{(heatmap)} \leftarrow \alpha \mathcal{M}_{k}^{(heatmap)} + (1-\alpha) \cdot \text{Extract}(\tau_k) \tag{7}$$

where α is the EMA decay rate, and $\text{Extract}(\cdot)$ is a non-parametric mapping function. Specifically, Extract aggregates terminal failure positions into the danger map and normalized state visitation frequencies of successful paths into the affinity map. These trajectory-driven statistics are then blended with the static layout heuristics $M_{heuristic}$ described in Section 3.3 (BFS distance field for affinity; corner / wall-adjacency / hole-neighborhood attenuation for danger), so that the final maps retain a dense spatial gradient at every cell even early in training. This formalizes a self-bootstrapping cycle: as the policy π_θ improves, the generated trajectories τ yield higher-quality updates to $\mathcal{M}$, which in turn provides more accurate potential functions Φ and augmented observations $\tilde{o}_t$ for subsequent policy optimization.

A.4 Algorithmic Formalization of AtlasVA Evolution

To avoid redundancy with standard multi-turn PPO loops (which are well-documented in prior frameworks such as VAGEN), Algorithm 1 specifically formalizes the core mathematical contributions of this work: the **Teacher-Free Visual Atlas Evolution** and the **Atlas-Grounded Visual Reward Shaping**. It details how the visual priors are extracted from raw trajectory statistics and converted into dynamic potential functions.

B Prompt Template

```
[System] Sokoban rules and output format ...
## Spatial Skill Maps
Danger zones (red): ⟨image⟩
Goal affinity (green): ⟨image⟩
## Visual Exemplars (optional, when pool is non-empty)
Positive cases: ⟨image⟩, ⟨image⟩, ...
Negative cases: ⟨image⟩, ⟨image⟩, ...
## Learned Principles
### General Principles / Push Strategies / Mistakes to Avoid ...

[User] [Initial Observation]: ⟨image⟩
Decide your next action(s).
```

Figure 10 Prompt template of AtlasVA. The three VSM layers (L1/L2/L3) are injected at fixed anchor positions, followed by the current observation in the user turn.

C Implementation and Environmental Details

This section provides the construction details of the Visual Skill Memory (VSM) and the exact reward assignment hyperparameters used across our evaluated environments.

C.1 Layer 1: Spatial Heatmaps (Dense Perceptual Priors)

While Section 3.2 introduces the concept of rendering M_{danger} and $M_{affinity}$ as RGB heatmaps, here we detail the precise tensor-to-image mapping pipeline used to bridge the discrete grid statistics and the VLM's continuous visual encoder.

Construction & Rendering. The environment layout is mapped to a discrete 2D grid $\mathbf{p} = (x, y)$. During the EMA evolution phase, the raw aggregated statistics $M^{danger}(\mathbf{p})$ and $M^{affinity}(\mathbf{p})$ are stored as floating-point tensors in $[0, 1]$. To render these into visual tokens without losing precision or confusing the VLM's pre-trained color semantics:

Algorithm 1 Teacher-Free Visual Atlas Evolution and Dense Reward Shaping

Input: Rollout buffer $\mathcal{B}$ from the current PPO epoch, containing spatial coordinates $\mathbf{p}_t$, observations o_t, actions a_t, and sparse rewards r_t^{env}.
State Variables: Historical Danger Map M_{danger}, Affinity Map $M_{affinity}$, Exemplar Pool $\mathcal{E}_{vis}$.
Hyperparameters: EMA decay α, reward scaling β, γ.

# Phase 1: Atlas-Grounded Visual Reward Shaping (Online)
for each transition $(\mathbf{p}_t, a_t, \mathbf{p}_{t+1})$ in $\mathcal{B}$ **do**
  $\Phi_{\text{affinity}} \leftarrow M_{affinity}(\mathbf{p}_{t+1})$
  $r_{affinity} \leftarrow \Phi_{\text{affinity}} - M_{affinity}(\mathbf{p}_t)$
  $r_{danger} \leftarrow -\beta \cdot M_{danger}(\mathbf{p}_{t+1})$
  $\tilde{r}_t \leftarrow r_t^{env} + r_{affinity} + r_{danger}$ # Construct dense visual reward
  Replace r_t^{env} with visual shaped reward $\tilde{r}_t$ in $\mathcal{B}$
end for
Execute Standard PPO Update for policy π_θ using the dense visual rewards $\tilde{r}_t$ in $\mathcal{B}$

