# RESOLVING STATE-REPRESENTATION MISMATCH: STATE-SPACE VISUAL REASONING FOR OPEN-LOOP VLA PLANNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Despite rapid progress in vision-language-action (VLA) models, existing reasoning paradigms still face a fundamental *state-representation mismatch* in open-loop planning. Given only an initial observation, models must internally simulate action-conditioned state transitions, whereas text-, pixel-, and latent-space reasoning can suffer from lossy spatial compression, error-accumulating visual generation, and bypass of intermediate latent tokens, respectively, undermining reliable long-horizon planning. We propose **State-Space Visual Reasoning** (SSVR), which decouples static visual context, language constraints, and a recurrent latent state. SSVR encodes the initial image and instruction once, then conditions each action prediction on the latent state and updates it with an action-conditioned GRU. Using Qwen2.5-VL as the backbone, SSVR achieves 99.5/99.6, 96.3/98.0, and 83.9/90.6 EM/PR on FrozenLake, Maze, and MiniBehavior, substantially outperforming prior methods. Extensive experiments support the effectiveness of recurrent state modeling for VLA open-loop planning across input transformations and transfer settings. By reusing static visual-textual context and updating a compact recurrent state, SSVR supports efficient multi-step inference, achieving up to $98.58\times$ faster Maze decoding rollouts than the evaluated baselines with the prefix cache prebuilt. Code is publicly available at `project page`.

## 1 INTRODUCTION

Vision-language-action (VLA) systems in robotics (Hu et al., 2023), GUI interaction (Agashe et al., 2025) and navigation (Shah et al., 2023) often need to plan before execution because feedback may be delayed or costly and mistakes may be irreversible. In this open-loop setting, the model observes the initial scene and instruction once, internally models how actions change the execution state, and outputs a complete action sequence. Planning reliability therefore depends on maintaining an informative internal state throughout the rollout.

Existing reasoning paradigms exhibit a fundamental *state-representation mismatch* with this requirement. They conflate two components with different temporal roles: high-dimensional scene context that remains largely static and execution state that evolves with actions. Text-space reasoning compresses map topology and dynamic state into language, making critical spatial relations easy to omit or blur; pixel-space reasoning repeatedly generates complete visual states, so lossy frame errors propagate into later decisions; and latent-space reasoning may bypass intermediate visual tokens. In open-loop rollout, textual omissions and pixel-generation errors can compound, while latent bypass can weaken the use of intermediate visual representations.

This asymmetry between static context and dynamic state is common in VLA open-loop planning (Hao et al., 2026; Wu et al., 2024a; Ivanitskiy et al., 2023; Jin et al., 2023). In the Maze example in Figure 1, the walls and goal remain fixed while only the agent position changes: prior paradigms repeatedly re-express the evolving state in text, images, or hidden vectors, whereas a human can trace the route by holding the map fixed and moving only a finger. Figure 2 shows the failures: topology hallucination in text-space reasoning, object duplication or disappearance in pixel-space rollout, and bypass of intermediate visual tokens in latent-space reasoning. Together, these observations motivate a reasoning paradigm better matched to open-loop state evolution.



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

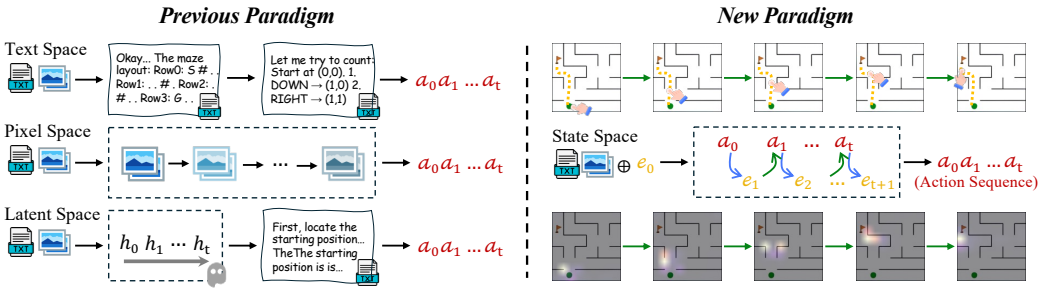



Figure 1: Motivation and evidence for aligning reasoning with dynamic state. Existing paradigms represent evolving state through compressed language, repeatedly generated images, or intermediate latent tokens. SSVR follows the finger-tracing intuition: a static scene is reused, a recurrent latent state is updated by actions through a GRU, and visual attention varies across planning steps.

We propose **State-Space Visual Reasoning** (SSVR), which addresses this mismatch by decoupling static visual context, language constraints, and a recurrent latent state while retaining the pretrained MLLM backbone (Bai et al., 2025). The initial image and instruction are encoded once into a reusable KV cache (Jiang et al., 2025; 2026b); at each step, the latent state token $e_t = z_t$ conditions next-action prediction, and the predicted action updates $z_t$ through a learned GRU transition, recursively advancing the latent state-action trajectory.

Experiments show three advantages of SSVR: **(1) strong planning performance**, with 99.5/99.6, 96.3/98.0, and 83.9/90.6 EM/PR on FrozenLake, Maze, and MiniBehavior, respectively, surpassing the strongest pixel baseline; **(2) robustness and generalization**, with resilience to prompt and image transformations, stronger FrozenLake-to-Maze transfer than VPRL (Xu et al., 2026), and a 22.9-point gain in native Maze VQA optimal-action hit rate without recurrent state conditioning (Table 6), while VQAv2[1] (Goyal et al., 2017) accuracy drops by only 3.9 points from native Qwen; and **(3) efficient reasoning**, with decoding-rollout speedups of $5.81\times$, $38.24\times$, and $98.58\times$ over LVR, Monet, and VPRL, respectively, using a prebuilt prefix cache.

## 2 Related Work

**Text-space visual reasoning.** MLLMs often map image features into language-model space and use text tokens as intermediate reasoning carriers (Liu et al., 2023; Alayrac et al., 2022; Liu et al., 2025c; Venhoff et al., 2025). CoT-style methods externalize spatial inference as symbols (Wu et al., 2024b; Jiang et al., 2026a) or enrich text reasoning with visual abstraction, perception tokens, region distillation, and tool use (Liu et al., 2025b; Bigverdi et al., 2024; Wei et al., 2026; Zheng et al., 2026; Hong et al., 2025). This suits knowledge QA and compactly verbalizable vision tasks, but verbalizing map-like inputs is lossy: text must maintain coordinates, adjacency, reachability, and action history, making critical topology easy to omit or blur as the rollout evolves. SSVR keeps language for goals and constraints, while an action-conditioned GRU maintains the recurrent planning state.

**Pixel-space visual reasoning.** Pixel-space methods construct explicit visual intermediates through drawing, annotation, programmatic editing, generated visual thoughts (Li et al., 2025b; Chern et al., 2025), or future image/state rollouts for VLA and spatial planning (Zhao et al., 2025; Wang et al., 2026; Xu et al., 2026). They preserve spatial structure and improve inspectability, but open-loop tasks often change only low-dimensional variables, such as position, carrying state, or phase, while the scene stays mostly fixed. Full image rollout therefore regenerates mostly static content; more importantly, each generated frame is a lossy state estimate whose rendering errors can propagate and compound when reused by later steps. SSVR preserves visual grounding and decouples static context from recurrent state.

