LOOPING VISUAL TOKENS FOR SCALING MULTIMODAL INTELLIGENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce *Looping Visual Tokens* (LVT), a mechanism for learning visual and spatial understanding through recurrent computation over visual tokens. LVT uses a shared transformer block to iteratively update a persistent visual state at the native image-token positions of a vision-language model (VLM). By reusing the same parameters and token positions across iterations, the block deepens visual computation without expanding the token sequence. We train this recurrent computation by aligning successive states with the backbone's perceptual representations of intermediate observations, and pass the final state to the model's task readout. Human videos and continuous environment-state sequences from simulation can provide supervision at scale, connecting LVT to learning from visual experience. We evaluate LVT on visual and spatial reasoning tasks. On fresh VSP maps beyond training path lengths, ordered supervision achieves 97.0% closed-loop success, compared with 93.2% for final-state supervision and 64.3% for answer-only recurrence. For unseen rotations at prescribed horizons, ordered supervision improves accuracy from 79.1% to 92.1%, while visual supervision alone reaches 87.2%. Textual process supervision remains stronger on the training grid size, while LVT generalizes better to larger grids. Decoding and interventions examine visual structure and state use in prediction. These results motivate LVT as a scalable mechanism for general-purpose visual learning. Anonymous repository: `https://anonymous.4open.science/r/LoopVLM-6F55/`.

1 INTRODUCTION

GPT-6 Astra has recently demonstrated strong spatial understanding in constructing and revising detailed 3D scenes (Ricouard, 2026). Such capabilities motivate learning mechanisms for reasoning about changes in the visual world. Mentally rotating an object or constructing a route across a map involves intermediate spatial configurations. Learning to represent these configurations connects visual understanding with multi-step reasoning.

Looped Transformers implement iterative computation through repeated application of shared parameters. Universal Transformers introduced recurrent refinement across depth (Dehghani et al., 2019), and subsequent work demonstrated its value for reasoning (Saunshi et al., 2025). Huginn scales inference computation through a shared latent update (Geiping et al., 2025), while LoopViT extends this principle to visual reasoning (Shu et al., 2026). Adjustable depth decouples computation from parameter count. In a pretrained vision-language model (VLM), this formulation raises the problem of defining and supervising the visual states carried between iterations.

We introduce *Looping Visual Tokens* (LVT), which uses observation sequences to supervise recurrent visual computation. A frozen VLM encodes intermediate observations into perceptual targets. A shared block updates the model's native image-token states to match these targets at the same layer boundary (Fig. 1). The updated states pass to the remaining layers for task prediction, connecting the learned visual process to the pretrained readout. Repeated updates provide additional computation over fixed state positions, and observation sequences supply process supervision without textual descriptions of the transformations.

We evaluate visual reasoning on 2D grid maps and spatial reasoning through mental rotation of 3D objects. On fresh VSP maps beyond training path lengths, answer-only recurrence, final-state supervision, and ordered supervision reach 64.3, 93.2, and 97.0% success under the same closed-loop

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.

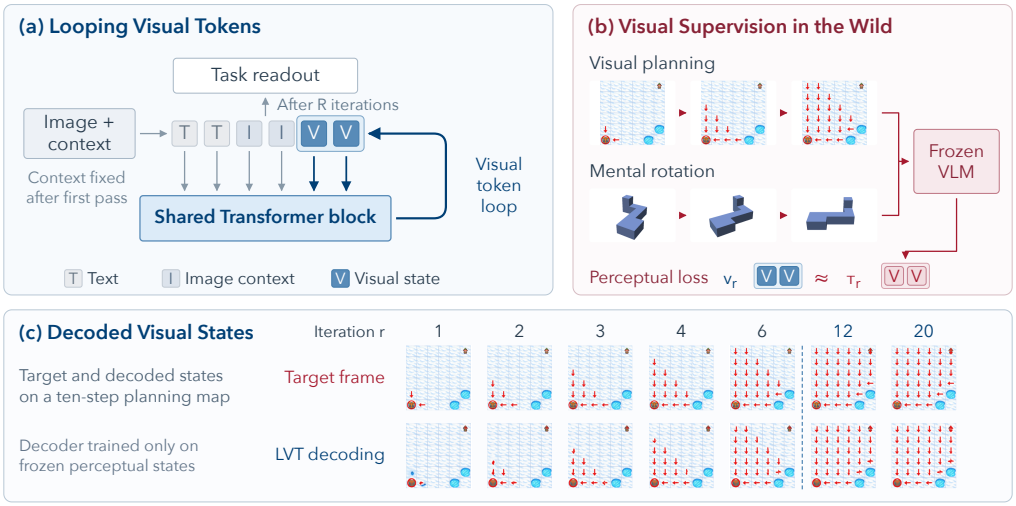


Figure 1: **Looping Visual Tokens (LVT). (a)** A shared block updates selected visual tokens in place, reusing weights and positions. Context stays fixed after the first pass; input encoding and injection are omitted. **(b)** A frozen VLM encodes rendered planning states and successive object views as perceptual targets. **(c)** Target frames and decoded LVT states on a ten-step planning map. The decoder is trained only on frozen perceptual states; the divider marks the six-step training horizon.

protocol of acting, re-observing, and resetting the model. Textual process supervision performs better on the training grid size, while LVT generalizes better to larger grids, where the text control makes transcription errors. On mental rotation, ordered supervision raises unseen-angle accuracy from 79.1 to 92.1% at prescribed horizons; visual supervision alone reaches 87.2%.

Our contributions are threefold. First, LVT connects recurrent visual states to both perceptual supervision and a pretrained task readout. Second, controlled comparisons identify the benefits of state supervision and observation order under longer transformations. Third, decoding and interventions distinguish visual state content from its use in prediction.

2 Related Work

Perceptual prediction as supervision. Predictive visual learning supervises representations through relationships among observations (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). PEARL applies predictive embedding supervision to VLMs, predicting a representation of an entire expert tool-use trajectory (Adhikari & Lapata, 2026). LVT aligns recurrent image positions with individual observation states and passes the updated sequence to the task readout.

Recurrent and latent reasoning. Shared-depth models include Universal Transformers, Huginn, and LoopViT (Dehghani et al., 2019; Geiping et al., 2025; Shu et al., 2026). Related work studies extrapolation from small instances (Schwarzschild et al., 2021; Bansal et al., 2022; Fan et al., 2025), recurrent planning (Tamar et al., 2016; Wang et al., 2025; Jolicoeur-Martineau, 2025), and halting (Graves, 2016; Banino et al., 2021). Looped World Models refine latent environment states for simulation (Lu et al., 2026). Coconut, Mirage, LVR, CoVT, and Monet introduce additional latent sequence positions (Hao et al., 2025; Yang et al., 2026; Li et al., 2026a; Qin et al., 2026; Wang et al., 2026). CVRR recurrently updates an image-conditioned question state while re-reading fixed visual evidence (Park et al., 2026). LVT makes the existing image positions mutable and supervises their progression with perceptual targets. The same image positions therefore serve as perceptual supervision targets and as the state carried between updates. We compare with a Mirage reimplementation under a matched training-step budget and examine state use through interventions (Secs. 4.2 and 5.2).

Visual state analysis. Feature inversion and representation autoencoders inspect hidden states (Dosovitskiy & Brox, 2016; Zheng et al., 2026). Recoverable content and its use in prediction are distinct questions (Li et al., 2026b; Park et al., 2026); we address them with fixed perceptual decoders and state interventions, respectively.

3 Looping Visual Tokens

LVT uses observations of a visual procedure to supervise recurrent computation. Given an initial image I_0 and context c, the model repeatedly updates the hidden states at selected image positions, then uses the resulting sequence to predict a task answer. Training images $I_1, \ldots, I_m$ specify the intended progression through a frozen perceptual encoder (Fig. 1). Observation sequences specify state evolution; fixed image positions preserve the connection to the pretrained task readout.

3.1 Recurrent Updates of Visual States

We write the multimodal embedding sequence as $E(c, I_0) \in \mathbb{R}^{N\times d}$, where N is the sequence length and d is the hidden width. The context c includes the text prompt and any additional images. Following recurrent-depth models (Dehghani et al., 2019; Geiping et al., 2025), we partition the decoder into an input stack P_ϕ (the prelude), a shared block C_θ, and an output stack D_ψ (the coda). The prelude produces $e = P_\phi(E(c, I_0))$, which initializes the full hidden sequence as $h_0 = e$. An index set $\mathcal{V}$ selects the visual state $v_k = [h_k]_\mathcal{V}$: map-image positions for planning, or probe-image positions for rotation. The rotation reference remains in the context, so the model can compare the evolving probe with an observed object.

At iteration k, a learned projection combines the current and initial states at the selected positions:

$$x_k^{(i)} = \begin{cases} W_{\text{in}}[h_{k-1}^{(i)}; e^{(i)}], & i \in \mathcal{V}, \\ h_{k-1}^{(i)}, & i \notin \mathcal{V}, \end{cases} \qquad u_k = C_\theta(x_k). \tag{1}$$

Here $W_{\text{in}} \in \mathbb{R}^{d\times 2d}$ is initialized to $[I\ 0]$. The initial encoding e thus remains available at every iteration, allowing the update to use both the original scene and the current state. The block reads the full sequence, but its output is written back selectively. Context positions take their first-pass outputs u_1 and remain fixed thereafter; visual positions follow the residual update

$$h_k^{(i)} = h_{k-1}^{(i)} + g_k^{(i)}\big(u_k^{(i)} - h_{k-1}^{(i)}\big), \qquad i \in \mathcal{V}. \tag{2}$$

The scalar $g_k^{(i)} \in [0, 1]$ controls how much of each proposed visual update is retained. Values near zero preserve the previous state; values near one accept the proposed state. For planning, a shared linear head predicts it from the concatenated RMS-normalized states $h_{k-1}^{(i)}$ and $u_k^{(i)}$, followed by a sigmoid. Rotation uses $g_k^{(i)} = 1$, directly replacing the selected states at each iteration. The gate permits state preservation; Appendix A.1 gives its initialization.

Within C_θ, positions belonging to the same image attend bidirectionally, while all other connections retain causal ordering. This permits information exchange across the image during an update, with access to preceding context. Text following an image cannot directly affect its queries in the recurrent block. The same block parameters, image positions, and positional encodings are reused across iterations. The image encoder and prelude run once; subsequent iterations update all selected positions together. Increasing the rollout length therefore adds applications of the shared block while keeping the state size fixed.

3.2 Supervision from Observation Sequences

A training example consists of $(c, I_0, I_1, \ldots, I_m, y)$, where y is the task answer when available. The images describe successive states of the desired procedure: observed orientations for rotation, or rendered solver expansions for planning. Only the initial image and context enter the student rollout. Each subsequent image is encoded independently in the same image slot and prompt context, using the frozen backbone with adapters disabled:

$$\tau_j = \text{stopgrad}\big([\bar{C}(\bar{P}(E(c, I_j)))]_\mathcal{V}\big)\,. \tag{3}$$

The teacher applies the reference block $\bar{C}$ once and stops at its output. Student and teacher states therefore have the same hidden width, token layout, and output-layer boundary. The frozen encoder defines position-wise targets independently of the preceding student states. Intermediate images supply training targets only; no image decoder is used in the student rollout. Planning targets use privileged solver information (Sec. 4.2).

We train the state at iteration k to match target $\tau_{j(k)}$, ordinarily with $j(k) = k$. Before comparison, both features are RMS-normalized and centered by a running per-position mean of normalized teacher features. Writing the centered features as $s_k^{(i)}$ and $t_{j(k)}^{(i)}$, we minimize the weighted cosine loss

$$\mathcal{L}_{\text{feat}} = \frac{\sum_{k,i} w_{k,i}\big[1 - \cos(s_k^{(i)}, t_{j(k)}^{(i)})\big]}{\sum_{k,i} w_{k,i}}, \qquad i \in \mathcal{V}. \tag{4}$$

Spatial weights emphasize cells changed by a planning expansion or the foreground swept by the rotating object. The correspondence between iteration and target specifies the intended progression. We vary this assignment and the available observations to separate the effects of state supervision and intermediate targets. We combine feature supervision with answer-token cross-entropy and task-specific auxiliary terms, $\mathcal{L} = \mathcal{L}_{\text{ans}} + \lambda\mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{aux}}$. Gradients propagate through the complete student rollout, while the teacher targets remain detached. The visual-only control removes answer and auxiliary losses. Since the feature objective is applied before the coda, this control tests whether the supervised states support the existing comparison readout without task-output supervision.

3.3 State Preservation and Task Prediction

Planning requires the update to continue a partial computation and retain its result once complete. We expose the model to both cases by sampling an intermediate starting image I_s with probability 0.5 and appending $q \in \{0, \ldots, 4\}$ terminal holding steps. The rollout then has length $R = m - s + q$, with target assignment $j(k) = \min(s + k, m)$. Repeated final targets supervise preservation, and a penalty of 0.01 times the mean gate opening encourages small updates when the state no longer needs to change. The image retains the original player and goal, and the answer remains the full path from the original start. Rotation uses the original observation trajectories and an auxiliary halting loss, without the gate or these trajectory augmentations. Appendix A.2 defines the auxiliary objectives.

After R updates, the coda processes h_R with ordinary causal attention, and the pretrained output head produces the answer. Thus, perceptually supervised states feed directly into the model's task readout; a separate image decoder is used only for analysis. The inference horizon is chosen by the evaluation protocol: planning uses a fixed budget for each environment call, whereas the main rotation evaluation prescribes the horizon from the known angle. State preservation and rollout termination are separate decisions.

Parameter sharing permits execution beyond the supervised horizon, but a stable state can also be an incorrect fixed point. The stopping diagnostics therefore evaluate whether state preservation reliably indicates task completion.

Our implementation adapts Qwen3-VL-4B-Instruct (Bai et al., 2025). Layers 0–11 form the prelude, 12–15 the shared block, and 16–35 the coda. The pretrained weights remain frozen; rank-16 LoRA adapters in all 36 layers contribute approximately 46M trainable parameters (Hu et al., 2022), alongside the input projection and auxiliary heads. The recurrent adapters are shared across iterations, and both tasks use full backpropagation through the update sequence. The student executes $32 + 4R$ decoder-layer applications before answer generation. Teacher encoding adds training computation only; Appendix A.3 details the accounting.

4 Experiments

We ask whether perceptual supervision improves task prediction, how intermediate targets contribute beyond final-state targets, and where the learned updates extrapolate. VSP and mental rotation provide visual and spatial reasoning settings, with a separate model trained for each task. Maze-Hard tests an extension with a different supervision and readout interface.

4.1 Experimental Setup

VSP and mental rotation adapt Qwen3-VL-4B with AdamW, learning rate 10^{-4}, and effective batch size 16. The principal VSP runs use 3,230 updates (four epochs), 384-pixel images, and gated updates. Rotation uses 6,000 updates (two epochs), 320-pixel images, and ungated updates. Repeated results report mean $\pm$ standard deviation over three seeds unless marked otherwise. Paired differences report 95% intervals from 10,000 item-level bootstrap resamples over a shared item set. The main SFT and recurrent conditions use matched training-step budgets within each task; Maze-Hard budgets are listed separately. Appendices A, B, and C.2 give training details.

