SCAFFOLDING MINDS: OPTIMIZING LATENT VISUAL TARGET REPRESENTATIONS FOR MULTIMODAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent reasoning has advanced multimodal reasoning through a two-stage training paradigm: (1) a helper image is encoded into latent tokens to teach visual chain-of-thought during a *supervised fine-tuning (SFT) stage*, and (2) these latent tokens are further refined with reward feedback during a *reinforcement learning (RL) stage*. In this paper, we identify two key limitations of this framework, one in each stage. First, the SFT stage typically relies on an off-the-shelf vision encoder to encode the helper image, yielding suboptimal latent representations that may not be well aligned with the downstream reasoning task. Second, existing RL methods treat the latent component only through deterministic regularization, which constrains policy drift but does not create alternative latent trajectories for exploration. To address these limitations, we propose **Scaffolding Minds**. Our approach learns a dedicated scaffolding encoder that provides an optimized target in latent space, and learns both the mean and variance of the RL sampler. We further show that these two improvements are complementary, together yielding substantial gains over strong baselines. Empirically, our method improves over the strongest latent-reasoning baseline by **+9.5** points on FrozenLake spatial planning, with the gain widening to **+19** points on the 32×32 grids, and by **+5.6** points on average across nine visual-centric reasoning benchmarks.

1 INTRODUCTION

Vision-language models (VLMs) have become increasingly capable at multimodal reasoning (Bai et al., 2025; Liu et al., 2023; Gemini Team, 2023). Methods such as Wei et al. (2022); Kojima et al. (2022); Xu et al. (2024); Zhang et al. (2024) rely on textual chain-of-thought, but lose important visual details (Wu & Xie, 2024; Fu et al., 2024; Tong et al., 2024b). To bridge this gap, latent visual reasoning (Yang et al., 2025b; Li et al., 2025b; Wang et al., 2026; Jeon et al., 2026; Qin et al., 2025; Zhang et al., 2025a; Tong et al., 2025; Chen et al., 2025) has emerged as a compelling solution. By interleaving small blocks of latent visual tokens directly into the reasoning chain, these models can "think" in both text and vision. Typically, these methods follow a two-stage training paradigm. In Stage 1 (SFT), they leverage a *helper image* (e.g., an annotated crop) that is available only during training to provide guidance in the latent space (encoded by an off-the-shelf vision encoder). In Stage 2, they use reinforcement learning (RL) to refine both latent and text generation.

We identify a limitation at each stage of this paradigm. First, in the SFT stage, the latent target is obtained by encoding the helper image with an off-the-shelf vision encoder. Because this encoder is trained for general-purpose visual representation rather than the downstream reasoning objective, its features may preserve perceptual content that is irrelevant to the task while failing to emphasize the intermediate evidence needed for reasoning. The latent generator is therefore supervised toward a convenient but suboptimal target, limiting how much useful reasoning information can be learned even when the downstream model is trainable. Second, existing RL methods either update only text tokens or apply deterministic regularization to the latent block. Text-only objectives provide no direct likelihood or credit assignment for latent actions, whereas deterministic regularization only constrains how far a latent representation drifts from a reference. Without sampling alternative latent actions for exploration, reward optimization cannot discover more useful latent reasoning trajectories, yielding small or unstable gains as shown in Figure 2.

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.