**Latent-space visual reasoning.** Latent-space methods replace pixels with continuous tokens, reducing generation cost and enabling lightweight reasoning in a shared vision-language space (Wang et al., 2025; Li et al., 2025a; Liu et al., 2025a; Yang et al., 2025; Chen et al., 2026; Tang et al., 2026;

[1] https://huggingface.co/datasets/HuggingFaceM4/VQAv2

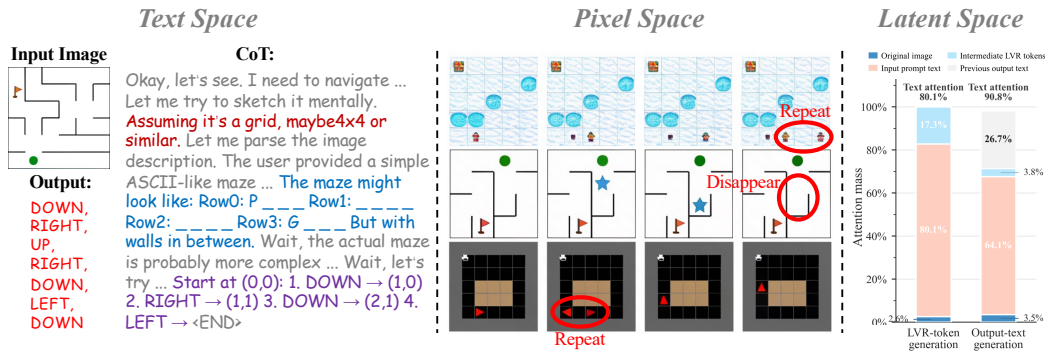



Figure 2: Representative failure modes of existing reasoning paradigms. Text-space reasoning can hallucinate Maze topology after compressing the image into language; repeatedly generated pixel-space states may duplicate or erase objects and propagate these errors; latent-space reasoning can bypass both image and intermediate latent tokens when producing the final answer.

Guo et al., 2026). Prior analyses report image-to-latent and latent-to-answer disconnections (Li et al., 2026; Yu et al., 2026), alongside representation homogenization, manifold separation, and input-output embedding mismatch under recursive use (Cui et al., 2026; Miao et al., 2026). SSVR uses a continuous recurrent state initialized from visual features, updated by an action-conditioned GRU, and injected as the query for action prediction.

## 3 Vision-Language-Action Open-Loop Planning

We study VLA open-loop planning. Given an initial image $I_0$ and instruction $x$, the model predicts an action sequence $\mathbf{a} = (a_0, \ldots, a_{H-1})$. The evaluation environment is defined as

$$\begin{aligned} \mathcal{T} &= (I_0, x, \mathcal{A}, s_0, T, g, H), \\ s_{t+1} &= T(s_t, a_t), \quad a_t \in \mathcal{A}, \end{aligned} \tag{1}$$

where $s_0$ is the initial environment state, $\mathcal{A}$ is the action set, $T$ is the environment transition, $g$ is the goal predicate, and $H$ is the planning horizon. The evaluator uses $T$ and $g$ to execute and score the plan; success requires $g(s_t) = 1$ for some $t \leq H$. In the default evaluation, $H$ is set to the ground-truth shortest-path length, and execution stops at the first terminal event.

Here, $I_0$ provides persistent scene context, $x$ specifies task objectives and constraints, and $s_t$ denotes the changing environment state. The model maintains a learned latent state $z_t$ from the initial visual features and action history. A suitable reasoning paradigm should reuse persistent context while updating its internal state according to predicted actions.

Figure 2 summarizes the resulting failures across existing paradigms, motivating a better reasoning paradigm.

## 4 State-Space Visual Reasoning

Figure 3 summarizes SSVR, which reuses static visual-textual context while recursively updating a GRU latent state.

### 4.1 Static Context and Dynamic State

Let $\theta = (\theta_0, \Delta\theta)$ collect the frozen pretrained parameters and trainable LoRA adapters, and let $\eta$ denote the trainable latent-state parameters. The decoder is $F_\theta$, and the visual frontend produces projected feature tokens $V_0 = \mathcal{E}_\theta^{\text{vis}}(I_0) \in \mathbb{R}^{N_v \times d}$. We pack these features with the instruction and template tokens into the input embedding sequence $C$ and compute the static prefix cache

$$(C, \mathcal{M}) = \text{Pack}_{\theta_0}(V_0, x), \qquad \mathcal{K} = F_\theta^{\text{prefill}}(C; \mathcal{M}). \tag{2}$$

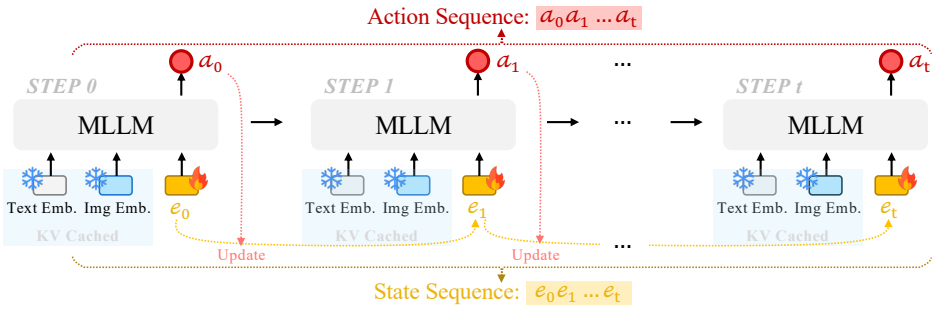



Figure 3: Overview of SSVR. The initial image and instruction are encoded once and kept as cached visual-textual context. Visual features initialize the latent state $z_0$. At step $t$, the model appends the state token $e_t = z_t$, predicts action $a_t$, and updates the latent state through an GRU.

The metadata $\mathcal{M}$ retains the attention mask and position information for decoding.

The GRU latent state $z_t \in \mathbb{R}^d$ represents the evolving planning state. Its initialization uses the mean visual feature:

$$c_0 = \frac{1}{N_v} \sum_{p=1}^{N_v} V_{0,p}, \qquad z_0 = f_\eta^{\text{init}}(V_0) = \text{LN}_\eta(q_\eta + W_\eta c_0), \qquad e_t = z_t, \tag{3}$$

where $q_\eta$, $W_\eta$, and the LayerNorm parameters are learned jointly with the GRU transition.

At step $t$, SSVR appends $e_t$ as an independent sequence position after the cached context and computes the decoder readout

$$u_t = F_\theta^{\text{decode}}(\mathcal{K}, e_t; \mathcal{M}). \tag{4}$$

Each decision reuses the same prefix cache and appended-token position. The current state token's temporary KV entries are discarded after decoding, implementing the sequence $C \parallel e_t$.

### 4.2 State-Conditioned Action Prediction

SSVR predicts one action at a time from the current latent state and cached context. Each $a \in \mathcal{A}$ is assigned a distinct single-token verbalizer, such as `UP`, `DOWN`, `LEFT`, or `RIGHT`. If $v(a)$ is its token ID, the action logit is

$$o_t(a) = \text{LMHead}_{\theta_0}(u_t)_{v(a)}, \quad a \in \mathcal{A}, \tag{5}$$

and the resulting policy is

$$\pi_{\theta,\eta}(a \mid C, z_t) = \frac{\exp(o_t(a))}{\sum_{a' \in \mathcal{A}} \exp(o_t(a'))}. \tag{6}$$

This reuses the pretrained language-model head, with probabilities normalized over the task action set.

### 4.3 Recursive State-Space Inference

With greedy decoding, SSVR alternates between state-conditioned action prediction and action-conditioned latent transition:

$$\begin{aligned} a_t &= \arg\max_{a \in \mathcal{A}} \pi_{\theta,\eta}(a \mid C, z_t), \\ z_{t+1} &= f_\eta^{\text{trans}}(z_t, a_t) = \text{LN}_\eta\big(\text{GRUCell}_\eta(E_\eta^a(a_t), z_t)\big), \qquad e_{t+1} = z_{t+1}. \end{aligned} \tag{7}$$

The learned action embedding $E_\eta^a(a_t)$ and previous latent state drive the GRU update. Initialization and transition share the LayerNorm module. This recursion produces an $H$-step plan using the fixed visual-textual context and predicted action history. The evaluator handles environment execution and terminal-event checks.