We compare five supervision conditions, with a separate Mirage baseline for VSP. *Supervised fine-tuning (SFT)* uses task answers without recurrence. *Answer-only recurrence* retains the visual update but omits intermediate-image supervision ($\lambda = 0$). *Final-state supervision* uses the final observation as the perceptual target, omitting the intermediate observations. *Ordered supervision* adds perceptual targets in sequence, matching each iteration to the corresponding observation. *Shuffled supervision* permutes the assignment of target images to iterations, keeping the student input and image-token order unchanged. All LVT conditions retain the auxiliary losses in Sec. 3.3; "answer-only" removes visual targets.

4.2 Visual Reasoning: Visual Spatial Planning

Task and targets. VSP asks a model to navigate a rendered grid to a goal while avoiding holes (Wu et al., 2024; Yang et al., 2026). We train on 12,000 generated maps and 921 eligible official training maps, with grid sizes 3×3–6×6 and shortest paths of length 1–6. A custom extrapolation set contains 300 unseen 6×6 maps with paths of length 7–12, disjoint from training and validation. Because configurations were compared on this split, 300 fresh maps (50 per length) were generated and fixed before their evaluation; we also report the 400-map official test. Supervision renders breadth-first search expanding from the goal: each newly reached safe cell receives an arrow toward a closer neighbor. Expansion stops at the player, and ties are broken as in the answer path.

Evaluation protocol. Open-loop evaluation decodes and executes a complete plan once. Closed-loop evaluation is chunked: each call predicts a full plan, executes its first two actions, re-renders the map at the new position, and starts a new call with reset context and latent state. Looped models use 20 updates per call, independent of the true path length. An episode ends after a predicted plan of at most two actions, entry into a hole, or 16 calls (20 on 9×9 and 10×10 maps). Success requires ending at the goal without entering a hole. All trained baselines share the protocol within each column; re-rendered observations are part of inference, unlike official open-loop evaluation.

State supervision and intermediate targets. Table 1 separates closed-loop and open-loop execution. On fresh maps, final-state supervision raises closed-loop success from $64.3 \pm 1.8\%$ for answer-only recurrence to $93.2 \pm 1.3\%$; ordered observations reach $97.0 \pm 0.3\%$. The paired gain from ordered over final-state supervision is $+3.8$ points [$+2.0$, $+5.7$], compared with $+32.7$ [$+28.0$, $+37.6$] over answer-only recurrence. The custom split shows the same pattern. State supervision thus accounts for most of the gain in this setting, with a further benefit from intermediate observations. Shuffled targets give $69.7 \pm 2.1\%$ on fresh maps. Because shuffling perturbs the consistency of the training targets, the final-state comparison more directly measures the additional benefit of intermediate observations.

Latent visual token baseline. We reimplement Mirage with the SFT backbone, adapter, and 3,228 updates (SFT: 3,230), aligning four latent positions with helper-image features before answer-only training. Route helpers give a fresh-map difference of $+0.2$ points [-2.3, $+2.9$] from SFT and -39.3 [-44.4, -34.4] from LVT. On larger grids, they stay 1.1–5.5 points below SFT (Table 2). LVT's final arrow-field helper gives 51.0% in one run. Training steps are matched; latent capacity, target sequences, and computation differ (Appendix B.7).

Execution feedback and recurrent depth. The gain concerns planning with repeated observations and short action prefixes: open-loop LVT reaches $18.1 \pm 6.4\%$ on the custom split, versus $18.7 \pm 9.3\%$ for answer-only recurrence. On fresh maps, closed-loop execution adds $+77.6$ points [$+74.0$, $+81.0$] for LVT, versus $+42.3$ [$+37.6$, $+47.1$] for SFT. Within that protocol, increasing the internal depth

Table 1: **VSP success (%).** Custom and fresh splits each contain 300 6×6 maps with 7–12-step paths; training paths have length 1–6. LVT rows share the gated update. Mean $\pm$ s.d. over three seeds; † one seed. Text CoT uses different image sampling (Appendix B.2).

(a) Closed-Loop Execution: Two Actions per Call

Configuration	Custom Maps	Fresh Maps
SFT	61.4 ± 1.3	57.4 ± 0.7
Mirage (route helper)	58.7 ± 1.5	57.7 ± 0.9
Mirage (arrow-field helper)†	55.7	51.0
LVT (answer supervision only)	67.1 ± 1.1	64.3 ± 1.8
LVT (final-state supervision)	92.0 ± 0.3	93.2 ± 1.3
LVT (ordered observations)	$\mathbf{96.8 \pm 1.7}$	$\mathbf{97.0 \pm 0.3}$
LVT (shuffled observations)	67.4 ± 2.7	69.7 ± 2.1
Text CoT	99.8 ± 0.2	99.1 ± 0.5

(b) Open-Loop Execution: One Complete Plan

Configuration	Custom Maps	Official Test (400 maps)
SFT	13.9 ± 0.8	93.9 ± 0.6
LVT (ordered observations)	18.1 ± 6.4	96.2 ± 0.1
Text CoT	98.3 ± 0.9	$99.75^{\dagger}$

Table 2: **Planning on larger grids (%).** Three-seed mean success under the same closed-loop protocol, at 64 pixels per cell. Training maps are at most 6×6 with paths of length 1–6. Mirage uses the route helper. Training-image sampling differs between methods. Multi-resolution text controls appear in Appendix B.2.

Grid	Path Length	SFT	Mirage	Text CoT	LVT
7×7	7–12	75.3	73.1	39.3	**98.4**
8×8	7–12	62.4	61.3	16.4	**96.7**
8×8	13–16	18.3	16.7	5.0	**80.6**
9×9	7–12	64.4	58.9	14.4	**88.3**
10×10	7–12	47.2	45.6	10.6	**81.1**

from 1 to 2, 4, 8, and 20 updates raises LVT success from 19.4 to 29.8, 44.3, 82.0, and 97.0%. Both recurrent depth and environment feedback contribute to performance.

Textual supervision and grid-size extrapolation. A textual chain-of-thought (CoT) control describes the map, the same search process, and the resulting path. It is stronger on 6×6 maps, including the fresh length-extrapolation split, where the paired LVT–CoT difference is -2.1 points $[-3.7, -0.7]$. On larger grids under the same closed-loop protocol, LVT retains 80.6–98.4% success, compared with 5.0–39.3% for text CoT (Table 2). Error inspection reveals systematic map-transcription errors: all 300 examined 7×7 open-loop outputs describe a 6×6 grid. A multi-resolution text control exhibits the same error on 146 of 150 examined maps (Appendix B.2). This identifies a failure of the learned textual format in these controls; their image sampling and output lengths differ from LVT, so the comparison does not isolate representation choice alone.

A 20-update LVT call uses 112 decoder-layer applications before answer generation, versus 36 for SFT. Teacher encoding adds training computation. Total inference cost depends on both recurrent depth and the number of environment calls, in addition to answer generation. Appendices B.3 and B.4 report grid-size, resolution, and training-map sensitivity.

4.3 Spatial Reasoning: Mental Rotation

Task and protocol. The model judges whether a reference object and a rotated probe depict the same object or a mirror pair (Shepard & Metzler, 1971). Subsequent views supervise probe-token updates; only the initial images and text enter the student. Training, validation, and testing use 2,400,

Table 3: **Mental rotation accuracy (%).** Training uses at most four 30° updates; testing requires five or six, prescribed from the known angle even in panel (a). Each axis is trained separately. Recurrent updates are ungated. Mean ± s.d. over three seeds; entries without s.d. are single runs.

(a) Without Angle in Prompt: Vertical Axis

Configuration	150°	180°
SFT	84.2 ± 1.3	64.3 ± 1.8
LVT (ordered observations)	$\mathbf{96.9 \pm 0.6}$	$\mathbf{86.4 \pm 1.8}$

(b) With Angle in Prompt: Unseen-Angle Average

Configuration	Vertical Axis	Image Plane
SFT	44.8	68.2
LVT (answer supervision only)	79.1 ± 1.1	94.8 ± 4.6
LVT (shuffled observations)	75.5	60.2 ± 2.5
LVT (ordered observations)	$\mathbf{92.1 \pm 0.3}$	97.3 ± 1.6

Table 4: **Maze-Hard exact match (%).** LVT and SFT share the backbone and grid readout; 1,000 official test mazes, one seed. Recurrent rows supervise the complete solution at every iteration. Prescribed horizons use solver trajectories; 32 iterations is a fixed budget.

Configuration	Training Updates	Prescribed Horizon	32 Iterations
SFT (grid readout)	1,000	0.0 (single pass)	
LVT (terminal-solution supervision)	1,000	55.5	55.0
LVT (partial-path inputs)	1,000	59.4	58.3
LVT (partial-path inputs)	2,000	**67.1**	**66.9**

300, and 300 disjoint objects. Training contains at most four 30° updates; the principal test has 1,200 trials requiring five or six (150° or 180°). Evaluation prescribes the horizon from the known angle. Vertical-axis rotation is primary; models for other axes are trained separately (Appendix C).

Prompt control and unseen angles. Table 3a compares models trained and evaluated without angle text. SFT is substantially stronger under this prompt condition, and LVT still improves accuracy at both unseen angles. The evaluator still sets LVT's loop count from the angle, testing execution at a prescribed horizon. With angle text, ordered supervision reaches $92.1 \pm 0.3\%$ averaged over unseen angles, versus $79.1 \pm 1.1\%$ for answer-only recurrence (panel b). A single recurrent update gives 66.2%.

Visual supervision without answer labels. With answer and halting losses removed, visual supervision alone reaches 87.2% on unseen angles. The unchanged pretrained comparison readout can therefore use states trained solely through the perceptual objective. Repeated final-view supervision gives $68.8 \pm 5.5\%$, and shuffled targets give 75.5%, showing a larger benefit from intermediate observations here than in VSP (Appendix D). The gain varies across transformations: image-plane rotation improves less over answer-only recurrence.

4.4 Extension to Maze-Hard

Maze-Hard provides 1,000 training and 1,000 test mazes on 30×30 grids, with canonical shortest paths of length 109–153 (Wang et al., 2025). We evaluate exact match to the entire solution grid. This experiment retains recurrent updates of visual tokens but replaces perceptual targets with a cell-classification loss: a shared readout head predicts the complete solution at every iteration. The single-pass SFT control uses the same backbone and grid head, trained on the same solutions. All configurations use 960-pixel images and rank-32 LoRA (details in Appendix G.2).

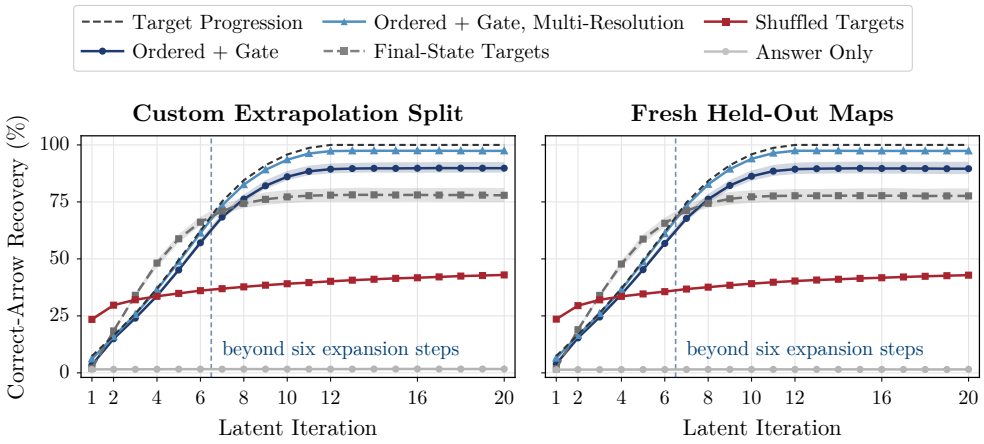


Figure 2: **Perceptual supervision shapes recurrent planning states.** Correct-arrow recovery on 300 maps per split. Lines show three-seed means; bands span the minimum and maximum. The decoder is trained only on frozen perceptual states. The vertical line marks the six-step training expansion limit. Multi-resolution training is a separate configuration (Appendix B.3).

With 1,000 training updates, terminal-solution supervision reaches 55.0% exact match at a fixed budget of 32 iterations, compared with 0.0% for single-pass SFT (Table 4). Training on partially drawn paths raises this to 58.3%; extending that training to 2,000 updates gives 66.9%. These single runs support recurrent refinement with a grid readout; they use terminal-solution classification rather than perceptual observation targets. Appendix G.2 gives feedback and stopping variants.

5 Analysis of Recurrent Visual Computation

5.1 Visual Structure Learned from Supervision

Planning states. An image decoder trained only on frozen perceptual states and fixed across conditions reveals progressive arrow-field recovery under ordered supervision on a ten-step map (Fig. 3a). On 300 fresh maps (three-seed means), ordered supervision yields 89.6% correct-arrow recovery at iteration 20, compared with 77.6% for final-state supervision, 42.9% for shuffled targets, and 1.5% for answer-only recurrence (Fig. 2). Ordered states recover 84.0% of the target rings beyond the six-step training limit. Thus, state supervision improves both planning accuracy and recoverable field structure, while intermediate observations further shape the progression. The additional benefit also depends on the evaluation setting: ordered supervision exceeds final-state supervision by $+21.7$ points $[+13.3, +30.0]$ on 13–16-step 8×8 paths, versus $+3.8$ on fresh 6×6 maps.

Rotation states. A representation autoencoder (RAE) provides the corresponding diagnostic for rotation (Fig. 3b). Its decoder is trained on frozen perceptual states of validation objects, never recurrent states, and reaches 0.97 foreground IoU on held-out perceptual reconstructions. Ordered supervision preserves recognizable structure at successive orientations for both unseen objects, including at untrained iterations 5–6; answer-only reconstructions progressively lose structure. Repeated final-view targets instead produce the aligned view early (Fig. 8b). These differences show how the supervision affects recoverable visual content. Trajectory fidelity nevertheless declines: nearest-view retrieval falls from 64% at iteration 2 to 22% at iteration 6 (8.3% chance). Moreover, reducing the feature-loss weight to 0.3 retains 92.7% task accuracy while retrieval is at most 16% (Appendix D). High task accuracy can therefore coexist with weak alignment to the prescribed intermediate views.