Phase 2: Teacher-Free Atlas Evolution (Offline)
Extract failed trajectories $\mathcal{T}_{fail}$ and successful trajectories $\mathcal{T}_{succ}$ from $\mathcal{B}$
2.1 Danger Map Evolution
Initialize batch danger map $M_{batch}^{danger} \leftarrow \mathbf{0}$
for each trajectory $\tau \in \mathcal{T}_{fail}$ **do**
  $\mathbf{p}_{terminal} \leftarrow$ terminal coordinate of τ
  $M_{batch}^{danger}(\mathbf{p}_{terminal}) \mathrel{+}= 1/|\mathcal{T}_{fail}|$
end for
$M_{danger} \leftarrow \alpha M_{danger} + (1-\alpha) M_{batch}^{danger}$
2.2 Affinity Map Evolution
Initialize batch affinity map $M_{batch}^{affinity} \leftarrow \mathbf{0}$
for each trajectory $\tau \in \mathcal{T}_{succ}$ **do**
  for each visited coordinate $\mathbf{p} \in \tau$ **do**
    $M_{batch}^{affinity}(\mathbf{p}) \mathrel{+}= 1/(\text{length of } \tau \times |\mathcal{T}_{succ}|)$
  end for
end for
$M_{affinity} \leftarrow \alpha M_{affinity} + (1-\alpha) M_{batch}^{affinity}$
2.3 Visual Exemplar Pool Update
Identify critical inflection keyframes from $\mathcal{T}_{fail}$ and $\mathcal{T}_{succ}$
Update $\mathcal{E}_{vis}$ via DINOv2 cosine distance matching and FIFO eviction
Return: Updated Visual Skill Memory $\{M_{danger}, M_{affinity}, \mathcal{E}_{vis}\}$

- **Alpha-Channel Gradient Mapping:** We apply a strict linear colormap. The danger tensor is mapped to a pure red channel (`R=255, G=0, B=0`) with the alpha (opacity) channel directly proportional to the danger value. The affinity tensor is identically mapped to a pure green channel (`R=0, G=255, B=0`).
- **Tokenization Strategy:** As shown in the prompt template (Appendix B), we inject these rendered heatmaps as standalone auxiliary image tokens (i.e., `<image>`) rather than directly alpha-blending them over the current observation o_t. Empirical testing revealed that direct blending often corrupts the VLM's recognition of small foreground interactive objects. By providing them as separate spatial reference maps, the VLM natively learns to cross-attend between the clean observation and the semantic heatmaps.

C.2 Layer 2: Visual Exemplars (Episodic Context)

To compensate for the loss of temporal causality in the aggregated heatmaps, Layer 2 maintains an episodic memory of critical keyframes ($\mathcal{E}_{vis}$). Here we detail the dynamic retrieval mechanism.

Construction and Retrieval. The exemplar pool is dynamically populated during the off-policy evolution phase.

We filter historical trajectories to extract "inflection point" frames—e.g., the exact frame immediately preceding an irreversible deadlock or a critical sub-goal completion.

During inference, we deploy a dense retrieval mechanism. Given the current observation o_t, we retrieve the top-k most structurally relevant frames from the exemplar pool:

$$\mathcal{E}_t = \arg\max_{e_i \in \mathcal{E}} \text{CosSim}(f_{\text{DINO}}(o_t), f_{\text{DINO}}(e_i)) \tag{8}$$

where $f_{\text{DINO}}(\cdot)$ denotes the feature embeddings extracted by a frozen DINOv2 encoder. DINOv2 is chosen over CLIP because its self-supervised objective yields superior patch-level spatial correspondences, which is crucial for 2D topological matching in environments like Sokoban. The retrieved frames are then partitioned into "Positive Cases" and "Negative Cases" and appended to the prompt, enabling zero-shot visual pattern matching.

C.3 Layer 3: Symbolic Text Skills (Semantic Grounding)

While the final prompt structure (General Principles, Push Strategies, Mistakes to Avoid) is illustrated in Listing 10, we clarify that these symbolic rules do not require any proprietary Teacher LLM (e.g., GPT-4) to generate or summarize. Instead, $\mathcal{S}_{text}$ is constructed directly from the environment's underlying rulebook and task descriptions.

This static text acts as the cognitive scaffolding that binds the VSM together: the textual rules dictate the foundational logic (e.g., what constitutes a valid push), while the self-evolved heatmaps (Layer 1) and retrieved exemplars (Layer 2) visually ground *where* and *how* that logic applies in the current continuous pixel space. By sourcing text skills directly from environment specifications, AtlasVA remains completely teacher-free.