Table 1: Comparison across reasoning paradigms. Entries report EM/PR (%). VQA Support: native VQA through text-based responses; *: official release without training on our datasets.

| Model | VQA Support | FrozenLake | Maze | MiniBehavior | Avg. |
|---|---|---|---|---|---|
| *Text Space* | | | | | |
| Gemini 2.0 Flash Direct* | ✔ | 21.2 / 47.6 | 8.3 / 31.4 | 0.7 / 29.8 | 10.1 / 36.3 |
| Gemini 2.0 Flash CoT* | ✔ | 27.6 / 52.5 | 6.9 / 29.8 | 4.0 / 31.2 | 12.8 / 37.8 |
| Gemini 2.5 Pro* | ✔ | 72.0 / 85.0 | 21.5 / 35.5 | 37.6 / 59.9 | 43.7 / 60.1 |
| Qwen2.5-VL-7B Direct* | ✔ | 1.2 / 15.0 | 0.6 / 14.5 | 0.3 / 9.8 | 0.7 / 13.1 |
| Qwen2.5-VL-7B CoT* | ✔ | 8.2 / 29.1 | 2.3 / 15.2 | 0.5 / 14.7 | 3.7 / 19.7 |
| Qwen2.5-VL-7B SFT | ✔ | 68.6 / 84.4 | 60.9 / 70.3 | 31.3 / 56.1 | 53.6 / 69.9 |
| *Pixel Space* | | | | | |
| VPFT | ✗ | 75.4 / 79.5 | 59.0 / 64.0 | 33.8 / 52.2 | 56.1 / 65.2 |
| VPRL | ✗ | 91.6 / 93.2 | 74.5 / 77.6 | 75.8 / 83.8 | 80.6 / 84.9 |
| *Latent Space* | | | | | |
| LVR* | ✔ | 6.8 / 16.0 | 5.0 / 10.2 | 0.0 / 8.0 | 3.9 / 11.3 |
| Monet* | ✔ | 5.0 / 13.0 | 2.3 / 5.0 | 0.0 / 7.5 | 2.4 / 8.5 |
| *Recurrent State (Ours)* | | | | | |
| SSVR | ✔ | **99.5 / 99.6** | **96.3 / 98.0** | **83.9 / 90.6** | **93.2 / 96.1** |

### 4.4 Soft-Target Training

For each training instance, environment transitions and path-quality rules identify safe actions $\mathcal{A}_{\text{safe}}(s_t)$ and optimal actions $\mathcal{A}_{\text{opt}}(s_t)$ offline at the reference decision state $s_t$. A reference action history is constructed by BFS from the episode start, and replayed through the GRU to obtain the supervised latent state $z_t$. For $\mathcal{A}_{\text{opt}}(s_t) \subseteq \mathcal{A}_{\text{safe}}(s_t)$ and a nonempty optimal set, the soft target is

$$q_\alpha(a \mid s_t) = (1-\alpha)\,\text{Unif}(\mathcal{A}_{\text{safe}}(s_t))(a) + \alpha\,\text{Unif}(\mathcal{A}_{\text{opt}}(s_t))(a). \tag{8}$$

An empty optimal set uses the uniform safe-action target; training retains items with nonempty safe sets. SSVR minimizes

$$\mathcal{L}_{\text{SFT}} = -\sum_{a\in\mathcal{A}} q_\alpha(a \mid s_t) \log \pi_{\theta,\eta}(a \mid C, z_t). \tag{9}$$

Gradients propagate through the decoder, differentiable prefix cache, and reference-history GRU updates. Training jointly learns the LoRA adapters, latent initializer, action embeddings, and GRU transition while keeping pretrained weights frozen. The coefficient $\alpha$ balances safe-action diversity and optimal-action preference; the main model uses $\alpha = 0.7$. Implementation details and pseudocode are provided in Appendix.

## 5 Experiments

We evaluate open-loop rollout planning on FrozenLake (Wu et al., 2024a), Maze (Ivanitskiy et al., 2023), and MiniBehavior (Jin et al., 2023), covering risk avoidance, topological path finding, and phased object interaction, respectively. The model uses the initial observation and instruction, and updates its latent state with its own predicted actions. Metrics are exact match (EM) and progress rate (PR) (Xu et al., 2026), evaluated against shortest optimal action sequences; full implementation, training, and evaluation settings are in the appendix.

### 5.1 Main Results

Table 1 reports aggregate performance. SSVR achieves the highest EM and PR on all three tasks, averaging 93.2/96.1 and outperforming the strongest pixel baseline, VPRL, by 12.6/11.2 percentage points. The largest improvement occurs on Maze, where EM/PR rises from 74.5/77.6 to 96.3/98.0, a gain of 21.8/20.4 points. SSVR also achieves 99.5/99.6 on FrozenLake and 83.9/90.6 on MiniBehavior, compared with VPRL's 91.6/93.2 and 75.8/83.8, respectively. These gains are consistent with

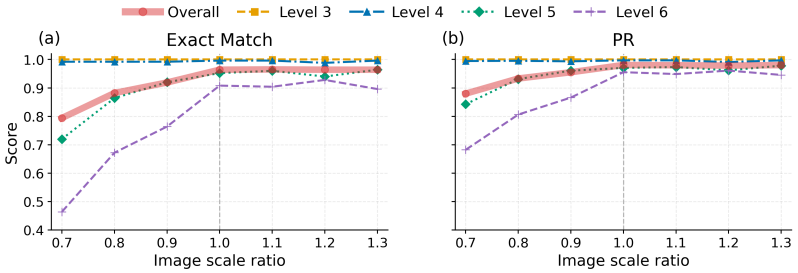



Figure 4: Maze EM and PR across image scale ratios 0.7–1.3, overall and by difficulty level.

Table 2: Prompt paraphrase robustness (%).

| Prompt | EM | PR | Legal |
|---|---|---|---|
| Default | 96.4 | 98.1 | **100.0** |
| Paraphrased | **96.8** | **98.3** | **100.0** |

Table 3: Native VQAv2 retention.

| Model | Acc. (%) | Δ (pp) |
|---|---|---|
| Native Qwen | **71.9** | 0.0 |
| SSVR | 68.0 | −3.9 |
| VPRL | ✗ | ✗ |

the benefit of conditioning decisions on an updated recurrent state, although MiniBehavior remains more challenging than the navigation tasks.

The baseline pattern is consistent with the proposed *state-representation mismatch*. In text space, verbalizing a map is a lossy compression: walls, narrow passages, and adjacency relations are easily blurred, and these omissions enter the evolving textual state. Task-specific text SFT reaches 53.6/69.9 EM/PR. In pixel space, each generated frame can introduce errors, including duplicated or disappearing objects in Figure 2. Subsequent steps condition on these generated states, allowing errors to propagate with rollout length. The evaluated latent-space baselines exhibit bypass and weak transfer. On our tasks, Monet ceases to emit latent tokens and falls back to text-only reasoning. During output generation, LVR assigns 3.5% attention to image tokens and 3.8% to intermediate latent tokens, while text tokens dominate (Figure 2), indicating limited direct attention to visual and intermediate latent representations. The released LVR and Monet models average 3.9% and 2.4% EM, respectively, in this transfer setting. SSVR learns an action-conditioned recurrent update and uses the resulting latent state for each action decision.

### 5.2 Robustness and Generalization

We evaluate SSVR across prompt paraphrasing, image transformations, cross-dataset transfer, and VQA retention.

#### 5.2.1 Input Robustness

With semantically equivalent prompt paraphrases (Table 2), EM increases from 96.4 to 96.8 and PR from 98.1 to 98.3, while the legal rate remains 100%. These small changes indicate stable performance under semantically equivalent wording.

We resize Maze inputs over ratios 0.7–1.3 to test scale robustness. Figure 4 reveals a difficulty-dependent response: Level 3 maintains 100% EM/PR throughout, and Level 4 EM remains between 98.8% and 99.6%. Shrinking substantially affects harder maps: at scale 0.7, Level-5 EM/PR falls from 95.20/97.03 to 72.00/84.26, and Level-6 EM/PR from 90.80/95.52 to 46.40/68.22, relative to scale 1.0. Performance under enlargement remains close to the native-scale baseline; at scale 1.3, Level-5 and Level-6 EM/PR are 96.40/97.81 and 89.60/94.56, respectively. This asymmetry suggests sensitivity to the loss of fine visual detail on difficult maps under downscaling, together with stable performance under enlargement.