5.2 State Use and Response to Visual Input

Decoding establishes recoverable content; interventions test whether the model uses the recurrent state. In VSP, replacing it with a state computed for a moved goal makes the first action follow that goal on 96–100% of eligible maps, while the original input and context remain fixed. This holds for

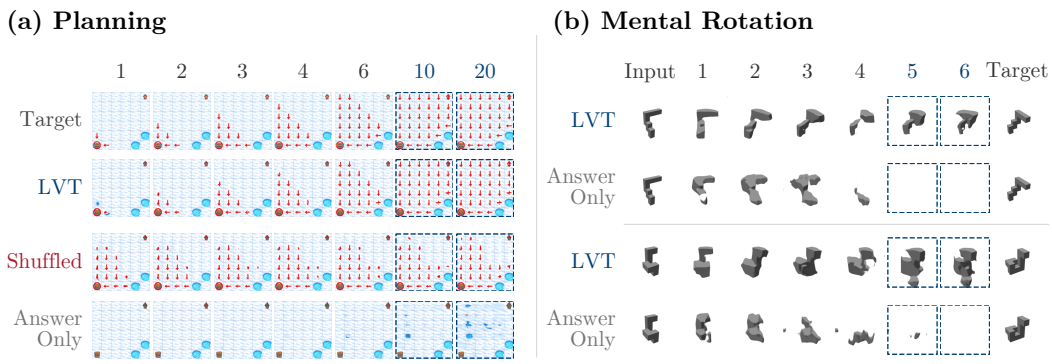


Figure 3: **Decoded recurrent states. (a)** A ten-step planning map: target fields and decoded states for LVT, shuffled, and answer-only supervision. **(b)** Two unseen rotation objects: input, iterations 1–6, and aligned target for LVT and answer-only recurrence. Decoders are trained only on frozen perceptual states. Dashed boxes mark iterations beyond the six-step planning expansion horizon or four-step rotation training horizon. Full sequences and teacher controls: Appendices B.5, E.2, and I.

ordered, final-state, shuffled, and answer-only supervision (Appendix B.6). Answer-only states thus influence prediction despite their low arrow recovery, separating state use from perceptual readability.

To test the influence of persistent observations, we insert a wall on the first optimal step only into the re-injected input. Ordered models reroute on 23–32% of test maps and 52–64% of fresh maps; multi-resolution ordered models reroute on 73–86%. Other recurrent conditions respond on 0–18%, near their control-wall noise. The stronger response under ordered supervision is consistent with greater use of re-injected visual evidence during later updates. State-only walls also reroute predictions, with rates of 48–100% across conditions.

For Mirage, planted-goal and state-only-wall latents leave the plan text unchanged on 93–100% of maps; zero and noise replacements also have little effect (Appendix B.7). These controls reveal low sensitivity to the appended latents in this reimplementation. A probe trained on all four latents recovers 58–60% of fresh-map arrow cells, versus 55–57% from the first latent alone; unlike LVT's decoder, this probe is fitted directly on the evaluated representation type.

For rotation, replacing the final probe state with its first-iteration state reduces accuracy from 91.92% to 23.67%; zeroing it gives 51.50% on 1,200 trials. The readout is sensitive to the evolved state, although these replacements also shift its input distribution and do not identify which intermediate orientation it uses.

5.3 State Preservation and Execution Length

Planning requires retaining useful states after the expansion is complete. In the seed-0 VSP ablation, removing the residual gate reduces closed-loop success from 98.7% to 78.7%; removing random-phase starts gives 91.0%, and causal within-image attention gives 69.7% (Table 15, Appendix D). The principal gated model maintains decoded field coverage through 20 updates (Fig. 2).

Preservation and stopping pose separate problems. VSP convergence-based stopping reaches 97.7% in one run. Rotation trained with at most three updates gives 90, 86, and 80% at four, five, and six prescribed updates (Appendix E.4).

6 Conclusion

LVT learns iterative visual computation by aligning recurrent image-token states with perceptual targets: state supervision improves planning and rotation, intermediate observations aid extrapolation, and gated updates preserve planning states. Decoding and interventions connect recoverable visual structure to prediction, while the Mirage reimplementation shows little sensitivity to the tested latent edits under a matched training-step budget.

AI USE STATEMENT

AI assistance was limited to manuscript polishing, including revisions to wording, organization, and presentation. The research design, experimental implementation, data generation, experiment execution, and scientific analyses were conducted by the authors. The authors are responsible for the scientific content, citations, and final text.

ETHICS STATEMENT

The main experiments study procedurally generated maps and rendered synthetic objects; they do not involve human participants or personal data. The work examines a model's internal computation rather than deployment in a physical environment. Broader applications would require evaluation under their own observation, feedback, and error conditions.

REPRODUCIBILITY STATEMENT

Section 3 defines the state update and perceptual objective. Appendix A gives the complete write-back rule, attention mask, target normalization, auxiliary losses, and computation accounting. Appendix B specifies planning targets, data splits, feedback and scoring protocols, and text-process controls. Appendices C and C.2 provide rotation data generation and training settings. Section 4.4 and Appendix G.2 specify Maze-Hard readouts, training budgets, and inference protocols. Seed counts and exceptions are identified in the result tables. Appendix E describes the diagnostic decoders, retrieval controls, and interventions used to assess the learned states.

REFERENCES

Ashutosh Adhikari and Mirella Lapata. Multimodal latent reasoning via predictive embeddings. *arXiv preprint arXiv:2604.08065*, 2026. URL https://arxiv.org/abs/2604.08065.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, June 2023. doi: 10.1109/cvpr52729.2023.01499. URL https://openaccess.thecvf.com/content/CVPR2023/html/Assran_Self-Supervised_Learning_From_Images_With_a_Joint-Embedding_Predictive_Architecture_CVPR_2023_paper.html.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder, 2021. URL https://arxiv.org/abs/2107.05407.

Arpit Bansal, Avi Schwarzschild, Eitan Borgnia, Zeyad Emam, Furong Huang, Micah Goldblum, and Tom Goldstein. End-to-end algorithm synthesis with recurrent networks: Logical extrapolation without overthinking. In *Advances in Neural Information Processing Systems*, 2022.

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

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyzdRiR9Y7.

Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4829–4837, June 2016. doi: 10.1109/cvpr.2016.522. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Dosovitskiy_Inverting_Visual_Representations_CVPR_2016_paper.html.

Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.15647.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems*, volume 38, pp. 41340–41391, 2025. doi: 10.52202/085713-1380. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/3b01972cf31e6fa0fe29e4b8b5c2a0a1-Paper-Conference.pdf.

Alex Graves. Adaptive computation time for recurrent neural networks, 2016. URL https://arxiv.org/abs/1603.08983.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025. URL https://arxiv.org/abs/2510.04871.

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 148076–148090, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/efa0939caa8a906fd0a61a9c60e46e8b-Paper-Conference.pdf.

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 *International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=l1cMErXg1P.

Hongyuan Adam Lu, Z. L. Victor Wei, Qun Zhang, Jinrui Zeng, Bowen Cao, Lingwei Meng, Mocheng Li, Zezhong Wang, Haonan Yin, Naifu Xue, Minyu Chen, Cenyuan Zhang, Zefan Zhang, Hao Wei, Jiawei Zhou, Haoran Xu, Hao Yang, Ronglai Zuo, Tongda Xu, Yonghao Li, Jian Chen, Hebin Wang, Zeyu Gao, Yang Li, Wei Zhao, Qimin Zhong, Siqi Liu, Yumeng Zhang, Leyan Cui, Zhangyu Wang, and Wai Lam. Looped world models. *arXiv preprint arXiv:2606.18208*, 2026. URL https://arxiv.org/abs/2606.18208.

Suhyeong Park, Junha Jung, and Jaewoo Kang. Reason through the latent! making latent visual reasoning necessary. *arXiv preprint arXiv:2609.06746*, 2026. URL https://arxiv.org/abs/2609.06746.

Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and XuDong Wang. Chain-of-visual-thought: Teaching VLMs to see and think better with continuous visual tokens. In *European Conference on Computer Vision (ECCV)*, 2026. URL https://arxiv.org/abs/2511.19418.

Thomas Ricouard. Architectural visualization with Astra. OpenAI Developers, September 2026. URL https://developers.openai.com/blog/architectural-visualization-with-astra. Published September 4, 2026; accessed September 12, 2026.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations*, volume 2025, pp. 14855–14881, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/2676109d49d1eb26d6bc584a8f556305-Paper-Conference.pdf.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. In *Advances in Neural Information Processing Systems*, 2021.

Roger N. Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, February 1971. ISSN 1095-9203. doi: 10.1126/science.171.3972.701. URL https://doi.org/10.1126/science.171.3972.701.

Wen-Jie Shu, Xuerui Qiu, Rui-Jie Zhu, Harold Haodong Chen, Yexin Liu, and Harry Yang. LoopViT: Scaling visual ARC with looped transformers. *arXiv preprint arXiv:2602.02156*, 2026. URL https://arxiv.org/abs/2602.02156.

Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems*, 2016.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025. URL https://arxiv.org/abs/2506.21734.

Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond image and language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12030–12040, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Wang_Monet_Reasoning_in_Latent_Visual_Space_Beyond_Image_and_Language_CVPR_2026_paper.html.

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, 2024. URL https://arxiv.org/abs/2407.01863.

Zeyuan Yang, Xueyang Yu, Delin Chen, Maohao Shen, and Chuang Gan. Machine mental imagery: Empower multimodal reasoning with latent visual tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33510–33520, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Yang_Machine_Mental_Imagery_Empower_Multimodal_Reasoning_with_Latent_Visual_Tokens_CVPR_2026_paper.html.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *International Conference on Learning Representations*, volume 2026, pp. 35791–35820, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/3c4141c12660ad3625eb4ae845e0a6f9-Paper-Conference.pdf.

APPENDIX CONTENTS

A **Implementation Details** . . . 14
A.1 State update and attention . . . 14
A.2 Perceptual targets and training objectives . . . 14
A.3 Inference and computation . . . 15
B **Additional VSP Results and Evaluation Protocols** . . . 15
B.1 Training and evaluation configurations . . . 15
B.2 Textual process supervision . . . 16
B.3 Sensitivity to grid size and image resolution . . . 16
B.4 Sensitivity to training-map quantity . . . 17
B.5 Decoded planning states . . . 18
B.6 Planning-state interventions . . . 19
B.7 Latent visual token baseline . . . 19
C **Mental Rotation: Data and Training** . . . 21
C.1 Comparison protocol and observation sequences . . . 21
C.2 Training details and hyperparameters . . . 22
C.3 Prompt for spatial comparison . . . 23
D **Additional Supervision and Update Ablations** . . . 23
E **Analysis of Rotation States** . . . 25
E.1 Diagnostic decoders . . . 25
E.2 Alignment and reconstruction . . . 25
E.3 Context access and reference copying . . . 27
E.4 Interventions and additional diagnostics . . . 27
F **Additional Rotation Settings** . . . 28
F.1 Rotation in the image plane . . . 28
G **Additional Tasks** . . . 29
G.1 Synthetic grid updates with convergence stopping . . . 29
G.2 Maze-Hard and explicit state readouts . . . 30
H **Contextual Comparison with API Reasoning Models** . . . 30
I **Additional Reconstructions** . . . 32

A Implementation Details

A.1 State update and attention

The model partitions the 36 decoder layers of Qwen3-VL-4B-Instruct into a 12-layer prelude, a four-layer shared block, and a 20-layer coda. All pretrained weights, including the vision encoder, remain frozen. Rank-16 LoRA adapters with scaling factor $\alpha = 32$ are applied to the attention and feed-forward projections. The hidden width is 2,560. The input-injection projection has shape 2560×5120 and is initialized to $[I\ 0]$. The selected state has 144 tokens at 384×384 pixels for VSP and 100 tokens at 320×320 for rotation.

For completeness, the write-back operation in Eq. 2 extends to the full sequence as

$$h_k^{(i)} = \begin{cases} h_{k-1}^{(i)} + g_k^{(i)}(u_k^{(i)} - h_{k-1}^{(i)}), & i \in \mathcal{V}, \\ u_1^{(i)}, & i \notin \mathcal{V},\ k = 1, \\ h_{k-1}^{(i)}, & i \notin \mathcal{V},\ k > 1. \end{cases} \tag{5}$$

The VSP gate uses $\mathrm{rms}(a) = a/\sqrt{\|a\|_2^2/d + \epsilon}$ and

$$g_k^{(i)} = \sigma\Big(w_g^\top [\mathrm{rms}(h_{k-1}^{(i)}); \mathrm{rms}(u_k^{(i)})] + b_g\Big). \tag{6}$$

Its parameters are shared across positions and iterations; the scalar output is broadcast over hidden channels. Initialization with $w_g = 0$ and $b_g = 4$ makes the initial update nearly fully open. The gate controls state preservation, independently of the rule that ends a rollout. Rotation uses the same state selection with $g_k^{(i)} = 1$.

The recurrent block reads the entire multimodal sequence even when only selected positions are written back. Attention is bidirectional within each image block and causal elsewhere; padding keys are masked. Consequently, preceding images and text may affect the visual update, while text following an image is not directly visible to that image's queries. The coda uses the original causal mask. The reference-blocking rotation control removes reference-image keys for probe queries inside the recurrent block; it does not change the prelude.

A.2 Perceptual targets and training objectives

The teacher encodes each intermediate image in the same image slot and prompt context as the student. Adapters are disabled, and the forward pass stops after one application of the reference recurrent block. Target states are detached. The student never receives the target image as an extra observation during its latent rollout.

Before evaluating Eq. 4, both features are RMS-normalized and centered using an exponential moving average of the normalized teacher features at each image position, with momentum 0.99. For multi-resolution VSP, separate means are maintained for each token-grid size. Rotation assigns weight 1 to the union of the object's foreground across the initial probe and all target views, and 0.1 elsewhere. This mask is shared across iterations, so shuffling the target views leaves the spatial weights unchanged. A token is foreground if any pixel in its 32×32 cell is darker than 245. VSP emphasizes cells changed by the current expansion step.

The combined objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{ans}} + \lambda \mathcal{L}_{\mathrm{feat}} + \lambda_g \mathcal{L}_{\mathrm{gate}} + \lambda_h \mathcal{L}_{\mathrm{halt}}, \qquad \mathcal{L}_{\mathrm{gate}} = \frac{1}{R|\mathcal{V}|} \sum_{k=1}^{R} \sum_{i \in \mathcal{V}} g_k^{(i)}. \tag{7}$$

Sums exclude padded or inactive iterations. VSP uses $\lambda_g = 0.01$ and no halting term. The main rotation configuration uses feature weight 1 and halting weight 0.1, without the gate cost. Its halting head is linear on the mean of RMS-normalized probe states and uses binary cross-entropy with target $\mathbf{1}[k = R]$. The visual-only control removes answer and halting losses. Alternative feature-weight schedules are listed in Appendix D.

For VSP, random-phase initialization selects an intermediate frame I_s with probability 0.5; otherwise $s = 0$. Terminal holding appends $q \in \{0, \ldots, 4\}$ final-state targets:

$$e = P_\phi(E(c, I_s)), \qquad R = m - s + q, \qquad j(k) = \min(s + k, m). \tag{8}$$

The original player and goal remain marked, and the answer remains the complete path from the original start. These augmentations supervise both continuation and preservation. They are absent from the reported rotation models.

A.3 Inference and computation

Each closed-loop VSP call resets the recurrent state and context. The external evaluator executes an action prefix and renders the actual environment state; it supplies no solver state or shortest-path information. The next call therefore observes execution feedback, rather than a carried-over latent block. Open-loop evaluation and rotation have no such feedback.

With partition depths L_P, L_C, L_D, the student performs

$$N_{\text{student}}(R) = L_P + RL_C + L_D = 32 + 4R \tag{9}$$

decoder-layer applications before autoregressive answer generation. Encoding m distinct target images adds $m(L_P + L_C) = 16m$ applications during training. Teacher computation is absent at inference. Layer counts exclude vision encoding and do not measure latency or FLOPs.