C.4 Environment Details and Reward Assignment

To facilitate reproducibility, we detail the exact reward assignment hyper-parameters across all evaluated environments in Table 3.

While the sparse environment rewards (Success and Failure) provide the terminal objective, AtlasVA introduces two new dense visual reward scaling coefficients: λ_{danger} (controlling the penalty for entering deadlock regions identified by the Danger Map) and $\lambda_{affinity}$ (controlling the step-wise gain for moving towards goal regions identified by the Affinity Map). We empirically found that tasks requiring high-precision, long-horizon coordinate manipulation (e.g., PrimitiveSkill) benefit from stronger visual guidance ($\lambda_{danger} = 0.3, \lambda_{affinity} = 0.3$) compared to pure grid-world planning tasks like Sokoban ($\lambda_{danger} = 0.05, \lambda_{affinity} = 0.05$).

Clarification on Reward Sparsity and Auxiliary Components. As formulated in Section 3.1, the underlying environment dynamics strictly provide sparse binary rewards $\mathcal{R} \in \{0, 1\}$. However, to successfully train VLM agents, our framework incorporates three auxiliary components (as observed in the reward waterfall analysis):

- **Format Penalty:** A system-level constraint applied when the VLM generates malformed actions or invalid JSON. This is a syntactic regularizer independent of the spatial environment dynamics.
- **Rotation Align & Stage Success:** To accommodate complex 3D environments, the visual shaping reward r_{visual} is extended with task-specific spatial heuristics. Specifically, a rotation alignment bonus is added for 3D Navigation to encourage facing the goal, and a stage-success delta is folded into the Affinity Gain for multi-step PrimitiveSkill manipulations. These are extensions of our proposed dense shaping mechanism, preserving the strictly sparse nature of the base environment.

C.5 GridState Abstraction and Simulator APIs

As discussed in Section 3.1, the *GridState* abstraction g_t extracts exact 2D/3D coordinates ($\mathbf{p}_t$) directly from the underlying simulator to construct the spatial heatmaps and compute dense visual rewards. Table 4 explicitly details the low-level simulator APIs accessed for each environment.

Crucially, this privileged state access is **strictly confined to the offline atlas evolution and reward computation pipeline during training**. At evaluation time, the VLM policy does not receive any of these low-level states. The policy's input consists entirely of the raw RGB observation o_t and the rendered visual memory prompts (heatmaps and exemplars), ensuring that the zero-shot comparisons in our main results remain strictly fair and vision-based.

Table 3 Reward assignment parameters across evaluated environments. Standard sparse rewards (Success/Failure) are heavily augmented by AtlasVA's continuous visual gradients, governed by λ_{danger} and $\lambda_{affinity}$.

Environment	Success	Failure/Timeout	Format Penalty	λ_{danger} (Penalty)	$\lambda_{affinity}$ (Gain)
Sokoban	+1.0	-0.1	-0.5	0.05	0.05
FrozenLake	+1.0	-0.1	-0.5	0.05	0.05
Navigation	+1.0	-0.1	-0.5	0.1	0.1
PrimitiveSkill	+1.0	0.0	-0.5	0.3	0.3

Table 4 Simulator APIs accessed by the GridState abstraction. These privileged states are used exclusively for training-time memory evolution and reward shaping, not as policy inputs.

Environment	Accessed Simulator API / Internal State
Sokoban	`room_state`, `room_fixed`, `player_position`, `boxes_on_target`
FrozenLake	`gym_env.s` (agent index), `gym_env.desc` (map layout)
Navigation	AI2-THOR metadata (agent poses, reachable positions, goal coordinates)
PrimitiveSkill	ManiSkill internal states (`_handle_info`, `last_info`, object poses)

C.6 Optimization and Hyperparameter Configurations

In Table 5, we detail the unified hyperparameters used for the Proximal Policy Optimization (PPO) training and the Visual Skill Memory (VSM) configurations. All experiments were conducted using 8 $\times$ NVIDIA RTX 6000 Ada Generation GPUs. The VLM base model is Qwen2.5-VL-3B-Instruct. We utilize the vLLM engine for asynchronous rollout generation, enforcing strict maximum lengths for prompt and response tokens to accommodate the injected multimodal memory.

The VSM hyperparameters, including the EMA decay rate ($\lambda = 0.85$) and the capacity of the visual exemplar pool (3 positive and 3 negative cases), were kept consistent across all environments to demonstrate the robustness and generalizability of the teacher-free evolution mechanism.