We further invert colors and synchronously flip images, evaluator coordinates, and action labels to test sensitivity to appearance and orientation. Table 4 shows that color inversion reduces EM from

Table 4: Color/flip robustness.

| Transformation | EM | PR | Legal Rate |
|---|---|---|---|
| Original | 96.40 | 98.06 | 100.00 |
| Color inversion | $94.00^{-2.40}$ | $96.57^{-1.49}$ | $99.90^{-0.10}$ |
| Vertical flip | $96.00^{-0.40}$ | $97.62^{-0.44}$ | $100.00^{0.00}$ |
| Horizontal flip | $96.00^{-0.40}$ | $97.77^{-0.29}$ | $100.00^{0.00}$ |
| Dual-direction flip | $95.00^{-1.40}$ | $97.00^{-1.06}$ | $100.00^{0.00}$ |

Table 5: FrozenLake→Maze transfer. BB+MazeIF uses a Maze-trained interface.

| Method | Maze EM | Maze PR |
|---|---|---|
| Qwen Direct | 0.6 | 14.5 |
| Qwen CoT | 2.3 | 15.2 |
| VPRL | 6.8 | 7.2 |
| SSVR (BB+MazeIF) | <u>30.9</u> | <u>36.9</u> |
| SSVR (Full) | **33.6** | **38.3** |

Table 6: Native Maze action-choice VQA in generation mode.

| Level | Base Opt. | Trained Opt. | Base Legal | Trained Legal |
|---|---|---|---|---|
| 3 | 24.00 | $50.40^{+26.40}$ | 43.60 | $72.80^{+29.20}$ |
| 4 | 22.80 | $49.20^{+26.40}$ | 49.60 | $72.80^{+23.20}$ |
| 5 | 24.00 | $44.80^{+20.80}$ | 53.60 | $72.80^{+19.20}$ |
| 6 | 24.40 | $42.40^{+18.00}$ | 60.00 | $73.60^{+13.60}$ |
| Avg. | **23.80** | $\mathbf{46.70}^{+22.90}$ | **51.70** | $\mathbf{73.00}^{+21.30}$ |

96.40 to 94.00 and PR from 98.06 to 96.57, with the legal rate remaining at 99.90%. Vertical and horizontal flips each reduce EM by 0.40 points to 96.00, while flipping both directions reduces it by 1.40 points to 95.00; all flips retain a 100% legal rate. These modest declines indicate robustness to the tested appearance and orientation changes, with greater sensitivity to color inversion. Since these transformations preserve topology, the results are consistent with the use of relational geometry.

#### 5.2.2 Cross-Dataset Transfer

FrozenLake→Maze transfer shifts from avoiding circular holes to parsing linear wall topology. It changes both visual primitives and connectivity cues, testing transferable path finding. `Full` applies the FrozenLake-trained backbone and GRU module directly to Maze. `BB+MazeIF` combines the FrozenLake-trained backbone adapters with the Maze-trained GRU/action interface, evaluating compatibility across separately trained components.

Table 5 shows that both variants substantially outperform Qwen and VPRL. `Full` raises EM from 2.3% for Qwen-CoT and 6.8% for VPRL to 33.6%. This transfer from holes to wall topology is consistent with learned use of connectivity cues. The gains also persist on hard maps: `Full` reaches 27.2/32.1 EM/PR on Level 5 and 22.4/29.9 on Level 6, versus VPRL's 2.4/2.6 and 4.0/4.6. `BB+MazeIF` is slightly weaker than `Full`, possibly because replacing the GRU interface disrupts the joint calibration learned during training.

#### 5.2.3 VQA Retention

We disable the GRU state-conditioning pathway and evaluate the trained backbone through its original LM head on native Maze action-choice VQA and VQAv2, using an LLM judge for VQAv2 semantic correctness. Table 6 shows that average optimal-action and legal-action hit rates rise from 23.8% to 46.7% and from 51.7% to 73.0%, respectively, with gains at every difficulty level. On VQAv2 (Table 3), SSVR retains 68.0% accuracy versus native Qwen's 71.9%, a 3.9-point decrease; VPRL does not support native text-response VQA. Thus, training with the GRU state module improves the backbone's native spatial action-choice capability while largely preserving general VQA ability.

### 5.3 Ablations

We analyze two key designs: soft supervision and the action-output interface.

Table 7: Soft-target coefficient ablation on Maze.

| $\alpha$ | Maze EM | Maze PR | Entropy | VQAv2 Acc. | Paraphrase EM | Scale 0.7 EM | Scale 1.3 EM |
|---|---|---|---|---|---|---|---|
| 1.0 | 91.8 | 95.7 | 0.0045 | 67.6 | 91.7 | 68.7 | 90.5 |
| 0.7 | **96.4** | **98.1** | 0.4833 | 68.0 | **96.8** | 79.4 | **96.4** |
| 0.4 | 95.5 | 97.5 | 0.7287 | **71.1** | 94.7 | **80.0** | 95.1 |

Table 8: Action-output ablation.

| Variant | Maze EM (%) | Maze PR (%) | VQAv2 (%) |
|---|---|---|---|
| SSVR | 96.3 | 98.0 | **68.0** |
| SSVR-head | **96.4**$^{+0.1}$ | **98.5**$^{+0.5}$ | 46.9$^{-21.1}$ |

#### 5.3.1 Soft-Target Coefficient

We vary only $\alpha$ to examine the balance between safe-action and optimal-action supervision. Table 7 shows that reducing $\alpha$ from 1.0 to 0.7 increases action entropy from 0.0045 to 0.4833 and improves Maze EM/PR from 91.8/95.7 to 96.4/98.1, with gains in all other evaluated metrics. Further reducing $\alpha$ to 0.4 raises entropy to 0.7287 and VQAv2 accuracy from 68.0 to 71.1, while slightly improving EM at scale 0.7 from 79.4 to 80.0. Maze EM/PR falls to 95.5/97.5, paraphrase EM drops from 96.8 to 94.7, and EM at scale 1.3 decreases from 96.4 to 95.1. Greater safe-action weight thus yields higher policy entropy with differing effects across planning and robustness metrics. We use $\alpha = 0.7$ because it achieves the best in-domain EM/PR, paraphrase EM, and enlarged-image EM among the tested settings, while $\alpha = 0.4$ favors VQA retention and robustness to image shrinking.

#### 5.3.2 Action Output

Table 8 shows that replacing the LM action head with a linear softmax classifier, while keeping the GRU fixed, increases Maze EM/PR from 96.3/98.0 to 96.4/98.5, gains of 0.1/0.5 percentage points, and reduces VQAv2 accuracy from 68.0 to 46.9, a loss of 21.1 points. VQAv2 uses native text generation through the original LM head for both variants. Reusing the LM head during planning training therefore retains more general VQA capability in this comparison while achieving comparable planning performance.

## 6 Discussion

### 6.1 Interpretability Analysis

We use gradient-weighted attention to visualize attribution from the GRU state-token query to image tokens, conditioned on the selected action logit. Figure 5 shows changing attribution across planning steps: nearby walls in Maze, nearby holes in FrozenLake, and a shift from printer to table across the `PICK`/`DROP` phases in MiniBehavior. This analysis measures the current decision's visual readout with the latent state and prefix cache held fixed, illustrating how static context is read under the evolving recurrent state.