B Additional VSP Results and Evaluation Protocols

B.1 Training and evaluation configurations

Table 5 distinguishes the two principal implementations. The shared formulation in Sec. 3 does not imply that every task uses the gate, holding, or stopping objective.

Table 5: **Task-specific settings.** Both tasks use a frozen Qwen3-VL-4B backbone, rank-16 LoRA in all decoder layers, input injection, and the same four-layer recurrent partition.

Setting	VSP Main Comparison	Mental Rotation Main Comparison
Visual State	Map Image, 144 Tokens at 384 px	Probe Image, 100 Tokens at 320 px
Process Target	Goal-Outward BFS Arrow Field	Successive 30° Views
Residual Gate	Learned Per-Token Scalar	Fixed to One
Terminal Holding	0–4 Extra Iterations	Absent
Random-Phase Start	Probability 0.5	Absent
Auxiliary Objective	Mean Gate Opening, Weight 0.01	Halting BCE, Weight 0.1
Training Budget	4 Epochs, 3,230 Updates	2 Epochs, 6,000 Updates
Evaluation Horizon	Fixed 20 per Call	Prescribed From Angle
Outer Feedback	Optional Action Execution and Rendering	None

VSP targets are drawn on the official map tiles. Each reachable safe cell receives one of four arrow symbols, and the goal receives a circle. Targets stop expanding when the wavefront reaches the player; the separate full-flood ablation continues until all reachable cells are covered. The changed-cell mask emphasizes newly updated regions at each transition. The shuffled arm uses the same set of target frames. For multi-resolution training, a batch samples one size from $\{256, 288, 320, 352, 384, 448, 512\}$; input and target images are resized together, and teacher running means are maintained separately for each token-grid size. The eight-epoch multi-resolution experiment is separate from the four-epoch, fixed-resolution main comparison.

The custom map generator excludes overlaps with training and validation maps. The main long-path split contains 300 6×6 maps. Size extrapolation uses 150 maps per 1–6- or 7–12-step condition at 7×7 and 8×8, and 60 maps for the 8×8, 13–16-step condition. Additional 9×9 and 10×10 evaluations use 60 maps per length range. The official 400-map test is reported independently; the generated splits are not official benchmark test sets.

The closed-loop evaluator accepts at most two moves per call and caps an episode at 16 calls (20 on 9×9 and 10×10 maps). It uses greedy decoding with up to 64 new tokens for action-only models. Out-of-bounds moves leave the player in place, a hole terminates the episode, and a short decoded plan ends execution. Success is checked by simulating all accepted moves and testing whether the final position is the goal without entering a hole. The input renderer redraws the actual player position without revealing the solver trajectory. The main deterministic protocol is distinct from an environment-termination variant used in stochastic-execution experiments. For completeness, the answer-only and shuffled recurrent controls reach $18.7 \pm 9.3\%$ and $15.9 \pm 6.7\%$ on the custom split in open loop, and $95.4 \pm 1.4\%$ and $96.0 \pm 1.3\%$ on the official test, respectively.

B.2 Textual process supervision

The text CoT control serializes the map with safe/hole/player/goal symbols, describes breadth-first expansion from the goal, traces a path, and emits a boxed answer. It uses the same backbone, training-map pool, and underlying solver process as the visual-target experiment. The final version (cot3) uses cumulative-plan text and oversamples officially rendered training images sixfold, raising their sampling share from approximately 7% to 31%. Training-example distributions and computation therefore differ. The final control generates approximately 600 reasoning tokens per open-loop example and approximately 360 per closed-loop call. LVT generates no reasoning text inside a latent rollout, but still generates action tokens and incurs the recurrent and repeated-call computation. These token counts alone do not measure latency or FLOPs.

On unseen 7×7 maps rendered with 64-pixel cells, cot3 reaches 50.7% on 1–6-step paths and 13.3% on 7–12-step paths under *open-loop* evaluation. All 300 outputs transcribe the maps as 6×6. Under the same closed-loop protocol (three-seed means), cot3 reaches 60.9 and 39.3%, versus 100.0 and 98.4% for LVT and 99.1 and 75.3% for SFT; Sec. 4.2 reports 8×8 and 9×9 maps. On short 8×8 paths, text CoT reaches 36.4%, versus 99.6% for LVT. The transcription error is directly observable; image sampling and generated sequence length still differ between the controls.

A multi-resolution text CoT control reaches 100.0% on the official test yet still describes 146 of 150 examined 7×7 maps as 6×6. Its closed-loop success (three-seed means) is 58.7% on short 7×7 paths, 16.2% on 7–12-step 8×8 paths, 5.6% on 13–16-step 8×8 paths, and 15.6% on 7–12-step 9×9 paths rendered at 576 pixels. This is a learned transcription-format failure in the evaluated setup, rather than evidence of an intrinsic limit on textual reasoning.

B.3 Sensitivity to grid size and image resolution

A separate eight-epoch experiment samples resolutions from 256 to 512 pixels while retaining the same training maps. Table 6 compares fixed-canvas and fixed-cell-size evaluation. LVT generalizes to larger grids on 7–12-step routes. For 13–16-step routes on 8×8 maps, success rises from $77.8 \pm 0.9\%$ at 384 pixels to $93.3 \pm 5.8\%$ at 512 pixels (SFT: 22.2 ± 2.5 and $23.3 \pm 2.9\%$). Generalization therefore depends on both route length and visual resolution. Table 7 varies only the loop targets on the fresh split and larger grids, with paired bootstrap intervals over maps; Table 8 splits 9×9 and 10×10 maps by path length. Appendix B.4 reports the data-quantity study.

Table 6: **Size and resolution extrapolation.** Closed-loop success (%) after multi-resolution training. Three seeds. Test sets contain 300 maps at 6×6 (custom and fresh splits), 150 for larger grids with 7–12 steps, and 60 with 13–16 steps.

Grid	Path Length	Image Size	SFT	LVT
6×6 (custom)	7–12	384	65.0 ± 2.9	98.2 ± 0.5
6×6 (fresh)	7–12	384	63.8 ± 1.3	99.4 ± 1.0
7×7	7–12	384	80.0 ± 1.8	98.9 ± 0.3
7×7	7–12	448	83.3 ± 1.2	99.3 ± 0.0
8×8	7–12	384	70.7 ± 2.9	95.8 ± 1.4
8×8	7–12	512	74.7 ± 2.9	98.2 ± 1.4
8×8	13–16	384	22.2 ± 2.5	77.8 ± 0.9
8×8	13–16	512	23.3 ± 2.9	93.3 ± 5.8

Table 7: **What the loops are trained toward.** VSP success (%) for looped models that differ only in the loop targets. Δ rows: paired difference of ordered targets from a control, with a 95% bootstrap interval over maps. The multi-resolution recipe has no shuffled-target model.

		6×6 fresh	7×7	8×8	8×8, 13–16
384 px	Answer only ($\lambda = 0$)	64.3±1.8	80.9±1.0	75.6±1.4	22.8±3.8
	Shuffled targets	69.7±2.1	83.3±0.7	75.1±0.8	21.1±2.5
	Final target only	93.2±1.3	96.0±0.7	90.9±2.0	58.9±9.2
	Ordered targets	**97.0**±0.3	**98.4**±1.0	**96.7**±1.3	**80.6**±3.5
	Δ vs. final only	+3.8 [2.0, 5.7]	+2.4 [0.7, 4.7]	+5.8 [2.9, 8.9]	+21.7 [13.3, 30.0]
	Δ vs. answer only	+32.7 [28.0, 37.6]	+17.6 [12.7, 22.9]	+21.1 [15.8, 26.9]	+57.8 [46.7, 68.3]
256–512	Answer only ($\lambda = 0$)	71.3±2.5	85.3±2.3	80.9±2.0	28.9±2.5
	Final target only	98.7±0.9	98.7±0.7	97.8±1.0	84.4±1.0
	Ordered targets	**99.4**±1.0	**99.3**±0.0	**98.2**±1.4	**93.3**±5.8

Table 8: **Larger maps by path length.** Closed-loop success (%) on 9×9 and 10×10 maps with 7–12 (a), 13–18 (b), and 19–24 (c) moves at 64 pixels per cell; "all" pools the three tiers. The confirmation set was generated after all other results and shares no map with the original tiers. Mean±s.d. over three seeds; †one seed; –: not evaluated.

	9×9				10×10			
Method	a	b	c	all	a	b	c	all
Original tiers, 60 maps each; trained at 384 px								
SFT	64.4±1.9	20.6±1.0	0.0±0.0	28.3±0.6	47.2±4.2	15.0±1.7	0.0±0.0	20.7±1.3
Text CoT	14.4±4.2	3.3±1.7	0.0±0.0	5.9±2.0	10.6±6.7	1.1±1.9	0.0±0.0	3.9±2.9
Answer only	81.7±4.4	33.3±6.0	0.0±0.0	38.3±2.5	68.3±3.3	35.0±3.3	0.6±1.0	34.6±2.1
Shuffled	76.7±1.7	32.8±3.5	0.0±0.0	36.5±1.2	71.7±3.3	33.3±1.7	1.7±0.0	35.6±1.7
Final only	85.6±1.9	47.8±2.5	4.4±1.0	45.9±1.2	78.3±1.7	40.6±2.5	5.0±1.7	41.3±0.3
LVT	88.3±5.8	47.8±11.7	11.1±2.5	49.1±6.4	81.1±6.7	38.9±7.5	8.9±4.2	43.0±5.5
Original tiers, 60 maps each; trained at 256–512 px								
SFT	75.6±2.5	23.9±1.9	0.0±0.0	33.1±0.3	62.8±8.2	31.7±7.3	0.0±0.0	31.5±3.1
Text CoT	15.6±8.2	2.2±1.9	0.0±0.0	5.9±3.2	4.4±4.8	2.2±3.8	0.0±0.0	2.2±2.9
Answer only	87.8±1.9	36.7†	0.0†	41.1†	76.7†	30.0†	1.7†	36.1†
Final only	93.3±1.7	61.7†	23.3†	58.9†	80.0†	45.0†	10.0†	45.0†
LVT	92.8±4.2	72.2±10.0	35.6±9.8	66.9±7.2	80.0±3.3	40.0±5.0	23.3±14.5	47.8±6.7
Confirmation set, 180 maps each; trained at 384 px								
SFT	56.3±2.2	23.0±2.8	0.0±0.0	26.4±0.9	40.0±3.9	18.7±0.8	0.0±0.0	19.6±1.1
Text CoT	11.7±5.5	–	–	–	–	–	–	–
Answer only	79.3±0.8	36.5±2.0	0.0±0.0	38.6±0.9	73.9±1.1	46.3±3.2	0.4±0.6	40.2±1.6
Shuffled	75.2±1.2	31.9±2.1	0.0±0.0	35.7±1.1	73.3±1.5	40.2±0.6	0.0±0.0	37.8±0.7
Final only	82.4±1.4	48.1±0.3	4.4±1.0	45.0±0.2	82.0±2.1	44.8±2.1	3.3±1.0	43.4±1.6
LVT	83.9±3.1	55.9±7.2	9.6±1.2	49.8±3.5	83.0±4.1	39.3±11.8	5.7±2.7	42.7±5.9
Confirmation set, 180 maps each; trained at 256–512 px								
SFT	73.1±2.6	31.1±4.0	0.2±0.3	34.8±2.3	65.6±1.5	35.6±4.9	0.2±0.3	33.8±1.9
Text CoT	13.1±6.4	–	–	–	–	–	–	–
Answer only	84.4±5.3	43.7±4.2	0.4±0.3	42.8±3.2	82.6±5.8	52.4±5.7	1.7±0.6	45.6±3.7
Final only	91.3±1.7	63.9±3.5	30.2±5.3	61.8±3.4	88.0±2.6	55.7±5.0	20.9±3.3	54.9±3.6
LVT	85.2±3.6	73.3±10.0	33.3±10.6	64.0±8.0	85.0±1.5	52.4±7.3	23.9±7.3	53.8±5.3

B.4 Sensitivity to training-map quantity

The data-quantity study holds training at approximately 3,200 optimizer updates and varies the number of distinct maps. Table 9 reports single-seed closed-loop results. LVT improves with more

distinct training maps and outperforms SFT at each tested sample count. Because feature supervision incurs target encoding, equal update counts do not match training computation.

Table 9: **Training-map quantity.** Single-run success (%) with fixed-size cells. Entries list 6×6 paths of length 7–12, 8×8 paths of length 7–12, and 8×8 paths of length 13–16, in that order.

Training Maps	LVT	SFT
1,000	85.7/84.7/53.3	38.0/42.0/13.3
3,000	94.7/97.3/75.0	54.3/59.3/13.3
12,921	98.3/98.7/96.7	68.3/72.7/26.7

B.5 Decoded planning states

The planning diagnostic decoder is trained on frozen perceptual states from 1,350 training maps, including blank maps, randomly selected expansion stages, and full-flood states. It is evaluated on 150 held-out maps. Figure 2 in the main text reports correct-arrow recovery on all 300 maps of each 7–12-step split for three seeds per configuration. Table 10 lists the three-seed means on the fresh split; on the custom split the per-iteration values differ by at most 0.7 points. No looped student state is used to train this decoder. Correct-arrow coverage measures the fraction of the final target arrow field whose direction can be recovered from decoded images. False arrows outside that target field are measured separately.

Table 10: **Decoded planning state after each iteration.** Correct-arrow recovery (%) of the final arrow field from the decoded state on 300 fresh maps (one call on the prompt map, 20 iterations; three-seed means). *Walk*: following the decoded arrows from the player reaches the goal at iteration 20. *Outside*: decoded arrows on cells outside the target field at iteration 20, as a percentage of the field.

	Iteration										
Training targets	1	2	3	4	5	6	8	12	20	Walk	Outside
Target progression	7.4	16.7	26.3	37.0	49.2	61.8	84.2	100	100	–	–
Trained at 384 px											
Ordered targets	4.0	15.3	24.5	34.4	45.3	56.8	76.2	89.4	89.6	21.2	0.7
Final target only	1.8	18.9	34.0	47.7	58.7	65.6	74.4	77.7	77.6	2.7	2.2
Shuffled targets	23.6	29.5	32.0	33.6	34.6	35.6	37.6	40.3	42.9	0.4	2.7
Answer only	1.4	1.4	1.4	1.5	1.5	1.5	1.5	1.5	1.5	0.0	0.0
Trained at 256–512 px											
Ordered targets	6.5	16.4	26.0	36.5	48.7	61.1	82.7	97.4	97.4	59.5	0.1
Final target only	5.7	23.5	37.4	51.7	65.2	76.3	89.8	95.3	95.2	46.4	0.4
Answer only	1.6	1.6	1.5	1.5	1.5	1.4	1.4	1.4	1.4	0.0	0.0

At iteration 20, correct-arrow recovery is 89.6% for ordered supervision (97.4% with multi-resolution training), 77.6% for final-state-only supervision, 42.9% for shuffled targets, and 1.5% for answer-only recurrence. Ordered states recover 84.0% of the rings beyond the six-step training limit (94.1% with multi-resolution training). In an earlier 60-map diagnostic with rounded measurements, the ungated model peaks near 91% at iteration 10 and falls to 79% by iteration 20, and the full-flood variant reaches approximately 95% recovery but only 84.7% closed-loop success, compared with 98.7% for the principal run. Recoverable field structure therefore does not by itself determine planning success: paths traced through the decoded field reach the goal on 21.2% of maps for the principal configuration (59.5% with multi-resolution training). Answer-only states have low arrow recovery, yet planting them changes the answer (Appendix B.6). State use and visual readability are distinct properties. Figure 3a selects iterations 1, 2, 3, 4, 6, 10, and 20 from the example below, with target fields and the ordered, shuffled, and answer-only conditions.