D Additional Qualitative Analysis

D.1 Projection of Continuous 3D Spaces into 2.5D Visual Priors

A natural question arises when applying AtlasVA to continuous 3D embodied navigation and 3D robotic manipulation tasks (e.g., ManiSkill): how does a 2D visual atlas accommodate 3D physical spaces?

AtlasVA handles this through an elegant dimensionality-reduction abstraction layer (via the `GridState` module). For 3D Navigation tasks, the environment's continuous reachable surfaces are projected onto the X-Z plane, dynamically discretizing the 3D room into a 2D floor plan (with a resolution of 0.25 meters per cell). For 3D Robotic Manipulation (PrimitiveSkill), the continuous operating table is mapped to a localized 2.5D workspace grid, while the critical Z-axis (height) information is implicitly preserved as task metadata.

During reward shaping, the agent's real-time continuous 3D coordinates are inversely mapped to this 2.5D heatmap. This allows AtlasVA to look up the evolving spatial potential and generate dense, coordinate-aligned gradients. Consequently, the agent executes complex 3D manipulations guided by highly interpretable 2.5D visual priors, significantly reducing the sample complexity typically associated with 3D continuous RL.

D.2 Visualization of Atlas-Grounded Visual Reward Shaping

To provide a concrete understanding of how the visual potential fields alleviate the credit assignment problem in sparse-reward environments, we visualize the step-by-step dense reward dynamics of a successful Sokoban rollout in Figure 11.

In this environment, the agent only receives a sparse extrinsic reward of $+1$ upon completing the entire task. Without intermediate feedback, VLM agents typically resort to random walks or fall into deadlocks. However, as illustrated

Table 5 Comprehensive hyperparameter configurations for PPO optimization and VSM components.

Category	Hyperparameter	Value
PPO & GAE	Actor Learning Rate	1×10^{-6}
	Critic Learning Rate	1×10^{-5}
	Train Batch Size (Rollout)	128
	PPO Mini-Batch Size	32
	KL Control Coefficient	0.0
	Number of GPUs	8
Rollout Engine	Inference Engine	vLLM (async)
	Max Batched Tokens	16384
	Max Prompt Length	5000 - 8000
	Max Response Length	4000
Layer 1: Heatmap	EMA Decay Rate (α)	0.85
	Render Cell Size (pixels)	40
	Active Field Types	Danger, Affinity
Layer 2: Exemplar	Max Positive / Negative Pool	3 / 3
	Max Exemplars Injected	4
	Retrieval Metric	DINOv2 Cosine Sim.
Layer 3: Text Skill	Top-K Skills Injected	3 (Grid), 4 (Swap)
	Pruning Frequency	Every 10 updates