### 6.2 Efficiency Analysis

#### 6.2.1 Qualitative Analysis

SSVR's efficiency advantage comes from static-context reuse and compact recurrent updates. Let $N_C$ be the number of static-context tokens, $N_E = 1$ the state-token count, $N_I$ the number of intermediate visual-frame tokens, and $H$ the planning horizon. With fixed decoder width and depth, the attention cost with a static prefix KV cache is

$$\mathcal{O}\left(N_C^2 + H(N_E N_C + N_E^2)\right). \tag{10}$$

The GRU additionally costs $\mathcal{O}(Hd^2)$ for input and hidden dimension $d$. A visual planner that processes one $N_I$-token intermediate frame per step against a fixed prefix has attention cost

$$\mathcal{O}\left(N_C^2 + H(N_I N_C + N_I^2)\right). \tag{11}$$

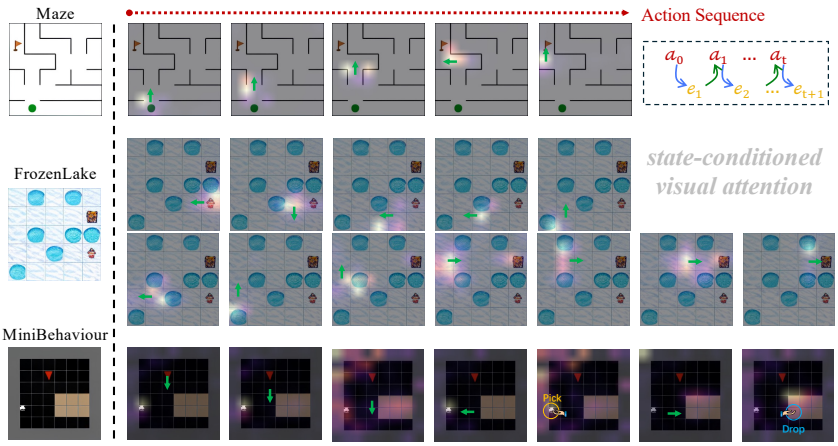



Figure 5: Gradient-weighted visual attribution from the GRU state-token query to image tokens. As the recurrent latent state changes, attribution highlights next-action-relevant regions: nearby walls in Maze, nearby holes in FrozenLake, and a phase-dependent shift from printer to table in MiniBehavior.

Table 9: Efficiency on Maze `case0`. †: KV cache; latency excludes preprocessing and, for SSVR†, prefix-cache construction. Multipliers are relative to SSVR†.

| Method | Step latency (ms) | Rollout latency (ms) | Throughput (cases/s) | Peak memory (GiB) |
|---|---|---|---|---|
| SSVR† | **45.611** (1.00x) | **182.442** (1.00x) | **5.481** (1.00x) | 15.918 (1.00x) |
| SSVR | 89.159 (1.95x) | 356.634 (1.95x) | 2.804 (0.51x) | 15.937 (1.00x) |
| LVR | 529.596 (11.61x) | 1059.193 (5.81x) | 0.944 (0.17x) | 17.609 (1.11x) |
| Monet | 1743.966 (38.24x) | 6975.866 (38.24x) | 0.143 (0.03x) | 15.705 (0.99x) |
| VPRL | 4496.111 (98.58x) | 17984.442 (98.58x) | 0.056 (0.01x) | **13.331** (0.84x) |

These expressions describe attention over the stated token sequences; total runtime also includes projections, feed-forward layers, and visual processing. Since $N_E \ll N_I$, SSVR reduces the per-step token workload while reusing the visual-textual prefix.

### 6.2.2 Quantitative Analysis

We benchmark all methods on the same Maze `case0` sample over 20 repetitions after warm-up, using CUDA events for latency and the PyTorch CUDA memory API for peak memory. Image/text preprocessing and prefix-cache construction are measured separately; cached SSVR rollout timing covers decoding with the prebuilt cache and recurrent state updates.

As shown in Table 9, enabling KV cache reduces SSVR step latency from 89.159 ms to 45.611 ms and decoding-rollout latency from 356.634 ms to 182.442 ms, giving about a $1.95\times$ speedup with nearly unchanged peak memory. Compared with LVR, Monet, and VPRL, cached SSVR has $5.81\times$, $38.24\times$, and $98.58\times$ lower rollout latency under this timing protocol, illustrating the computational benefit of reusing static visual context during recurrent planning.

## 7 Conclusion

This paper identifies *state-representation mismatch* as an obstacle in VLA open-loop planning: existing paradigms can encode evolving state through lossy textual compression, error-accumulating pixel rollouts, or bypass-prone latent tokens, even though most visual context remains static. SSVR addresses this mismatch by encoding the initial visual-textual context once and recursively updating a learned latent state through an action-conditioned GRU. Across three planning tasks, this design yields strong in-domain performance, robust behavior under linguistic and visual transformations, cross-dataset transfer, and efficient cached decoding while largely retaining general VQA ability. These results support action-conditioned recurrent state modeling as a useful design principle for VLA open-loop planning.

## AI-Assisted Content Disclosure

Generative AI tools assisted with linguistic polishing and technical revision of the manuscript, including method descriptions and mathematical notation. The authors are responsible for verifying the revised text against the implementation and experimental records and for all claims, content, and findings in the paper.

## References

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Wang. Agent s: An open agentic framework that uses computers like a human. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 22924–22946, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/394c7c30ea87b5c3521b4d9e9d419071-Paper-Conference.pdf.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Mahtab Bigverdi, Zelun Luo, Cheng-Yu Hsieh, Ethan Shen, Dongping Chen, Linda G. Shapiro, and Ranjay Krishna. Perception tokens enhance visual reasoning in multimodal language models, 2024. URL https://arxiv.org/abs/2412.03548.

Chao Chen, Zhixin Ma, Yongqi Li, Yupeng Hu, Yinwei Wei, Wenjie Li, and Liqiang Nie. Reasoning in the dark: Interleaved vision-text reasoning in latent space. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 39117–39129, 2026.

Ethan Chern, Zhulin Hu, Steffi Chern, Siqi Kou, Jiadi Su, Yan Ma, Zhijie Deng, and Pengfei Liu. Thinking with generated images, 2025. URL https://arxiv.org/abs/2505.22525.

Jin Cui, Xinyue Long, Xunyong Zhang, Yadong Zhang, Chuanchang Su, Jingye Gan, Boran Zhao, and Pengju Ren. Retrieve, integrate, and synthesize: Spatial-semantic grounded latent visual reasoning, 2026. URL https://arxiv.org/abs/2605.07106.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Ziyu Guo, Rain Liu, Xinyan Chen, and Pheng Ann Heng. Atlas: Agentic or latent visual reasoning? one word is enough for both. *arXiv preprint*, 2026.

Yilun Hao, Yongchao Chen, Chuchu Fan, and Yang Zhang. Simulation to rules: A dual-vlm framework for formal visual planning, 2026. URL https://arxiv.org/abs/2510.03182.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. Deepeyesv2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025.

Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning, 2023. URL https://arxiv.org/abs/2311.17842.

Michael Igorevich Ivanitskiy, Rusheb Shah, Alex F. Spies, Tilman Räuker, Dan Valentine, Can Rager, Lucia Quirke, Chris Mathwin, Guillaume Corlouer, Cecilia Diniz Behn, and Samy Wu Fung. A configurable library for generating and manipulating maze datasets, 2023. URL https://arxiv.org/abs/2309.10498.

Yitao Jiang, Roy Xing, Luyang Zhao, Brian Plancher, Muhao Chen, and Devin Balkcom. Ascii art turns llms into vla controllers, 2026a. URL https://arxiv.org/abs/2606.21470.

Zhonghua Jiang, Kunxi Li, Yiyun Zhou, Sihao Liu, Zhaode Wang, Chengfei lv, and Shengyu Zhang. Purekv: Plug-and-play kv cache optimization with spatial-temporal sparse attention for vision-language large models, 2025. URL https://arxiv.org/abs/2510.25600.

Zhonghua Jiang, Kui Chen, Kunxi Li, Keting Yin, Yiyun Zhou, Zhaode Wang, Chengfei Lv, and Shengyu Zhang. Acckv: Towards efficient audio-video llms inference via adaptive-focusing and cross-calibration kv cache optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 5494–5502, 2026b.