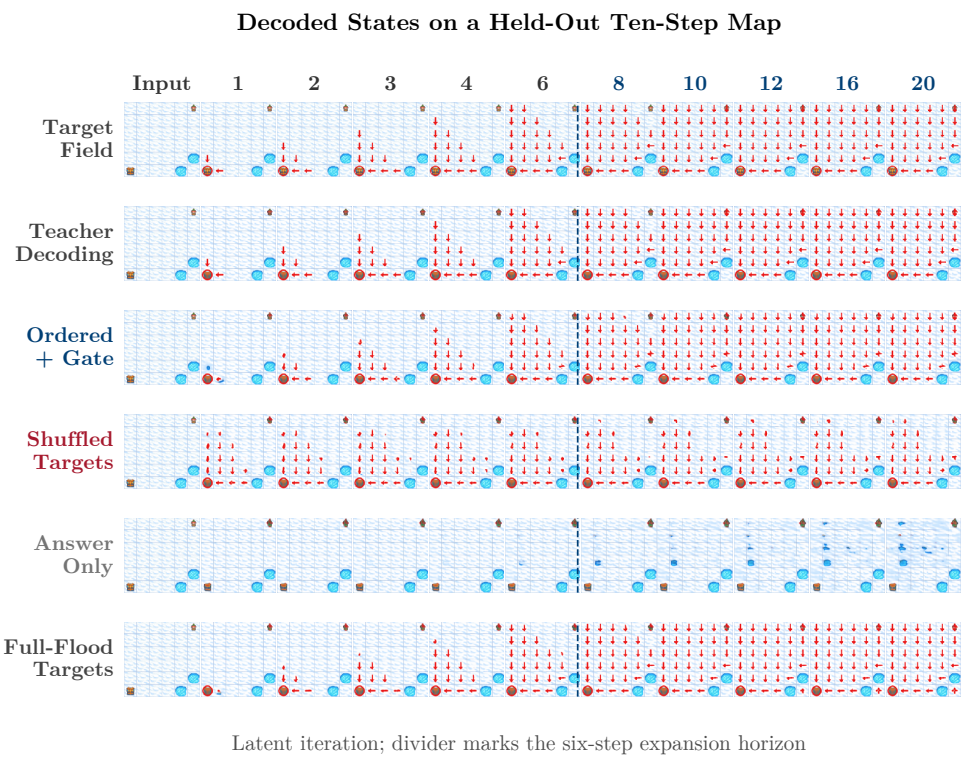


Figure 4: **A ten-step planning example.** Target frames, perceptual-decoder controls, and decoded recurrent states for ordered, shuffled, answer-only, and full-flood training. The ordered model develops an arrow field even though its single open-loop plan on this example is incorrect. These diagnostic reconstructions are never supplied to the student during inference.

B.6 Planning-state interventions

Interventions use a single decode with 20 updates on the 52 in-distribution test maps and the first 120 fresh maps, with three seeds per configuration (Table 11). For planting, the moved goal g' is chosen so that its optimal first moves are disjoint from those toward the original goal; the input image, prelude, re-injected input, and question still show the original goal. The state is replaced at iteration 20, at iteration 12 followed by eight updates on the original input, or at iteration $\lceil d'/2 \rceil$, where d' is the path length to g'; the three timings agree within two points. Planting a state from the model's own run on the same map leaves the answer unchanged on all 7 test and 8 fresh maps for every configuration. Answer-only and shuffled models choose the optimal first action less often on fresh maps (78–85%), so their control-wall changes set the noise level for their input-only rerouting.

B.7 Latent visual token baseline

We reimplement Mirage (Yang et al., 2026) from its public training code on the SFT configuration: the same backbone, rank-16 LoRA, optimizer, learning rate, batch size, 384-pixel images, and training maps, with 3,228 updates (SFT: 3,230) and a frozen vision encoder. Before the answer, the model emits a start token, $K = 4$ latent positions, and an end token, using three unused special tokens of the tokenizer. In the first stage (two epochs), the vision encoder embeds the helper image, its tokens are averaged over four contiguous chunks, and the four resulting vectors z_k are inserted at the latent positions; the loss is $0.1\,\mathcal{L}_{\mathrm{CE}} + \frac{1}{K}\sum_k \big(1 - \cos(h_{k-1}, z_k)\big)$, where h_{k-1} is the final hidden state at the preceding position. In the second stage (two epochs), each latent position receives the final hidden state at the preceding position, and only the answer loss is used. At inference, the latent block is inserted before decoding and regenerated in every closed-loop call. The route helper draws the optimal path on the map, as in Mirage's VSP setup (three seeds); the arrow-field helper is LVT's final target frame, the complete breadth-first arrow field (one seed).

Table 11: **VSP planning-state interventions (%).** Single decode with 20 updates; entries give the range over three seeds as in-distribution test / fresh maps. Planting replaces the state with one computed after moving the goal and reports first actions that follow the moved goal ($n = 49$ / 112). Walls block the first optimal step only in the recurrent state or only in the re-injected input and report rerouted first actions ($n = 22$ / 44). The control wall leaves the optimal path unchanged; the last column reports unchanged first actions on fresh maps. MR: multi-resolution training. Mirage rows apply the same edits to its four latent vectors (Appendix B.7); their fresh-map planted-goal and state-only-wall entries reflect the unedited model's own first actions, since both edits leave the plan text unchanged on 93–100% of maps. † One seed.

Configuration	Planted Goal Followed	State-Only Wall Rerouted	Input-Only Wall Rerouted	Control Wall Unchanged
Ordered	98–100/100	96–100/86–96	23–32/52–64	98–100
Ordered (MR)	100/100	96–100/96–98	73–77/73–86	100
Final-State-Only	96–100/100	96/86–89	0–4/2–9	93–98
Final-State-Only (MR)	100/100	96–100/96–98	9–18/2–7	98–100
Shuffled	98/99–100	82–91/48–52	0–4/11	80–84
Answer-Only	96–98/99–100	82–86/55–66	0–4/16–18	73–77
Answer-Only (MR)	96–100/99–100	77–96/59–75	0/14–18	75–82
Mirage (route helper)	2–4/19–22	5–9/11–18	82/55–59	70–75
Mirage (arrow-field helper)†	4/26	9/18	77/41	68

On the official open-loop test, route and arrow-field helpers reach $94.2 \pm 0.1\%$ and 94.5%, respectively. Under the closed-loop protocol on larger grids (Table 2), the route helper gives 73.1 ± 2.8, 61.3 ± 4.1, 16.7 ± 3.3, and $58.9 \pm 5.9\%$ on 7×7, 8×8, 8×8 with 13–16-step paths, and 9×9 maps, and $45.6 \pm 1.9\%$ on 10×10 maps (SFT: $47.2 \pm 4.2\%$); the arrow-field helper gives 69.3, 60.7, 13.3, 60.0, and 45.0% (Table 12).

Table 12: **Latent visual tokens (Mirage) on VSP (%).** Same backbone, adapter, optimizer, training maps, and number of updates as SFT. Official test: whole plan in one call; other columns: closed-loop execution as in Table 2. Helper image: the route (three seeds, mean±s.d.) or the final arrow field (one seed).

Method	Official	6×6 fresh	7×7	8×8	9×9	10×10	8×8, 13–16
SFT	93.9±0.6	57.4±0.7	75.3±1.2	62.4±1.7	64.4±1.9	47.2±4.2	18.3±1.7
Mirage (route)	94.2±0.1	57.7±0.9	73.1±2.8	61.3±4.1	58.9±5.9	45.6±1.9	16.7±3.3
Mirage (field)	94.5	51.0	69.3	60.7	60.0	45.0	13.3
LVT	**96.2**±0.1	**97.0**±0.3	**98.4**±1.0	**96.7**±1.3	**88.3**±5.8	**81.1**±6.7	**80.6**±3.5

Interventions use the maps, moved goals, and wall cells of Appendix B.6, with the four latent vectors as the state. Planting replaces them with the latents computed on the map with the moved goal; state-only walls compute the latents on the walled map but keep the original input, and input-only walls do the reverse. Planted and state-only-wall latents leave the generated plan text unchanged on 96–100% of test and 93–98% of fresh maps, and an input-only wall produces the same text as the walled map with its own latents on 100% and 91–95% of maps. Replacing the latents with those of another map, with zeros, or with norm-matched Gaussian noise leaves the text unchanged on 96–100% of test and 88–98% of fresh maps. In closed-loop evaluation on fresh maps, zeroed and noised latents give 57.3 and 57.3% for the route helper (unedited: 57.0%) and 50.7 and 51.0% for the arrow-field helper (51.0%).

The readability probe is a per-cell multinomial logistic regression (36 cells; four arrows or none) on standardized features. It is fit on latents from 1,899 6×6 training maps, with weight decay chosen on a held-out 20%, and scored on the arrow-field cells of 100 in-distribution test maps and the 300 fresh maps. All four latents give 45–51% (route helper) and 61% (arrow-field helper) on test maps and 58–60% on fresh maps; the first latent alone, computed before any latent position, gives 47–52%

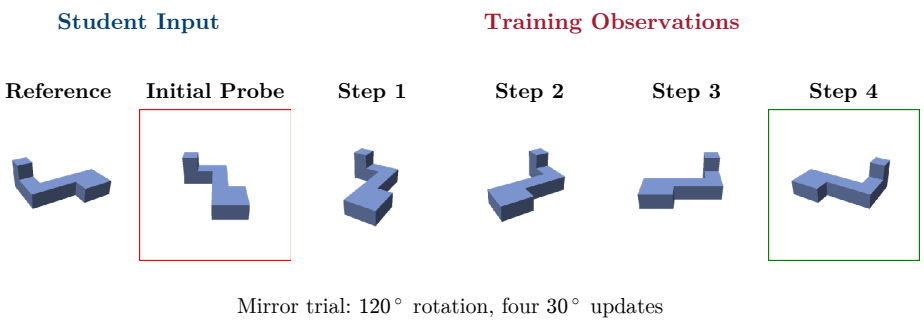


Figure 5: **Spatial sequences for evaluating LVT.** The reference and initial probe form the student input. Subsequent observations supervise the recurrent visual state through the frozen backbone. This mirror trial uses $\Delta = 120°$ and $R = 4$; the final observation is the reflected reference view.

and 55–57%. All four latents recover complete fields on at most 4% of test maps. The probe is fit on latent states, whereas the LVT decoder is trained only on frozen perceptual states; the two recovery measurements use different diagnostic protocols. For the arrow-field helper, the latents are closer to the encoding of the same map without arrows (mean cosine 0.68) than to its own final arrow field (0.64) on every test and fresh map.

C Mental Rotation: Data and Training

C.1 Comparison protocol and observation sequences

We use Shepard–Metzler object comparisons (Shepard & Metzler, 1971) to obtain an explicit target for each visual update. The context contains a reference image and an instruction; the initial probe depicts either the same object or its mirror under a rotation Δ specified in the text. The model predicts *same* or *mirror*. The target sequence rotates the probe back in 30° increments, with $R = \max(1, \Delta/30°)$; zero-angle trials use one identity target. The final observation matches the reference for *same* trials and its left–right reflection up to shading for *mirror* trials (Fig. 5). Intermediate observations are available only as training targets.

Objects are chains of ten cubes, filtered for chirality and absence of rotational symmetry about the chosen axis, and de-duplicated up to rotation and mirroring. The training, validation, and test splits contain 2,400, 300, and 300 disjoint objects. Training uses 48,000 trials with $\Delta \leq 120°$ and at most four recurrent updates. Testing uses 4,200 trials on new objects. The main metric is accuracy on the 1,200 trials at 150° and 180° (600 per angle), which require five and six updates. This protocol tests generalization across objects and beyond the training horizon. The primary experiment rotates objects about the vertical axis; image-plane rotation uses a separately trained model and a separate object split from the same shape family. Rendering details follow below; image-plane experiments are specified in Appendix F.1.

Shapes. An object is a chain of ten unit cubes made of $n \in \{4, 5, 6\}$ straight arms; consecutive arms share a corner cube and are perpendicular, arm lengths are at least two cubes, and arm lengths are sampled uniformly from all compositions of $10 + n - 1$ into n parts. Chains that self-intersect are rejected. A shape is kept only if (i) it is chiral, i.e. the canonical form of its mirror image under the 24 proper rotations of the cube differs from its own, and (ii) it has no rotational symmetry about the rendering axis (checked for rotations by 90°, 180° and 270° in the cube lattice; only these can be exact symmetries, so the twelve 30° views of a kept shape are pairwise distinct). Shapes are de-duplicated by the canonical form of the pair {shape, mirror}, so no test shape is a rotation or a mirror image of a training shape. With four arms only a few hundred distinct chiral chains exist, which is why five- and six-arm chains are included; 3,000 shapes are drawn with a fixed seed.

Rendering. A dependency-free rasteriser (numpy) draws the exterior faces of the cubes with flat Lambertian shading (ambient 0.35, diffuse 0.65, light direction $(-0.35, 0.65, -0.68)$ in camera coordinates), 2× supersampling and a white background. The camera sits in the plane x=0 at an

elevation of $30°$, at distance $4r$ from the centroid, where r is the object's bounding radius; the focal length is set so that the object fills 92% of the frame at its closest point, which keeps the scale identical across the views of an object and its mirror. The object is rotated about $+y$ by $30°a$ for $a = 0..11$; the mirror object is $\mathrm{mirror}_x(O)$, rendered the same way. Images are 320×320. Because the camera lies in the mirror plane, the view of the mirror object at the reference pose is the left–right flip of the reference view up to shading (the light is not left–right symmetric, so shading of side faces differs). 72,000 images in total; a checker script verifies for every trial that the last frame is the reference view (*same*) or has the silhouette of its flip (*mirror*).

Trials. Angle indices are integers modulo 12 with $\Delta = 30d$, $d \in \{0..6\}$, and direction $s \in \{+1, -1\}$ (s=+1, "clockwise", throughout the main experiments). For *same* trials the probe is O at index $\theta_0 + sd$ and frame k is O at $\theta_0 + sd - sk$. For *mirror* trials about the vertical axis, $\mathrm{mirror}_x R_y(\theta) = R_y(-\theta)\,\mathrm{mirror}_x$, so the probe is M at index $sd - \theta_0$ and frame k is M at $sd - \theta_0 - sk$; the last frame, M at $-\theta_0$, is mirror_x of the posed reference. About the horizontal axis mirror_x commutes with R_x, so the mirror probe is M at $\theta_0 + sd$ (the vertical-axis construction instead uses the anti-commuting relation above). For d=0 the process is the identity: one iteration whose target is the probe view itself. Per object, split and label we draw two start angles θ_0 for training and one for evaluation, giving 67,200 / 4,200 / 4,200 trials before the training-angle filter.