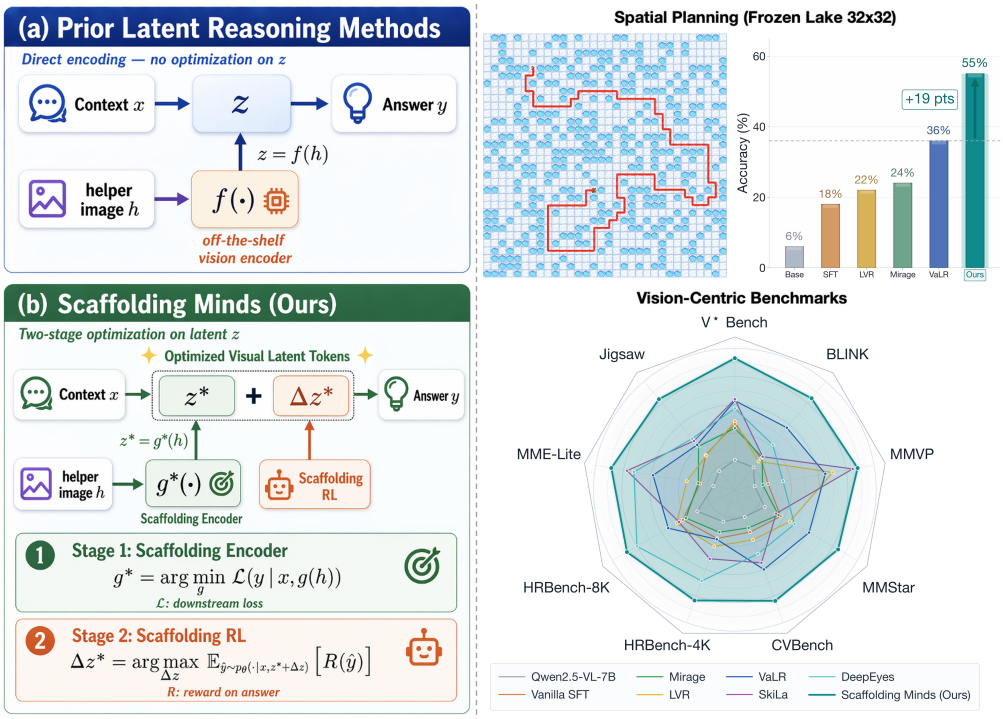


Figure 1: **Scaffolding Minds. Left:** Prior latent-reasoning methods derive latent targets by encoding a training-time helper image with a frozen off-the-shelf vision encoder. Our method instead learns a dedicated scaffolding encoder that optimizes the latent target for downstream reasoning, then uses a learned Gaussian policy to sample reward-guided residual latent actions for direct exploration. **Right:** Scaffolding Minds improves the strongest prior latent baseline by $+9.5$ points on average across FrozenLake levels, with the gain widening to $+19$ points on the hardest 32×32 grids, and by $+5.6$ points on average across nine visual-centric reasoning benchmarks.

In this paper, we present **Scaffolding Minds** (Figure 1, left) to address these two issues. First, for the SFT stage, instead of using fixed off-the-shelf features as the target, we *learn* a *scaffolding encoder* to encode a helper image into a latent target optimized for the reasoning task.

Second, we introduce *Scaffolding RL*, which replaces deterministic latent regularization with an explicit Gaussian sampler over residual latent actions. Two lightweight heads learn its input-adaptive mean and variance, and the policy samples these actions during rollout to explore alternative latent reasoning trajectories (Figure 2). Moreover, we find the scaffolding encoder and scaffolding RL are complementary, yielding performance boosts when combined.

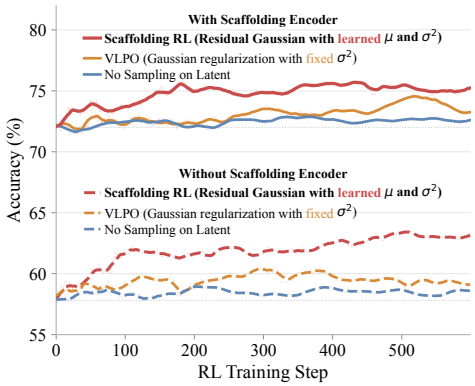


Figure 2: RL performance comparison on the FrozenLake benchmark (average accuracy over Levels 8, 16, 24, and 32).

We evaluate *Scaffolding Minds* on FrozenLake spatial planning (across grid sizes from 8×8 to 32×32) and nine visual-centric reasoning benchmarks: V* (Wu & Xie, 2024), BLINK (Fu et al., 2024), MMVP (Tong et al., 2024b), MMStar (Chen et al., 2024a), CVBench (Tong et al., 2024a), HRBench-4K and HRBench-8K (Wang et al., 2024a), MME-RealWorld-Lite (Zhang et al., 2025c), and Jigsaw (Lyu et al., 2025). Across all suites, our method consistently outperforms the strongest prior latent baseline. On FrozenLake, it raises average accuracy by $+9.5$ points, with a gain of $+19$ points on the most challenging 32×32 grids. Across the nine visual-centric reasoning benchmarks, we achieve an average accuracy increase of $+5.6$ points.