Emily Jin, Jiaheng Hu, Zhuoyi Huang, Ruohan Zhang, Jiajun Wu, Li Fei-Fei, and Roberto Martín-Martín. Mini-behavior: A procedurally generated benchmark for long-horizon decision-making in embodied ai, 2023. URL https://arxiv.org/abs/2310.01824.

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning, 2025a. URL https://arxiv.org/abs/2509.24251.

Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought, 2025b. URL https://arxiv.org/abs/2501.07542.

You Li, Chi Chen, Yanghao Li, Fanhu Zeng, Kaiyu Huang, Jinan Xu, and Maosong Sun. Imagination helps visual reasoning, but not yet in latent space. In *Forty-third International Conference on Machine Learning*, 2026.

Chengzhi Liu, Yuzhe Yang, Yue Fan, Qingyue Wei, Sheng Liu, and Xin Eric Wang. Reasoning within the mind: Dynamic multimodal interleaving in latent space. *arXiv preprint arXiv:2512.12623*, 2025a.

Dairu Liu, Ziyue Wang, Minyuan Ruan, Fuwen Luo, Chi Chen, Peng Li, and Yang Liu. Thinking with visual abstract: Enhancing multimodal reasoning via visual abstraction, 2025b. URL https://arxiv.org/abs/2505.20164.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Jingping Liu, Ziyan Liu, Zhedong Cen, Yan Zhou, Yinan Zou, Weiyan Zhang, Haiyun Jiang, and Tong Ruan. Can multimodal large language models understand spatial relations? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 620–632, 2025c.

Yanting Miao, Yutao Sun, Dexin Wang, Mengyu Zhou, Pascal Poupart, Lei Lv, Qi Zhao, Li Wang, Hao Li, Xiaoxi Jiang, and Guanjun Jiang. Fill the gap: A granular alignment paradigm for visual reasoning in multimodal large language models, 2026. URL https://arxiv.org/abs/2605.12374.

Dhruv Shah, Błażej Osiński, brian ichter, and Sergey Levine. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 492–504. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/shah23b.html.

Yuesen Tang, Yiming Yang, Tengfei Bao, and Yu Tong. Thinking in latent space: Progressive multimodal simplification for visual reasoning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=5IA4cx8Gzz.

Constantin Venhoff, Ashkan Khakzar, Sonia Joseph, Philip Torr, and Neel Nanda. How visual representations map to language feature space in multimodal llms. *arXiv preprint arXiv:2506.11976*, 2025.

Chaoyang Wang, Wenrui Bao, Sicheng Gao, Bingxin Xu, Yu Tian, Yogesh Singh Rawat, Yunhao Ge, and Yuzhang Shang. Vla-thinker: Boosting vision-language-action models through thinking-with-image reasoning. *ArXiv*, abs/2603.14523, 2026. URL https://api.semanticscholar.org/CorpusID:286572547.

Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond images and language, 2025. URL https://arxiv.org/abs/2511.21395.

Lai Wei, Liangbo He, Jun Lan, Lingzhong Dong, Yutong Cai, Siyuan Li, Huijia Zhu, Weiqiang Wang, Linghe Kong, Yue Wang, Zhuosheng Zhang, and Weiran Huang. Zooming without zooming: Region-to-image distillation for fine-grained multimodal perception, 2026. URL https://arxiv.org/abs/2602.11858.

Qiucheng Wu, Handong Zhao, Michael Saxon, Trung Bui, William Yang Wang, Yang Zhang, and Shiyu Chang. Vsp: Assessing the dual challenges of perception and reasoning in spatial planning tasks for vlms, 2024a. URL https://arxiv.org/abs/2407.01863.

Wenshan Wu, Shaoguang Mao, Yadong Zhang, Yan Xia, Li Dong, Lei Cui, and Furu Wei. Mind's eye of llms: Visualization-of-thought elicits spatial reasoning in large language models, 2024b. URL https://arxiv.org/abs/2404.03622.

Yi Xu, Chengzu Li, Han Zhou, Xingchen Wan, Caiqi Zhang, Anna Korhonen, and Ivan Vulić. Visual planning: Let's think only with images. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=wsnse46kRO.

Zeyuan Yang, Xueyang Yu, Delin Chen, Maohao Shen, and Chuang Gan. Machine mental imagery: Empower multimodal reasoning with latent visual tokens, 2025. URL https://arxiv.org/abs/2506.17218.

Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Guanting Dong, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, Guibin Zhang, Jiale Tao, Jiayi Zhang, Siyuan Ma, Kaituo Feng, Haojie Huang, Youxing Li, Ronghao Chen, Huacan Wang, Chenglin Wu, Zikun Su, Xiaogang Xu, Kelu Yao, Kun Wang, Chen Gao, Yue Liao, Ruqi Huang, Tao Jin, Zhucun Xue, Cheng Tan, Jiangning Zhang, Wenqi Ren, Yanwei Fu, Yong Liu, Yu Wang, Xiangyu Yue, Yu-Gang Jiang, and Shuicheng Yan. The latent space: Foundation, evolution, mechanism, ability, and outlook, 2026. URL https://arxiv.org/abs/2604.02029.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713, 2025. doi: 10.1109/CVPR52734.2025.00166.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning, 2026. URL https://arxiv.org/abs/2505.14362.

# A Evaluation Metrics

We use Exact Match (EM) and Progress Rate (PR) to evaluate open-loop plans. Consider an instance with $M$ shortest optimal action sequences of length $n > 0$,

$$\mathbf{a}^{(m)} = (a_1^{(m)}, \ldots, a_n^{(m)}), \qquad m \in \{1, \ldots, M\}, \tag{12}$$

where $a_j^{(m)}$ is the $j$-th action of optimal sequence $m$. Let the predicted action sequence under the evaluation budget $H = n$ be $\hat{\mathbf{a}} = (\hat{a}_1, \ldots, \hat{a}_n)$; missing actions are treated as mismatches. The instance-level EM is

$$\mathrm{EM} = \max_{m \in \{1,\ldots,M\}} \prod_{j=1}^{n} \mathbb{I}\left[\hat{a}_j = a_j^{(m)}\right]. \tag{13}$$

Thus, EM is one if the prediction exactly follows any shortest optimal action sequence and zero otherwise. The maximum accounts for all equally short optimal action sequences.

PR measures the longest optimal prefix:

$$\mathrm{PR} = \max_{m \in \{1,\ldots,M\}} \frac{1}{n} \sum_{j=1}^{n} \prod_{k=1}^{j} \mathbb{I}\left[\hat{a}_k = a_k^{(m)}\right]. \tag{14}$$

If the first $\ell$ actions form an optimal prefix and the first error occurs at action $\ell{+}1$, then $\mathrm{PR} = \ell/n$. In the deterministic, unit-cost evaluation environments, an action is optimal when its transition is valid and reduces the task-specific shortest-path distance-to-goal by exactly one; successful termination has distance zero. This distance-based test identifies prefixes of shortest optimal action sequences. Dataset-level EM and PR are the arithmetic means of the corresponding instance-level scores and are reported as percentages in the main paper.

# B Training and Inference Algorithms

Algorithms 1 and 2 implement the method in Section 4, using $\theta = (\theta_0, \Delta\theta)$ and latent-state parameters $\eta$. The initializer $f_\eta^{\mathrm{init}}$ and transition $f_\eta^{\mathrm{trans}}$ follow Equations 3 and 7.

## B.1 Implementation Details

**Packing and batching.** The metadata $\mathcal{M}$ contains the prefix attention mask, multimodal position indices, appended-token position, and image-token key indices. Visual pooling includes valid image tokens. Padded reference actions preserve the latent state during batched history replay.

**Cache handling.** An in-place cache is restored to the prefix length $N_C$ after each decode, preserving tensors required for training gradients. Training computes fresh differentiable caches for each minibatch. A full-sequence forward on $C \,\|\, e_t$ provides an equivalent formulation in evaluation mode, where dropout is disabled.