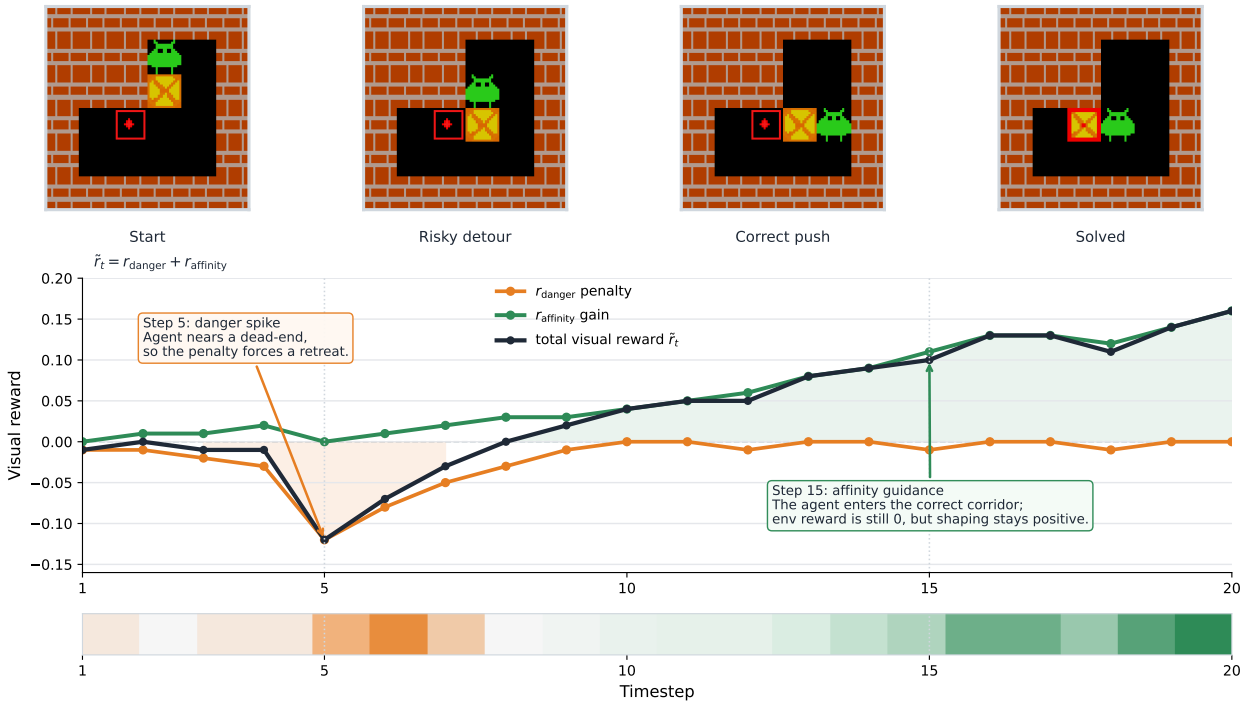

Figure 11 Step-by-step visual reward dynamics in a Sokoban trajectory. The plot expands on the reward composition discussed in Section 3.4 and Figure 9 by demonstrating how AtlasVA converts a single terminal sparse reward into a continuous optimization landscape over time. The Danger Penalty (orange) issues immediate corrective feedback when the agent approaches deadlocks, while the Affinity Gain (green) provides sustained positive signals as the agent navigates along viable sub-goal paths.

by the trajectory curves, the proposed Atlas-Grounded Visual Reward Shaping mechanism provides a continuous, coordinate-specific gradient throughout the episode:

Figure 12 Qualitative rollout comparison between Qwen2.5-VL-72B and AtlasVA. Across spatially demanding tasks (Sokoban, FrozenLake, and PrimitiveSkill Swap), the 72B baseline struggles with geometric reasoning, often falling into deadlocks or failing long-horizon manipulations due to spatial blindness. In contrast, AtlasVA (3B) leverages injected spatial heatmaps (danger in red, affinity in green) to intuitively perceive topological hazards and sub-goals, successfully completing the tasks through precise, native visual grounding.

- **Danger Penalties (Orange Curve):** When the agent takes an action that moves it toward an irreversible corner or wall trap, the visual potential field Φ_{danger} immediately induces a negative penalty. This provides a prompt

corrective signal, forcing the agent to backtrack before a terminal failure occurs.

- **Affinity Gains (Green Curve):** Conversely, as the agent pushes the box along the structurally valid paths toward the goal manifold, the $\Phi_{affinity}$ potential yields sustained positive rewards. These continuous step-level gains guide the agent through the long-horizon task.

By seamlessly replacing sparse feedback with dense, spatially grounded visual gradients, AtlasVA effectively bridges the modality gap and stabilizes multi-turn RL optimization.

D.3 Qualitative Rollout Comparison

To provide deeper intuition into how the Visual Skill Memory fundamentally alters the agent's spatial reasoning capabilities, we visualize complete rollout trajectories in Figure 12. We compare our proposed AtlasVA against a strong baseline, Qwen2.5-VL-72B, across three spatially demanding environments: Sokoban, FrozenLake, and the PrimitiveSkill Swap task.

As illustrated in the baseline trajectories of Figure 12, standard VLMs frequently exhibit "spatial blindness." Despite having a massive parameter count and robust text reasoning capabilities, the 72B baseline struggles to ground abstract textual rules into the unannotated 2D geometric layout. In Sokoban and FrozenLake, this modality mismatch causes the baseline to push boxes into irreversible corners or step into topological traps. In the PrimitiveSkill Swap task, it fails to execute the precise, long-horizon coordinate manipulations required to complete the rearrangement.

Conversely, the AtlasVA trajectories demonstrate the power of multimodal spatial grounding. By natively injecting the self-evolved spatial heatmaps—where red regions highlight topological deadlocks (Danger) and green regions indicate optimal sub-goal paths (Affinity)—AtlasVA transforms complex geometric planning into intuitive visual pattern matching. Even when initialized from a compact 3B parameter model, AtlasVA successfully navigates these bottlenecks without relying on external text summaries, showcasing the immense sample efficiency and spatial awareness unlocked by our teacher-free visual memory.