2 PRELIMINARY: LATENT REASONING FOR VLMS

Latent visual reasoning (Yang et al., 2025b; Li et al., 2025b; Wang et al., 2026; Jeon et al., 2026) inserts a block of dedicated latent tokens $\mathbf{z}$ between $\langle\text{BOT}\rangle$ and $\langle\text{EOT}\rangle$ inside the VLM's chain-of-thought, with their target set to the vision embedding of a helper image $\mathbf{h}$ produced by an off-the-shelf vision encoder f. The helper image $\mathbf{h}$ is only available at training time. The reasoning sequence takes the form $[\mathbf{x}, \mathbf{c}^{(1)}, \langle\text{BOT}\rangle, \mathbf{z}^{(1)}, \langle\text{EOT}\rangle, \ldots, \mathbf{c}^{(B)}, \langle\text{BOT}\rangle, \mathbf{z}^{(B)}, \langle\text{EOT}\rangle, \mathbf{c}^{(B+1)}, \mathbf{y}]$, where $\mathbf{x}$ is the input image and query, each $\mathbf{c}^{(b)}$ is optional text reasoning, each $\mathbf{z}^{(b)} \in \mathbb{R}^{K\times d}$ holds K latent tokens, $B \geq 1$ is the number of latent blocks, and $\mathbf{y}$ is the answer. We write $\mathbf{z}$ when referring to a single block. As in prior work (Wang et al., 2026), the VLM itself decides when to enter latent reasoning by emitting $\langle\text{BOT}\rangle$, and it returns to text after the block, so a reasoning chain may contain a single latent block or several blocks interleaved with text. Existing pipelines train the VLM in two stages: an *SFT stage* and an *RL stage*.

SFT stage. The goal of the SFT stage is to teach the VLM to generate the latent block $\mathbf{z}$ from the input image and text query alone (without the helper image), so that no helper image is needed at inference. Essentially, a frozen off-the-shelf vision encoder f maps $\mathbf{h}$ to a target latent block, and the VLM is trained to match its generated latent tokens to this target while jointly fitting the answer $\mathbf{y}$ under next-token prediction.

RL stage. The SFT-stage model is further refined under a task reward (Figure 2). Existing latent-reasoning RL methods handle the text and latent components differently.

Text-centric policy optimization. Standard recipes apply GRPO (Shao et al., 2024b) to the language-modeling head, restricting policy optimization to the discrete text trajectory (Li et al., 2025b). Given input $\mathbf{x}$, the VLM samples G text trajectories and optimizes $\mathcal{L}_{\text{text-RL}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\min\big(\rho_{i,t}(\theta)A_i,\ \text{clip}(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon)A_i\big)$, where $\rho_{i,t}(\theta)$ is the current-to-old probability ratio for text token t, A_i is the group-relative advantage, and ϵ is the clipping threshold. Because this objective defines policy probabilities only over discrete text tokens, it neither samples nor directly explores alternative actions in the continuous latent block.

Visual-Latent Policy Optimization (VLPO) (Wang et al., 2026). VLPO additionally incorporates deterministic latent embeddings through a fixed-variance Gaussian formulation. For an old-policy rollout latent $\mathbf{z}_{i,t}^{\text{old}}$ and the current-policy deterministic output $\mathbf{z}_{i,t}^{\theta}$, its latent factor is $r_{i,t}(\theta) = \exp\big(-\|\mathbf{z}_{i,t}^{\text{old}} - \mathbf{z}_{i,t}^{\theta}\|_2^2/(2\sigma_0^2)\big)$. Here, the current-policy output is the Gaussian mean, the old-policy rollout latent is the point being evaluated, and the variance σ_0^2 is fixed. Thus, the Gaussian acts as latent regularization that penalizes drift between deterministic latent outputs. It is not a rollout distribution: VLPO does not sample a latent action from this Gaussian and thus creates no alternative latent trajectories for exploration.

3 METHOD: SCAFFOLDING MINDS

This section first identifies two limitations of existing latent visual reasoning pipelines, then presents **Scaffolding Minds**, our two-stage framework that addresses both.