**Action-head ablation.** The classifier variant uses $(W_\omega u_t + b_\omega)_a$ for the action logit and includes $\omega$ in the optimizer and saved model. The GRU update and soft-target supervision follow the main method.

## B.2 Training Pseudocode

The training set $\mathcal{D}$ contains tuples $(I_0, x, \mathbf{a}_{<t}^{\mathrm{ref}}, \mathcal{A}_{\mathrm{safe}}, \mathcal{A}_{\mathrm{opt}})$. Each tuple supervises one reachable decision state; an empty history denotes the initial decision. The safe set is the base admissible set after the supervision protocol's action filters. Algorithm 1 averages the per-item loss in Equation 9 over the minibatch. In the pseudocode, $q_\alpha(a)$ denotes the target at the reference decision state of the current item.

**Algorithm 1** Soft-target training with a GRU latent state

**Require:** $\mathcal{D}$ as defined above; frozen pretrained parameters $\theta_0$; trainable LoRA parameters $\Delta\theta$; latent initializer $f_\eta^{\text{init}}$; transition $f_\eta^{\text{trans}}$; injective action verbalizer $v$; $\alpha \in [0, 1]$

**Ensure:** Trained $\Delta\theta$ and $\eta$

Initialize $\Delta\theta$ and $\eta$; keep $\theta_0$ frozen
**for** each training minibatch $\mathcal{B} \subset \mathcal{D}$ **do**
  Clear optimizer gradients; $\mathcal{L} \leftarrow 0$
  **for** each $(I_0, x, \mathbf{a}_{<t}^{\text{ref}}, \mathcal{A}_{\text{safe}}, \mathcal{A}_{\text{opt}}) \in \mathcal{B}$ **do**
    $V_0 \leftarrow \mathcal{E}_\theta^{\text{vis}}(I_0)$
    $(C, \mathcal{M}) \leftarrow \text{Pack}_{\theta_0}(V_0, x)$
    $\mathcal{K} \leftarrow F_\theta^{\text{prefill}}(C; \mathcal{M})$ ▷ retain the training computation graph
    $z \leftarrow f_\eta^{\text{init}}(V_0)$
    **for** $k = 0, \ldots, t-1$ **do**
      $z \leftarrow f_\eta^{\text{trans}}(z, a_k^{\text{ref}})$ ▷ differentiable reference-history replay
    **end for**
    $e \leftarrow z$
    $u \leftarrow F_\theta^{\text{decode}}(\mathcal{K}, e; \mathcal{M})$ ▷ read-only prefix cache; one appended token
    **for** each action $a \in \mathcal{A}$ **do**
      $o(a) \leftarrow \text{LMHead}_{\theta_0}(u)_{v(a)}$
    **end for**
    $\pi_{\theta,\eta}(a \mid C, z) \leftarrow \text{softmax}_{a \in \mathcal{A}}(o(a))$
    Construct $q_\alpha$ using Equation 8 and its empty-optimal-set fallback
    $\mathcal{L} \leftarrow \mathcal{L} - \sum_{a \in \mathcal{A}} q_\alpha(a) \log \pi_{\theta,\eta}(a \mid C, z)$
  **end for**
  Backpropagate $\mathcal{L}/|\mathcal{B}|$ through decoding, prefill, and all replayed GRU transitions
  Update only $(\Delta\theta, \eta)$ using the optimizer
  Discard all minibatch KV caches and recurrent-state computation graphs
**end for**
**return** $\Delta\theta, \eta$

### B.3 Inference Pseudocode

Algorithm 2 uses the horizon and evaluation protocol in Section 3. Greedy selection uses a fixed tie-breaking rule. Every predicted action drives the GRU transition, including actions marked invalid by the evaluator.

**Algorithm 2** Open-loop GRU rollout with a read-only visual–textual KV cache

**Require:** Initial image $I_0$; instruction $x$; planning horizon $H$; trained parameters $(\theta_0, \Delta\theta, \eta)$; action verbalizer $v$
**Ensure:** Action plan $\mathbf{a}$ of length $H$

Set all modules to evaluation mode; disable gradients for ordinary rollout
$V_0 \leftarrow \mathcal{E}_\theta^{\mathrm{vis}}(I_0)$
$(C, \mathcal{M}) \leftarrow \mathrm{Pack}_{\theta_0}(V_0, x)$
$\mathcal{K} \leftarrow F_\theta^{\mathrm{prefill}}(C; \mathcal{M})$ ▷ compute the static prefix once
$z \leftarrow f_\eta^{\mathrm{init}}(V_0)$; $\mathbf{a} \leftarrow [\,]$
**for** $t = 0, \ldots, H-1$ **do**
  $e_t \leftarrow z$
  $u_t \leftarrow F_\theta^{\mathrm{decode}}(\mathcal{K}, e_t; \mathcal{M})$ ▷ same prefix and appended position; discard temporary state KV
  **for** each action $a \in \mathcal{A}$ **do**
    $o_t(a) \leftarrow \mathrm{LMHead}_{\theta_0}(u_t)_{v(a)}$
  **end for**
  $a_t \leftarrow \arg\max_{a \in \mathcal{A}} o_t(a)$ ▷ use a fixed tie-breaking rule
  Append $a_t$ to $\mathbf{a}$
  $z \leftarrow f_\eta^{\mathrm{trans}}(z, a_t)$ ▷ learned action-conditioned latent update
**end for**
**return** $\mathbf{a}$

## C State-Conditioned Visual Attribution Analysis

This section defines visual attribution for the GRU latent-state variant under the same projected-image-token and cached-decoding assumptions. At step $t$, the conditioning state is the recurrent vector $z_t$ derived from the initial image features and the predicted action history.

### C.1 Gradient-Weighted Attribution Method

**Attention indexing under cached decoding.** Let $\mathcal{P} \subset \{0, \ldots, N_C - 1\}$ be the image-token *key positions* in the packed static prefix. During a one-token cached decode, each decoder layer has one current query and $N_C + 1$ keys, including the current state token. For decoder layer $\ell$ and query-attention head $j$, write the relevant attention matrix as

$$A_t^{\ell,j} \in \mathbb{R}^{1 \times (N_C+1)}. \tag{15}$$

The state query has local row index 0 and full-sequence position $N_C$. The state-to-image attention is

$$a_{t,p}^{\ell,j} = A_t^{\ell,j}[0, p], \qquad p \in \mathcal{P}. \tag{16}$$

For a full-sequence forward on $C \,\|\, e_t$, the query row is $N_C$. Image-token indices follow the packed key order, including interleaved special and text tokens. In grouped-query attention, the averaging below uses query-attention heads.

**Diagnostic forward with gradients.** Ordinary rollout uses inference without gradients. For attribution, we snapshot the pre-action latent state $z_t$, retain the same static prefix and selected action, and perform a separate diagnostic one-token decode with gradient tracking enabled. The diagnostic pass holds model parameters and the stored rollout state fixed, with dropout disabled. The state snapshot and prefix KV cache are detached from the rollout graph, and a fresh state-embedding leaf enables gradient tracking for the diagnostic decode.

The diagnostic forward exposes the differentiable attention probabilities used in the attention-value product. An explicit attention implementation supports this gradient calculation. Numerical agreement with the ordinary decoder is checked when switching attention implementations, and the explained action remains fixed to the rollout selection. Temporary state-token KV entries are discarded after decoding.

**Action-conditioned gradient.** Let the action selected by the ordinary rollout be

$$\hat{a}_t = \arg\max_{a \in \mathcal{A}} o_t(a). \tag{17}$$

The selected action $\hat{a}_t$ is held fixed during differentiation. Let $o_t^{\mathrm{diag}}(\hat{a}_t)$ denote the corresponding pre-softmax action logit in the diagnostic forward. Its derivative with respect to the attention tensor is

$$g_{t,p}^{\ell,j} = \left[ \frac{\partial o_t^{\mathrm{diag}}(\hat{a}_t)}{\partial A_t^{\ell,j}} \right]_{0,p}, \qquad p \in \mathcal{P}. \tag{18}$$

Computationally, the gradient is obtained with respect to the attention tensor used by the forward pass and is then indexed at $[0, p]$.