C.2 Training details and hyperparameters

Table 13 specifies the mental-rotation adaptation settings and differences among the reported runs. VSP settings are given separately in Appendix B.

Table 13: Qwen3-VL-4B mental-rotation hyperparameters. Default settings and run-specific overrides.

Setting	Specification
Backbone	Qwen3-VL-4B-Instruct, bf16, SDPA attention, all weights frozen (vision encoder included)
Recurrent Block	decoder layers 12–15 of 36 (c7 variant: 8–11); weight-tied across iterations
Adapters	LoRA rank 16, α=32, on q/k/v/o and gate/up/down projections of all 36 layers; the recurrent block's adapters shared across iterations (b9: one set per iteration)
Injection	$W_{\mathrm{in}} \in \mathbb{R}^{2560\times 5120}$ initialised to $[I\ 0]$, fp32, applied at probe-image positions before every block application
Update Rule	iteration 1 rewrites all positions; iterations ≥ 2 rewrite probe-image positions only
Block Attention	causal, plus bidirectional within each image block; keys of padding masked; c9: probe queries cannot attend to reference-image keys
Halting Head	linear on the mean of RMS-normalised probe states after each iteration; BCE against "r=R", weight 0.1 (0 for c3, c16)
Feature Loss	RMS-normalise student and teacher states; subtract an EMA (momentum 0.99) of the per-position teacher mean; $1 - \cos$ per token; weight 1 on tokens in the union of the foreground of probe and all frames, 0.1 elsewhere (foreground = any pixel darker than 245 in the token's 32×32 cell)
Loss Weights	answer 1.0; feature λ: decayed runs 1.0 for the first 35% of steps then 0.1; constant runs 1.0 (c8/c13: 0.3); c11: answer 2.0; c16: answer 0
Optimiser	AdamW, lr 10^{-4}, weight decay 0.01, $\beta = (0.9, 0.98)$, 100 warm-up steps then cosine to 0, gradient clipping 1.0
Batch / Steps	8 trials $\times$ 2 accumulation; 2 epochs over 48,000 trials = 6,000 steps (c10/c15: 3 epochs); full backpropagation through all iterations
Loops per Trial	$R = \max(1, \Delta/30°) \leq 4$ in training; the batch runs to its maximum R and samples with smaller R stop updating
Teacher	same network, adapters disabled, prompt identical, probe slot = frame k; one batched pass through the prelude and recurrent block for all frames; forward stops at the block output
Seeds	$0, 1, 2$ where three seeds are reported; seed controls initialisation of adapters and data order

Implementation and evaluation checks. The reported evaluation uses identity targets for zero-angle trials, transformation-specific mirror-index arithmetic, and the loss schedules recorded in each run configuration. Dataset checks verify the relation between the final observation and the reference for both labels. Representation diagnostics use centered, foreground-weighted similarities, and expanded candidate sets include the reference object's views. These checks validate both.

C.3 Prompt for spatial comparison

```
<|im_start|>user
You are shown two 3D objects made of cubes, each viewed
from the same camera.  The second object may have been
rotated about the vertical axis.<|vision_start|>[image
1]<|vision_end|><|vision_start|>[image 2]<|vision_end|>The
second object is either the first object rotated clockwise
by 90 degrees about the vertical axis, or the mirror image
of the first object rotated the same way.  Which is it?
Answer with one word:  same or mirror.<|im_end|>
<|im_start|>assistant
same<|im_end|>
```

D Additional Supervision and Update Ablations

Table 14 reports supervision and optimization controls for Secs. 4.3 and 5.3.

Table 14: **Training and visual-update ablations.** Unseen-angle accuracy (%) in the primary spatial reasoning task. The main setting uses $\lambda = 1$; decayed supervision reduces λ from 1 to 0.1 after 35% of training. Repeated conditions report mean $\pm$ standard deviation over three seeds. The b9 adapter-sharing control uses the decayed schedule and is compared with b2.

ID	Configuration	Accuracy (%)
c2	Ordered Sequence (constant weight; main configuration)	92.1 ± 0.3
c1	Final Observation at Every Iteration (constant weight)	68.8 ± 5.5
c3	Final Observation with a Single Block Application	69.2
c12	Input Injection Removed (constant weight)	87.2
c16	Visual Supervision Only; Answer and Halting Losses Removed	87.2
c9	Reference Attention Restricted Inside the Recurrent Block (constant weight)	96.5
c13	Ordered Sequence (constant $\lambda = 0.3$)	92.7
c15	Ordered Sequence (three training epochs)	94.6
c10	Final-View Supervision (three training epochs)	59.4
b2	Ordered Sequence (decayed weight)	85.3 ± 0.6
b9	Iteration-Specific Recurrent Block Adapters (decayed weight)	67.5
b3	Shuffled Sequence (decayed weight)	84.1 ± 1.0
b4	Final Observation at the Last Iteration Only (decayed weight)	78.8
b8	Final Observation at Every Iteration (decayed weight)	79.7 ± 8.9
c4	Input Injection Removed (decayed weight)	85.0

Planning controls. Table 15 gives the VSP configuration ablations on the custom long-path split. These are seed-0 results; the 98.7% principal run is one of the three runs summarized as $96.8 \pm 1.7\%$ in Table 1. Final-state-only and shuffled supervision are likewise shown as individual runs here and as three-seed means in the main comparison. Route-extension and wavefront targets test different observation content, while gate and attention ablations alter the recurrent update.

Large-angle errors. With angle text in the prompt, the non-recurrent fine-tuned model reaches approximately 23% at $180°$, while the ordered model reaches approximately 87%. The stronger SFT

Table 15: **VSP ablations.** Closed-loop success (%) on 300 custom extrapolation maps, seed 0. Three-seed means: main 96.8 ± 1.7, final-state-only 92.0 ± 0.3, shuffled 67.4 ± 2.7.

Configuration	Success	Configuration	Success
LVT (main configuration)	98.7	Final-State-Only Supervision	91.7
Shuffled Targets	65.0	Route-Extension Targets	60.7
Without Residual Gate	78.7	Colored-Wavefront Targets	61.3
Without Random-Phase Start	91.0	Monochrome-Wavefront Targets	64.3
Two-Layer Recurrent Block	91.7	Convergence-Based Stopping	97.7
Causal Within-Image Attention	69.7	Learned Answer-Quality Stopping	34.3

result without angle text ($64.3 \pm 1.8\%$; Table 3) shows that prompt formulation contributes to the original baseline's failure. The original comparison alone does not identify a specific visual shortcut.

Input injection and parameter sharing. Removing input injection under constant feature supervision (c12) retains 87.2% accuracy, but nearest-view retrieval is at most 8%, approximately chance. Under decayed supervision, the corresponding removal (c4) yields 85.0%, close to $85.3 \pm 0.6\%$ with injection. The initial representation consequently plays a larger role in maintaining an aligned trajectory than these accuracy differences alone would suggest. With the same decayed supervision as b2, iteration-specific recurrent block adapters (b9) reach 67.5%, compared with $85.3 \pm 0.6\%$ for shared adapters. The pretrained layers of the recurrent block remain shared in both variants; b9 changes the adapter parameters used at each iteration with additional iteration-specific adapters outside the four-step training horizon. This diagnostic changes the parameter count and uses iteration-specific adapters at untrained depths, so it does not isolate weight sharing.

Sequence order and terminal-view supervision. At constant feature-loss weight, assigning the same observation targets to shuffled iterations reduces accuracy from $92.1 \pm 0.3\%$ to 75.5% (Table 3) and disrupts trajectory alignment after the first iteration. Shuffling creates inconsistent iteration targets; the final-view controls below provide an additional comparison without this perturbation. Supervising every iteration with the final aligned observation (c1) gives $68.8 \pm 5.5\%$, despite producing strong terminal-view reconstructions: silhouette IoU with the aligned view increases from 0.51 to 0.76, and that view is retrieved in 50–67% of mirror trials with the expanded 24-view candidate set. A single-pass terminal-view model (c3) reaches 69.2%. For the repeated terminal-view condition, doubling the answer-loss weight gives 55.2%, while restricting reference attention gives $79.5 \pm 1.8\%$. Terminal-view reconstruction and task generalization are therefore distinct evaluation criteria.

Visual supervision without task-output losses. The model trained with visual supervision alone (c16) reaches 87.2% on unseen angles and 93.4% on trained angles. Its nearest-view retrieval is 83, 78, 58, 50, 45, and 30% over iterations 1–6. This condition tests the learning interface directly: observed states alone can train an update that the pretrained task readout uses for spatial comparison. It supports learning from visual experience in this controlled setting. It also suggests that maximizing answer accuracy and maximizing alignment with perceptual targets need not produce the same intermediate representation.

Supervision strength and optimization. A smaller constant feature weight, $\lambda = 0.3$ (c13), retains 92.7% accuracy but gives nearest-view retrieval of at most 16%. In these configurations, the strength of the visual objective affects state alignment more than task accuracy. Extending ordered-supervision training to three epochs (c15) raises accuracy to 94.6% and improves retrieval to 78, 73, 55, 50, 42, and 31% across iterations 1–6. Three epochs of terminal-view supervision (c10) instead yield 59.4%. Longer training therefore does not have the same effect across supervision regimes.

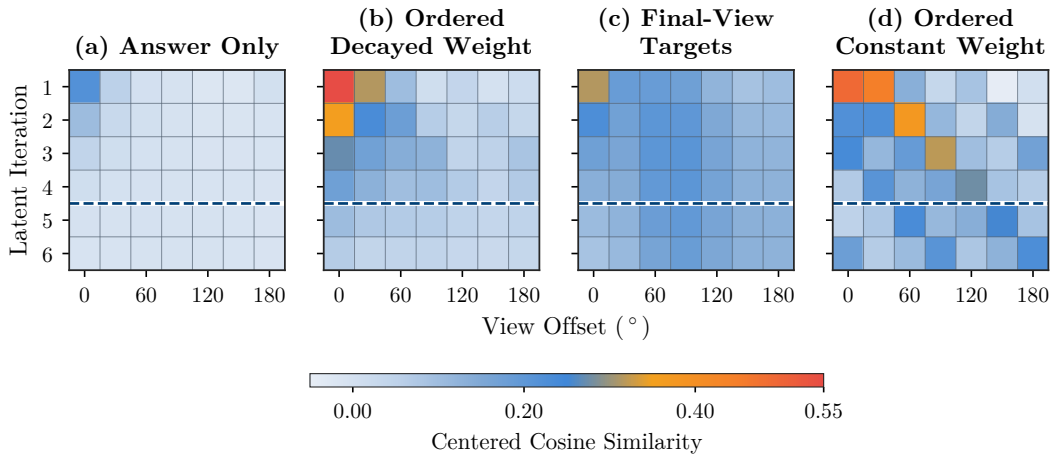


Figure 6: **Alignment between recurrent states and observed views.** Rows index iteration r and columns index the view rotated back by j steps. Similarities use centered, foreground-weighted cosine scores on diagnostic subsets of 96–128 held-out trials. Ordered supervision with a constant feature-loss weight produces a diagonal pattern beyond the maximum training depth of four.

E Analysis of Rotation States

E.1 Diagnostic decoders

Both decoders take the RMS-normalized, then centered output states of the recurrent block at the 100 probe positions (a $10\times10\times2560$ grid) and are trained only on teacher states of validation-split objects (160 and 240 objects respectively, all twelve views each), never on a looped state; the same per-position mean used for centering is stored with the decoder.

The *silhouette decoder* is a linear projection to 256 channels followed by two residual 3×3 convolutions and a 1×1 output over the token grid, trained with binary cross-entropy against the per-token foreground mask (2,000 steps, batch 64, AdamW 10^{-3}); on held-out validation objects it reaches 0.996 token accuracy and 0.98 IoU. A per-token *linear* probe on the same states reaches only 0.30 IoU: local content is present in the mid-layer states but entangled with context, and a decoder with spatial support is needed to read it.

The *image decoder* is a linear projection to 192 channels followed by five stages of nearest-neighbour up-sampling and two 3×3 convolutions each ($10 \to 320$ pixels), trained to predict the darkness image $1 - \text{gray}$ with an L_1 loss in which foreground pixels weigh seven times background pixels (5,000 steps, batch 32); a plain pixel L_1 collapses to an all-white image. On held-out objects it reaches 0.003 L_1 and 0.97 pixel-level foreground IoU. Recurrent-state images are shown at half resolution.

E.2 Alignment and reconstruction

Alignment with the observation sequence. The similarity matrices in Fig. 6 distinguish the recurrent states learned under different supervision regimes. With constant-weight ordered supervision (c2), the similarity peak advances along the target-view sequence and remains visible at iterations 5 and 6. Nearest-view retrieval is 56, 64, 47, 42, 25, and 22% over iterations 1–6, compared with a chance level of approximately 8.3%. The corresponding angular errors are 13, 20, 37, 46, 71, and 77°. Thus, alignment persists beyond the trained depth but becomes less precise as the rollout extends. At iteration 3, silhouette IoU is higher with the target than with the initial probe (0.62 versus 0.50).

For answer-only recurrence (b1), similarity to all perceptual views decreases with depth and the silhouette decoder produces unstructured output. Under decayed feature supervision (b2), similarity broadens toward later views without the same advancing peak. Thus, a successful final readout does not ensure that the intermediate state follows the observed trajectory. The recurrent visual state and its supervision must be evaluated together.

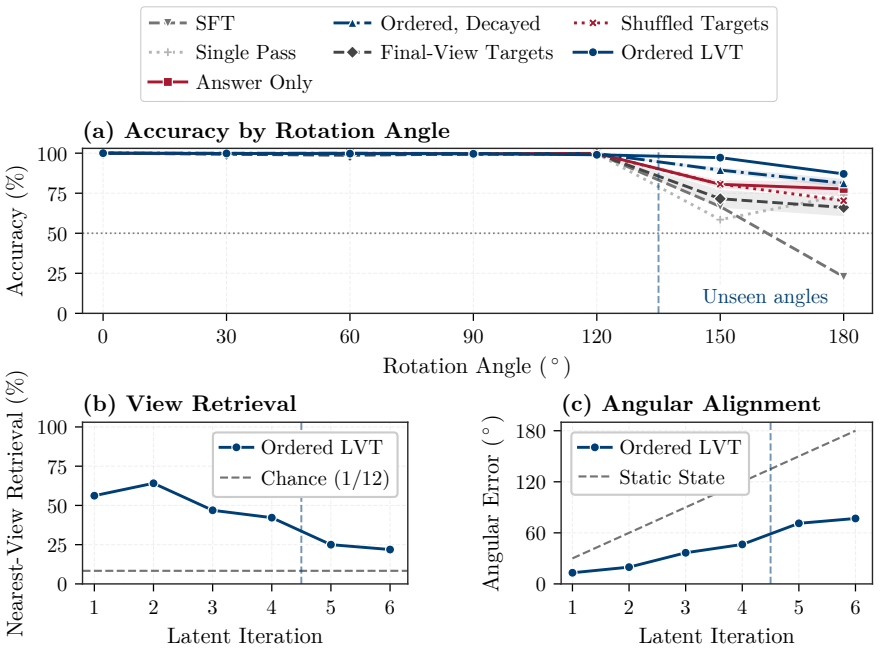


Figure 7: **Task accuracy and state alignment.** (a) Mean and population s.d. across three seeds, or individual runs. (b,c) Retrieval and angular error for c2, seed 0. Dashed lines mark unseen angles in (a) and the four-step training horizon in (b,c).