3.1 MOTIVATION

Existing latent visual reasoning pipelines suffer from two limitations: (1) at the SFT stage, the latent target is produced by a *frozen* off-the-shelf vision encoder f, which is not optimized to capture reasoning-sensitive features from the helper image $\mathbf{h}$; and (2) at the RL stage, existing methods either leave the latent block outside RL or use a deterministic, fixed-variance Gaussian formulation, without sampling latent actions for exploration.

Both limitations share a common root cause: the latent target is not optimized for the reasoning task, and RL does not directly explore alternative latent trajectories. To fill this gap, we propose **Scaffolding Minds** (Figure 3): a *scaffolding encoder* that learns the latent target end-to-end through the downstream task loss, and *Scaffolding RL* that samples residual latent actions from an input-adaptive Gaussian whose mean and variance are conditioned on the generated latent.

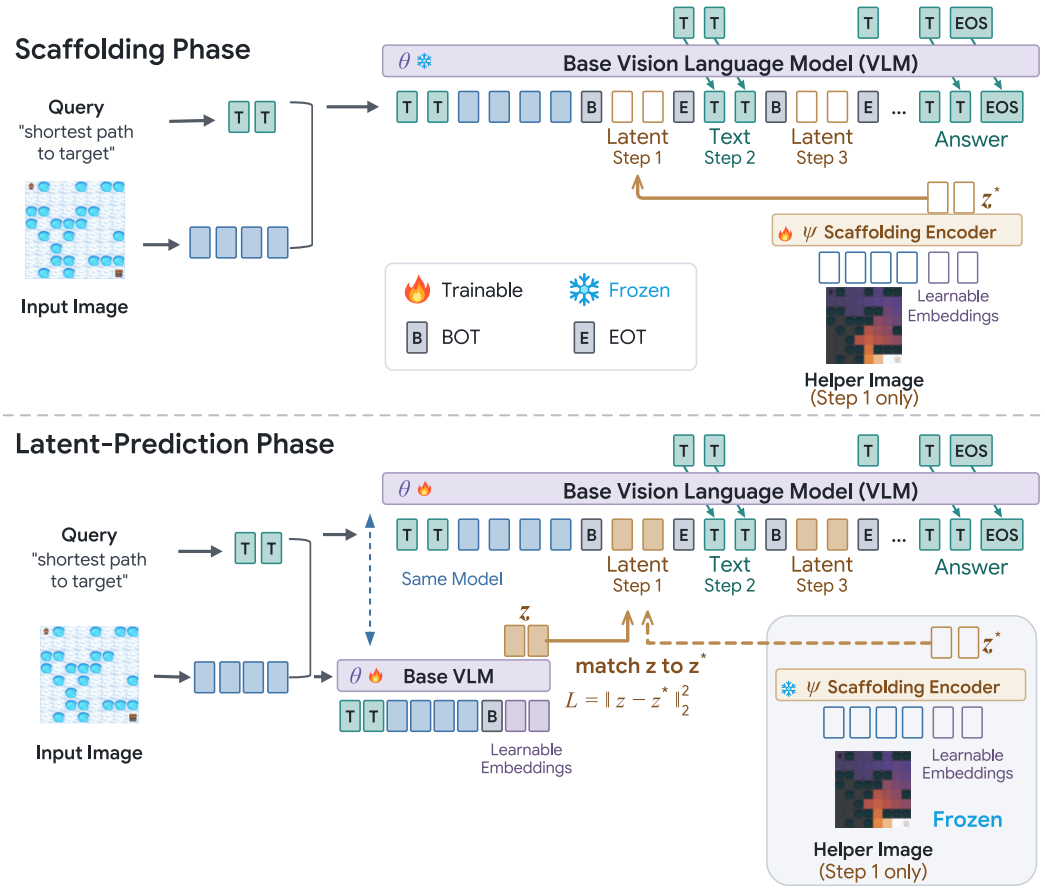


Figure 3: **SFT stage of Scaffolding Minds. Scaffolding Phase** (top): the scaffolding encoder ψ is trained to compress the helper image into target latent tokens $\mathbf{z}^*$ that drive the frozen base VLM to produce the correct answer. **Latent-Prediction Phase** (bottom): the VLM (θ) is then trained to predict latent tokens $\mathbf{z}$ from the input $\mathbf{x}$ alone, supervised by ℓ_2 matching $\mathcal{L} = \|\mathbf{z} - \mathbf{z}^*\|_2^2$ to the (now frozen) scaffolding encoder's targets, while the helper image is no longer used at inference.