We retain positive gradient-weighted attention and average it over the selected decoder layers $\mathcal{L}$ and their query-attention heads:

$$r_t(p) = \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} \frac{1}{N_h^\ell} \sum_{j=1}^{N_h^\ell} \left[ a_{t,p}^{\ell,j} g_{t,p}^{\ell,j} \right]_+, \qquad [z]_+ = \max(z, 0). \tag{19}$$

Here $N_h^\ell$ is the number of query-attention heads at layer $\ell$; if it is constant across layers, this reduces to division by $|\mathcal{L}| N_h$. A positive product indicates positive local sensitivity of the selected action logit to the corresponding attention entry, weighted by the attention magnitude.

For visualization, the scores are normalized separately at each step:

$$\widetilde{r}_t(p) = \frac{r_t(p)}{\max_{p' \in \mathcal{P}} r_t(p') + \epsilon}, \qquad \epsilon > 0. \tag{20}$$

If every score is zero, the displayed attribution map is zero. The scores are mapped to the spatial grid of the *actual decoder-visible image feature tokens*, accounting for visual token merging or spatial reordering. They are then bilinearly upsampled to the initial-image resolution and alpha-blended with $I_0$. Per-step normalization displays relative attribution within each panel.

**Interpretation and scope.** All panels within a trajectory use the same initial image, instruction, and static-prefix KV cache. Across steps, the GRU latent state $z_t$, its appended token $e_t = z_t$, the decoder readout, and the selected action can change. The diagnostic backward pass holds model parameters and the stored rollout state fixed.

Conditioned on fixed $z_t$ and prefix keys and values, this analysis describes the current decision's state-to-image-token readout through local sensitivity of the selected action logit.

Accordingly, $\widetilde{r}_t$ is a *latent-state- and action-conditioned attribution map*. The first panel of each row shows the unchanged initial observation; subsequent panels display the attribution overlay and selected action.

### C.2 Qualitative Results across Tasks

We next examine complete trajectories from FrozenLake, Maze, and MiniBehavior. These tasks respectively test whether visual attribution follows changing local hazards, wall-constrained topology, and phase-dependent object interactions while the visual input remains fixed.

Figure 6 contains examples with different map sizes, start locations, goals, and path lengths. In FrozenLake, high attribution frequently appears around hazards adjacent to the position reached by the predicted actions, which are relevant to avoiding an immediately failing transition. In Maze, attribution tracks the relevant wall segment or opening as the predicted plan progresses. Highlighted regions can include neighboring obstacles and corridors relevant to the current decision.

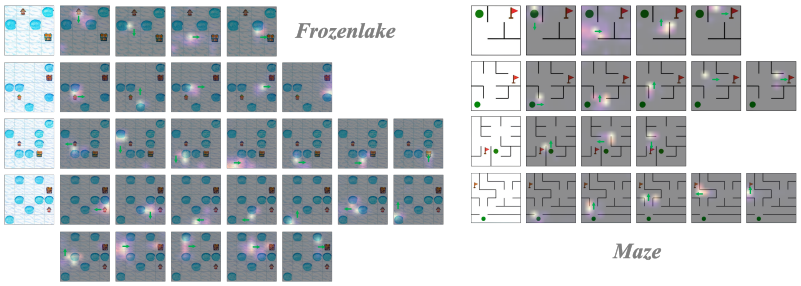



Figure 6: Additional FrozenLake and Maze trajectories. In FrozenLake (left), attribution emphasizes nearby holes and safe local passages as the GRU latent state evolves with the predicted actions. In Maze (right), it moves across the fixed map and concentrates on relevant walls and openings. The shift across steps is consistent with latent-state-conditioned reading of a persistent scene.




Figure 7: Additional MiniBehavior trajectories. Each row follows a complete two-phase plan: the agent first navigates to the printer and executes `PICK`, then navigates to the table and executes `DROP`. The GRU updates its latent state from the predicted action history. Attribution correspondingly shifts from the printer and its approach path to the table and its approach path.

MiniBehavior further tests whether attribution changes with the task phase represented implicitly through the action history. As shown in Figure 7, the pre-pick panels emphasize the printer and navigational constraints on the way to it. After `PICK`, the visual input remains fixed, and the GRU latent state is updated using the selected action; attribution then moves toward the table and relevant boundaries before `DROP`. This phase-dependent change provides qualitative evidence that the same scene is re-read according to the evolving latent state.

# D Implementation and Experimental Details

## D.1 Task Interfaces and Data

FrozenLake[2] and Maze[3] use four actions, `UP`, `DOWN`, `LEFT`, and `RIGHT`. Their model state is a GRU latent vector updated from the action history. MiniBehavior[4] additionally uses `PICK` and `DROP`, with carrying information modeled implicitly in the same recurrent state.

Initial-map tokens are decoded once with the MUSE VQ-VAE into $256 \times 256$ images and cached on disk; Qwen's native image processor then constructs the multimodal input. Rollout uses the initial observation and the recurrent state updated from predicted actions.

## D.2 Optimization

All main GRU-based models initialize the backbone from Qwen2.5-VL-7B-Instruct. The pretrained backbone weights are frozen and adapted using LoRA, while the GRU state module is learned

[2] https://huggingface.co/yixu1/VPRL-7B-FrozenLake
[3] https://huggingface.co/yixu1/VPRL-7B-Maze
[4] https://huggingface.co/yixu1/VPRL-7B-MiniBehaviour

Table 10: Dataset-specific state and action interfaces.

| Task | Model state | Action tokens |
|---|---|---|
| FrozenLake | $z_t \in \mathbb{R}^{3584}$ | U/D/L/R |
| Maze | $z_t \in \mathbb{R}^{3584}$ | U/D/L/R |
| MiniBehavior | $z_t \in \mathbb{R}^{3584}$ | U/D/L/R/PICK/DROP |

jointly. LoRA is applied to the attention projections and MLP projections listed in Table 11. We use the soft target from the main paper with $\alpha = 0.7$.

Table 11: Main training and evaluation configuration. The learning-rate scheduler is constructed for ten epochs, while training is stopped after epoch five.

| Setting | Value |
|---|---|
| Backbone | Qwen2.5-VL-7B-Instruct |
| Trainable adaptation | LoRA + GRU state module |
| LoRA rank / `lora_alpha` / dropout | 32 / 64 / 0.1 |
| LoRA target modules | `q/k/v/o_proj`, `gate/down/up_proj` |
| Optimizer | fused AdamW |
| Peak learning rate | $1.5 \times 10^{-4}$ |
| Schedule / warmup | cosine / 10% of scheduled steps |
| Scheduled / executed epochs | 10 / 5 |
| Per-device batch size | 16 |
| Gradient accumulation | 1 |
| Number of training processes | 8 |
| Training hardware | 8× NVIDIA H800 GPUs |
| Effective global batch size | 128 |
| Training precision | bfloat16 mixed precision with TF32 enabled |
| Initialization / Trainer seed | 2026 / 42 |
| Checkpointing | once per epoch |
| Soft-target coefficient $\alpha$ | 0.7 |
| GRU input / hidden dimension | 3584 / 3584 |
| Image reconstruction resolution | $256 \times 256$ |
| Evaluation decoding | greedy over task action tokens |
| Evaluation environment dynamics | deterministic, no slip |

All models are trained on eight NVIDIA H800 GPUs with one distributed training process per GPU. The per-device batch size is 16 with one gradient accumulation step, giving an effective global batch size of 128. Training uses bfloat16 mixed precision, fused AdamW, a peak learning rate of $1.5 \times 10^{-4}$, and a cosine schedule with a 10% warmup. The schedule is parameterized for ten epochs, and the reported checkpoint is saved after epoch five.