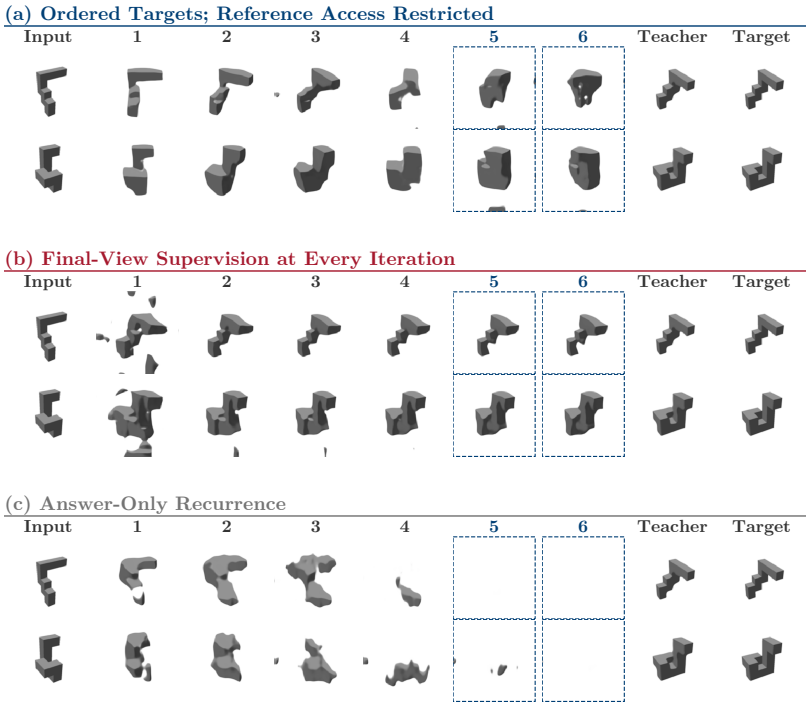


Figure 8: **Reconstructions of recurrent states.** Two held-out objects per block. Columns: input, iterations 1–6, teacher decoding, and aligned target. Conditions: **(a)** ordered targets with restricted reference access (c9); **(b)** repeated final targets (c1); **(c)** answer-only (b1). The decoder uses only validation perceptual states; dashed boxes mark untrained iterations 5–6. Full panels: Appendix I.

Table 16: **Reference-object retrieval control.** Results on mirror trials with $R \geq 2$, as percentages of trials. Left: retrieval of the final aligned probe view among the probe object's twelve views. Right: retrieval of the reference image after adding the reference object's twelve views. Columns index diagnostic iterations 1–6.

	Aligned View: 12 Candidates						Reference Image: 24 Candidates					
	1	2	3	4	5	6	1	2	3	4	5	6
b1 (answer only)	0	0	5	5	7	12	0	0	7	36	55	76
b2 (ordered, decayed)	0	0	0	20	27	39	0	5	5	57	100	100
b8 (final view, decayed)	2	12	24	37	39	39	2	24	43	71	88	90
b3 (shuffled, decayed)	0	0	0	0	0	2	0	0	0	0	0	2
c2 (ordered, constant)	See trajectory analysis in Sec. 5.1						0	5	7	21	21	55

Decoding intermediate states. The main-text rotation panel (Fig. 3b) uses the first two test objects in the archived reconstruction panels, comparing ordered supervision (c2) with answer-only recurrence (b1). The perceptual teacher-state controls and repeated-final-view condition remain in the supplementary panels. Figure 8 applies the same perceptual-state decoder to recurrent representations from different conditions. Ordered supervision produces identifiable object structure at successive orientations. For the terminal-view condition (c1), decoded states instead approach the aligned view from the second iteration onward, including mirror trials for which that view is absent from the input. Answer-only states yield progressively less structured reconstructions. The decoder's reconstruction of the teacher state provides a reference for interpreting these images. These reconstructions complement the supervision ablations: matching a terminal view and preserving a sequence of spatial states lead to different downstream behavior.

E.3 Context access and reference copying

Expanded retrieval candidates. LVT preserves contextual image tokens so that the evolving visual state can use them through attention. In this task, that access also creates a potential shortcut: the reference coincides with the final target on *same* trials. A retrieval analysis restricted to the probe object's views cannot fully distinguish target alignment from reference copying. We therefore add the reference object's twelve views to form a 24-candidate set and examine *mirror* trials, for which the aligned probe view differs from the reference. Table 16 reports retrieval of the aligned probe view in the original candidate set and retrieval of the reference image in the expanded set.

With decayed ordered supervision (b2), reference retrieval reaches 57% at iteration 4 and 100% at iteration 6. For decayed terminal-view supervision (b8), it reaches 71% and 90% at those iterations. The answer-only model also becomes increasingly aligned with the reference, whereas the shuffled decayed model has little reference retrieval despite similar task accuracy to b2. These results are consistent with a reference-copying shortcut under decayed weights, which motivates the constant-weight configuration.

Effect of supervision and reference access. Constant-weight supervision reduces early reference retrieval: for c2, the expanded candidate set selects the aligned probe view in 38% of mirror trials and the reference in 7% at iteration 3. Restricting probe-to-reference attention inside the recurrent block (c9) gives 33% aligned-view retrieval and 5% reference retrieval at iteration 3, and reaches 96.5% task accuracy (99% at 150° and 94% at 180°). Nearest-view retrieval of the step-specific target is 67, 73, 75, 58, 56, and 39% across iterations 1–6. This ablation shows that the pattern of context access matters alongside the recurrent update and its targets. The restriction is local to the recurrent block; reference information can still enter through the prelude.

E.4 Interventions and additional diagnostics

Final-state interventions. On 1,200 unseen-angle trials, replacing the final probe state with its first-iteration value reduces accuracy from 91.92% to 23.67%; replacing it with zeros gives 51.50%. Only the selected probe positions are replaced, immediately before the coda. Context states, model weights, and same/mirror scoring remain fixed. The evolved state therefore affects prediction.

Re-encoding an intermediate observation. After iteration $j \in \{1, 2\}$, we replace the probe state with the frozen teacher encoding of the corresponding observation and continue with the original injection anchor and iteration count. For the ordered model (c2), accuracy on the affected trials changes from 94 to 96% at $j = 1$ and from 90 to 94% at $j = 2$. Final-state cosine similarity to the aligned-frame teacher increases from 0.50 to 0.53 and from 0.46 to 0.58, respectively. For answer-only recurrence (b1), accuracy remains 96%, with final cosine similarity of 0.10–0.15 both with and without the replacement.

Longer transformation sequences. The primary model is trained with at most four updates and reaches approximately 97–98% at five and 87% at six (Fig. 7). Restricting training to three updates gives 90, 86, and 80% at four, five, and six.

Terminal-view supervision and reference access. Retrieval differs sharply between the two labels. Among the probe object's twelve views, b8 retrieves the aligned view at iteration 6 in 96% of *same* trials but only 39% of *mirror* trials; b1 reaches 78% and 12%, and b3 reaches 0% for both. These diagnostics use trials with $R \geq 2$. On same-object trials, the aligned probe and reference views coincide, so retrieval alone cannot distinguish transformation from copying.

Terminal-view supervision also varies substantially across seeds. With decayed feature weights, b8 reaches 85.9, 67.1, and 86.2%; with constant weights, c1 reaches 63.7, 76.4, and 66.3%. Ordered trajectory supervision is more consistent in the reported runs (c2: 92.0, 92.6, and 91.8%).

F Additional Rotation Settings

F.1 Rotation in the image plane

Data. A third set of 3,000 chiral objects (generation seed 400; the same shape family and filters as Appendix C) is rendered with the same camera, after which the rotation is applied to the camera-space coordinates so that the image turns rigidly in its own plane by $30°a$ for $a = 0..11$. Mirroring anti-commutes with this rotation exactly as for the vertical axis, so the trial construction is unchanged: the mirror probe is M at index $sd - \theta_0$ and frame k is M at $sd - \theta_0 - sk$; the checker script confirms every last frame. Training uses $\Delta \leq 120°$ and testing $\Delta \leq 180°$ as in the main experiments; hyper-parameters are those of Appendix C.2. Three seeds per condition.

Per-seed results. Unseen-angle accuracy (%): answer only 99.4, 90.2, 94.7; shuffled frames 62.1, 57.4, 61.0; ordered frames 98.2, 98.2, 95.4; no loop 68.2 (single run). At 180°: answer only 99.2, 81.7, 90.2; shuffled 34.7, 25.7, 32.8; ordered 96.7, 96.7, 92.0. In-distribution accuracy is 99.9–100% for every model. Nearest-view retrieval of frame r for ordered frames, per seed and iteration: 88/83/81/73/70/72, 86/81/83/72/62/70 and 88/83/80/70/59/61%. Figure 9 shows the loop-by-view similarity matrices; the ordered-frame matrix has a secondary ridge six views away (the object turned by 180° in the plane shares much of its appearance), which is absent for the other axes. Decoded states are shown in Fig. 13; Table 17 summarises the conditions.

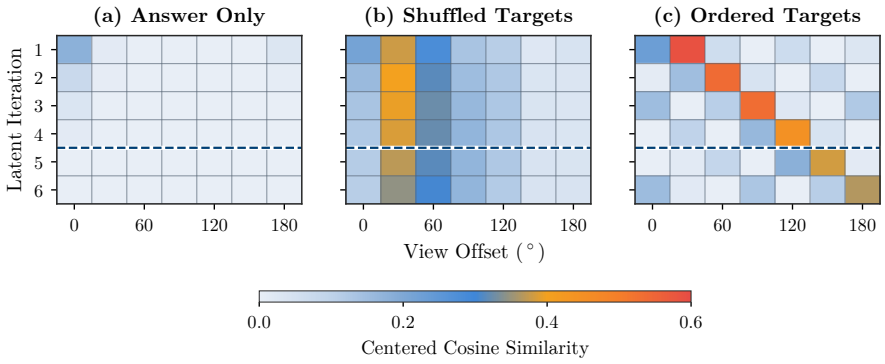


Figure 9: **Rotation in the image plane: loop-by-view similarity** (centred, foreground-weighted cosine between the state after iteration r and the teacher states of the probe object rotated back by j steps; 64 test objects, seed 0). The dashed line marks the largest depth run during training. The ordered-frame model (right) has a diagonal pattern extending into the two untrained depths, together with the secondary ridge described in the text. The shuffled condition (middle) has weak later-step alignment; answer-only recurrence (left) has low similarity to the candidate perceptual states.

Table 17: Recurrent models for rotation in the image plane (three seeds; mean $\pm$ s.d.). Retrieval: nearest of the twelve views to the state after iteration r is frame r (chance 8%); angle error is that of the nearest view ($30r^\circ$ for a static state). The non-recurrent control reaches 68.2% accuracy.

Run	Supervision	Accuracy (%)	Retrieval (%) $r = 1, \ldots, 6$	Angular Error (°) $r = 1, \ldots, 6$
z-b1	Answer Only	94.8 ± 4.6	$2, 8, 7, 10, 3, 8$	$30, 58, 80, 91, 92, 98$
z-c17	Shuffled Targets	60.2 ± 2.5	$34, 34, 13, 11, 1, 0$	$31, 27, 43, 66, 96, 125$
z-c2	Ordered Targets	$\mathbf{97.3 \pm 1.6}$	$\mathbf{87, 82, 81, 72, 64, 68}$	$\mathbf{4, 9, 13, 17, 30, 36}$

G Additional Tasks

These extensions use different data, readouts, or stopping policies; their scores are reported separately from the principal evaluations. Results are single runs unless repeats are specified.

G.1 Synthetic grid updates with convergence stopping

Fix-lite contains flood-fill, gravity, and left/up ray-propagation tasks, with instructions to continue until the state stops changing and no supplied step count. Training covers depths 1–12 with 0–4 terminal holding steps. From iteration 2, inference stops when the *maximum* relative change over mutable-image tokens falls below a threshold chosen by log-scale Otsu separation on validation data. Fixed thresholds of 0.15–0.2 give similar results. This differs from the mean RMS-normalized directional-change statistic in the VSP convergence variant. At depths 1–12, exact-match accuracy is 100, 100, 97, 100, 100, 96, 98, 98, 97, 99, 98, and 99% (100 examples per depth). Held-out flood-fill depths 13–17 give 96, 96, 92, 92, and 92% (25 examples per depth).

Path-lite uses 16×16 mazes with wavefront expansion and deterministic backtracking. Training on 16-step windows, weighting changed cells, and decoding and re-rendering grids between windows gives 90.0% exact match on 200 examples at depths 12–40, versus 0% for SFT. Training includes long trajectories, so the test extends beyond the unroll window, rather than beyond all training depths. The task-specific readout and decoded-grid feedback differ from the main VSP protocol.

ARC-lite tests five known rules on 10×10 grids with a supplied step count. After training at depths at most four, ordered supervision gives 81, 76, and 67% exact match at depths 5–7 (100 examples each), versus 13, 0, and 0% for answer-only recurrence and 1, 0, and 0% for SFT. This tests synthetic rule execution, not official ARC-AGI rule induction.

G.2 MAZE-HARD AND EXPLICIT STATE READOUTS

Training and scoring. We use AdamW at learning rate 10^{-4} and effective batch size 16. Cell cross-entropy assigns weight 1 to free cells and 0.2 to other cells. A solver using eight grid steps per update for wavefront expansion and path tracing prescribes 28–38 iterations. Training adds up to three terminal holding updates. The partial-path variant redraws half the training inputs with a random prefix of the correct path, sometimes inserting or omitting path cells, and extends holding to eight updates. All test inputs are original, unsolved mazes. Rank-32 LoRA uses scaling parameter 64; 1,000 and 2,000 updates correspond to 16 and 32 epochs. Scoring maps wavefront classes to free cells and requires the complete canonical solution. SFT reaches 90.1% cell accuracy and 0% exact match.

Feedback and stopping. For the 2,000-update model, decoding, rendering, and re-encoding between six 16-iteration chunks gives 62.5% exact match. A separate board-state variant uses an arrow field for local propagation and path construction, trained on four-step windows of ordered board targets. Soft-probability feedback reaches 51.6%; all 1,000 examples stop when the decoded board is unchanged, averaging 201 iterations.

H CONTEXTUAL COMPARISON WITH API REASONING MODELS

On a separate Shepard–Metzler probe, we measure API-model accuracy and reported reasoning-token usage (Fig. 10; Table 18). The 400 trials use 10 objects, image-plane and depth rotations from 0° to 180° in 20° increments, and both same and mirror labels.

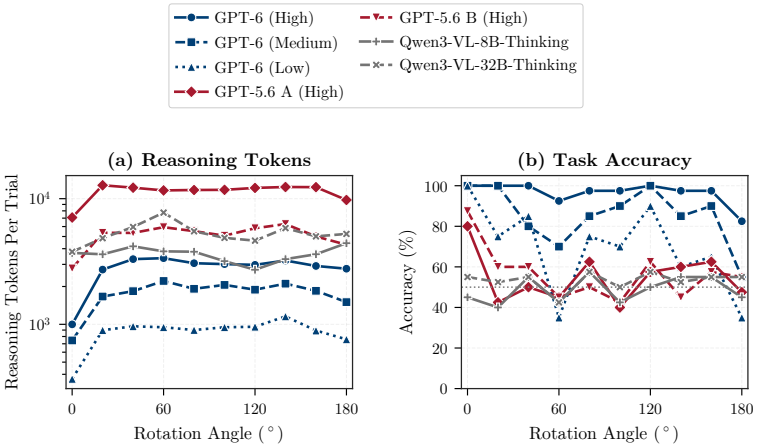


Figure 10: **API reasoning models on the spatial probe.** Accuracy and mean reported reasoning tokens (log scale) by angle, recomputed from available logs. Means include zero counts.

GPT-6 at high reasoning effort reaches 96.5% overall and 82% at 180°; lower effort settings obtain lower accuracy in their evaluated subsets. Mean token usage rises from about 1k at 0° to 3k at 20° and is roughly flat thereafter. GPT-5.6 and Qwen3-VL-8B/32B-Thinking remain near chance despite higher token counts in several settings. Separate text path-integration controls reach 94–100% with fewer than 100 reported reasoning tokens on average.

Table 18: **API reasoning-model probes.** Accuracy (%) and mean reasoning tokens from the trial logs, including zero counts. Visual rotation and textual path integration are separate tasks.

Model (reasoning effort)	n	Accuracy	Reasoning Tokens
Visual Rotation			
GPT-6 (high)	400	96.5	2,835
GPT-6 (medium)	200	85.5	1,778
GPT-6 (low)	200	69.0	879
GPT-6 (high; 18-cube objects)	400	95.8	4,631
GPT-5.6 A (high)	400	54.8	11,391
GPT-5.6 B (high)	400	56.5	5,140
Qwen3-VL-8B-Thinking	400	48.8	3,632
Qwen3-VL-32B-Thinking	400	53.3	5,343
Textual Path Integration			
GPT-6 (high)	72	100.0	53
GPT-5.6 A (high)	72	98.6	99
GPT-5.6 B (high)	72	94.4	81

These measurements do not reveal internal visual representations. LVT generates no intermediate text but executes $32 + 4R$ decoder-layer applications, plus visual encoding and projection (Sec. 4.2). Differences in objects, cameras, angle grids, and answer scoring preclude a controlled accuracy or efficiency comparison; the text controls are separate tasks.

I Additional Reconstructions

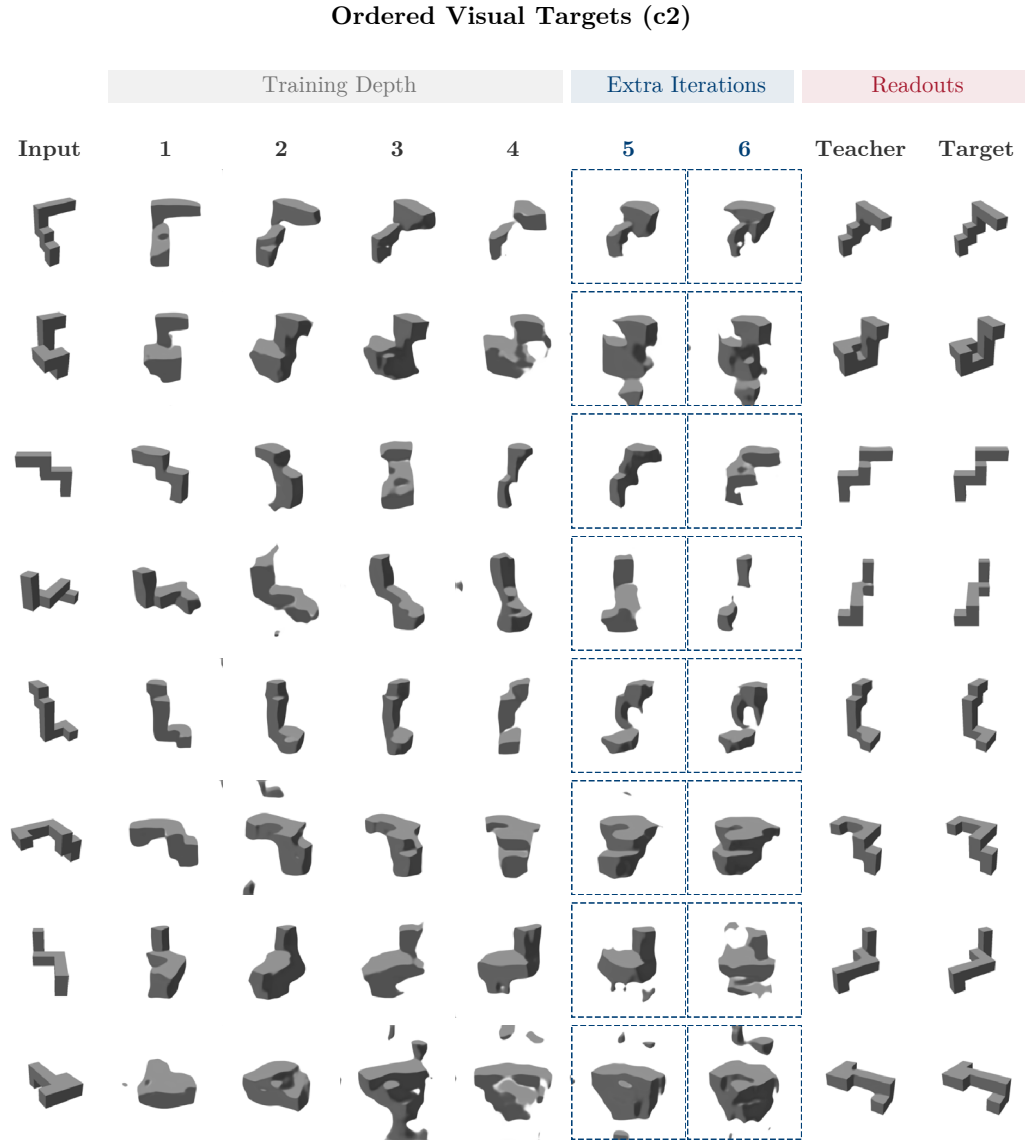


Figure 11: c2 (ordered frames, constant weight): decoded states for eight test objects. Columns show the input, iterations 1–6, the perceptual teacher-state decoding, and the aligned target. Dashed boxes mark iterations beyond the four-step training horizon.

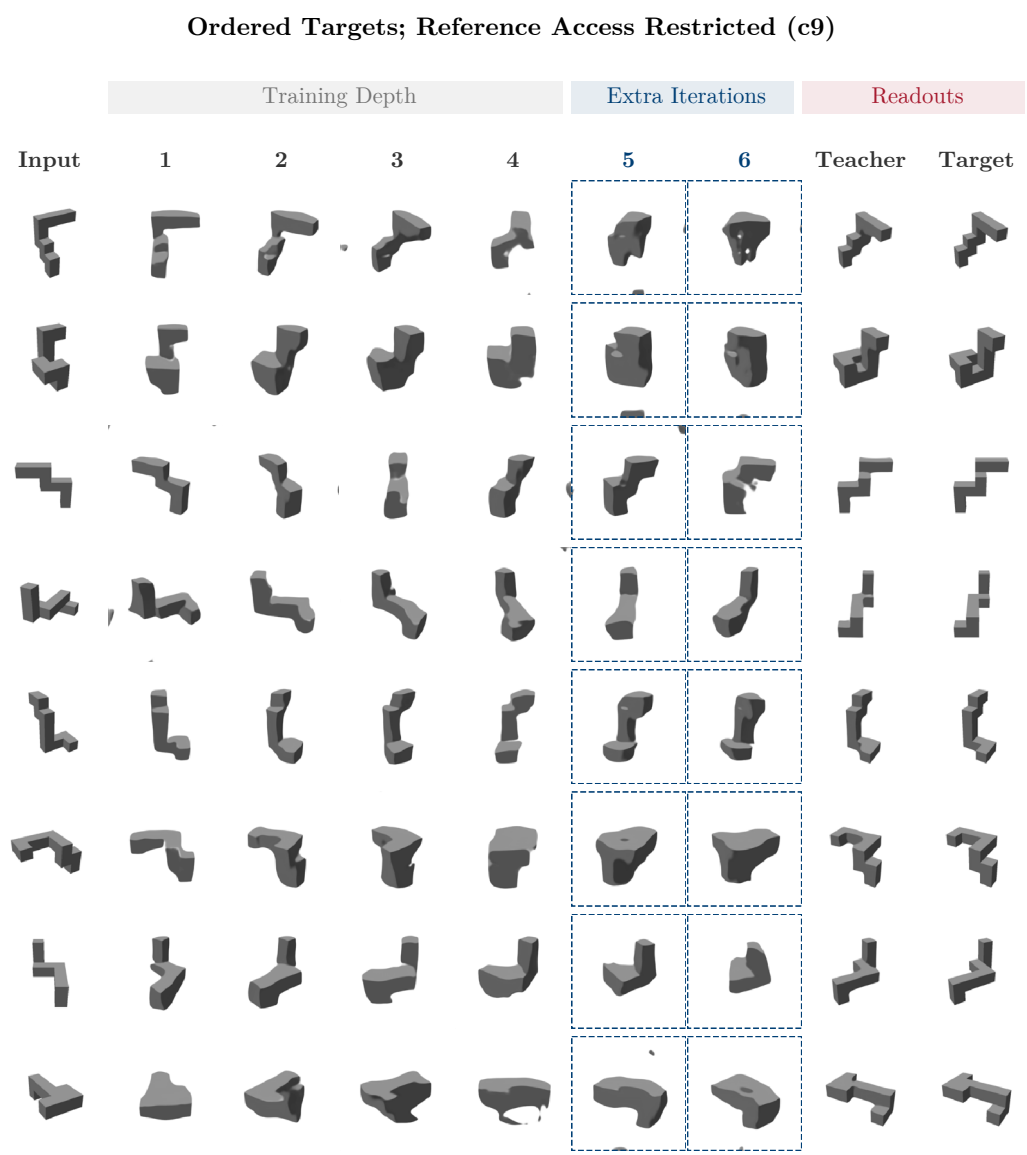


Figure 12: c9: decoded states for the same eight objects, with probe-to-reference attention restricted inside the recurrent block. Column definitions and dashed boxes follow the c2 reconstruction panel.

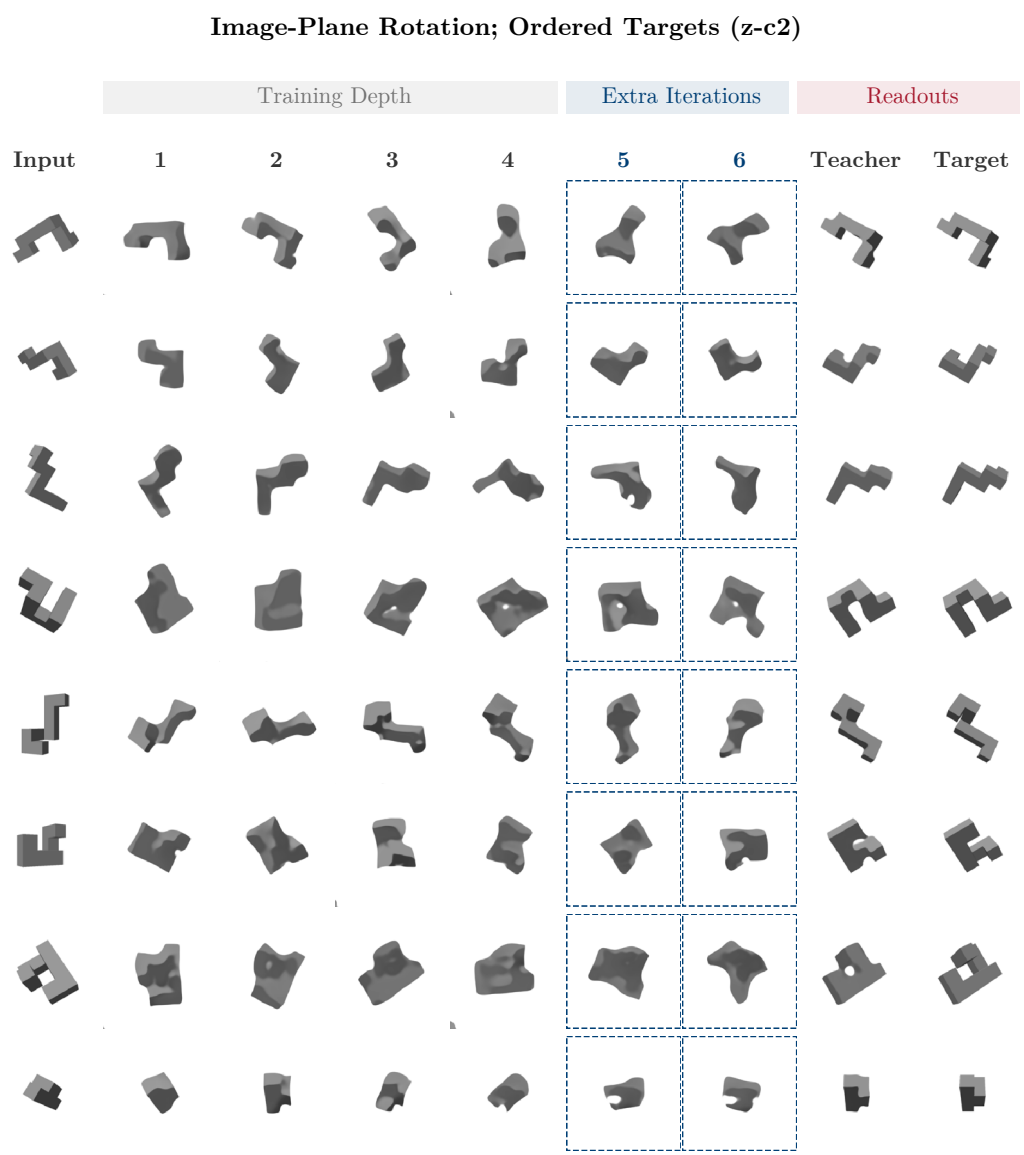


Figure 13: Rotation in the image plane (z-c2): decoded states for eight test objects. The decoder is retrained on perceptual states of validation objects of this data set (held-out pixel IoU 0.96). In the 24-candidate reference control on mirror trials the nearest candidate is the aligned view of the probe object for 20, 38 and 31% of trials at iterations 2–4 and the reference view for at most 4%, providing evidence of target-specific information on this diagnostic subset.