3.2 Stage 1: SFT via Scaffolding Encoder

The SFT stage is to learn an optimized latent target representation $\mathbf{z}^* \in \mathbb{R}^{K\times d}$ from the helper image $\mathbf{h}$, and then use $\mathbf{z}^*$ as supervision to teach the VLM to produce its own latent block $\mathbf{z}$ from the input $\mathbf{x}$ alone, so that no helper image is needed at inference. As shown in Figure 3, we split this into two phases: (i) the *Scaffolding Phase* trains a dedicated scaffolding encoder with model parameters ψ to map $\mathbf{h}$ to the target $\mathbf{z}^*$, optimized end-to-end through the frozen base VLM's task loss; and (ii) the *Latent-Prediction Phase* fine-tunes the VLM to regress its own latent block $\mathbf{z}$ toward the (now frozen) target $\mathbf{z}^*$, while jointly fitting the answer $\mathbf{y}$ under next-token prediction. After Stage 1, the scaffolding encoder ψ and helper image $\mathbf{h}$ are both discarded; only the VLM is used at inference.

Scaffolding Phase. As illustrated in Figure 1, our key intuition is to replace the frozen off-the-shelf encoder f used by prior latent reasoning methods — which simply set the latent target to $f(\mathbf{h})$ — with a *learned* function $g^* : \mathbf{h} \mapsto \mathbf{z}^*$ trained directly through the downstream task, so that the resulting $\mathbf{z}^*$ is *optimized for reasoning* rather than tied to a generic vision feature space.

We represent the optimized scaffolding encoder as g^*_ψ. It is a trainable copy of the VLM's vision encoder followed by a cross-attention pooling module (Figure 3, top). It encodes the helper image $\mathbf{h}$ into K latent tokens $\mathbf{z}^* = g^*_\psi(\mathbf{h}) \in \mathbb{R}^{K\times d}$, which are directly inserted at the latent positions of the VLM, after the input $\mathbf{x}$, and the VLM produces the answer $\mathbf{y}$. Given the training input $\mathbf{x}$ (image and text query) and the teacher-forced answer $\mathbf{y}$, we *freeze* the base VLM and update only ψ end-to-end through the standard token-level cross-entropy loss $\mathcal{L}_{\text{CE}}$ on the answer tokens:

$$\mathcal{L}_{\text{scaffolding}}(\psi) \;=\; \mathcal{L}_{\text{CE}}\big(\mathbf{y} \,\big|\, \mathbf{x},\, \mathbf{z}^*\big)\,. \tag{1}$$

In this way, the scaffolding encoder learns to produce a $\mathbf{z}^*$ that is trained to capture the reasoning-sensitive features the frozen VLM needs to answer correctly. A trace may contain several helper images, one for each latent block. To encode all of them in one forward pass, we pack them into a single sequence in which each helper image is followed by its K learnable embeddings, $[\mathbf{h}^{(1)}, \mathbf{e}^{(1)}, \mathbf{h}^{(2)}, \mathbf{e}^{(2)}, \ldots]$, and apply an attention mask under which each group of embeddings sees only the helper image right before it. The mask keeps the target blocks independent of one another, so they can share a single forward pass (Appendix D.1).

Latent-Prediction Phase. Since the helper image $\mathbf{h}$ is unavailable at inference, the VLM is trained to predict its own latent block $\mathbf{z}$ that approaches the optimized target $\mathbf{z}^*$ from the input $\mathbf{x}$ alone (Figure 3, bottom). Concretely, we freeze the scaffolding encoder g^*_ψ and fine-tune the base VLM with parameters θ. The latent tokens $\mathbf{z}$ are not generated autoregressively; instead, they are generated from K learnable embeddings that pass through the VLM transformer using the input $\mathbf{x}$ as context. This design speeds up inference by producing all K tokens of a block in a single forward pass. During training, we use teacher forcing: each $\langle\text{BOT}\rangle$ is followed by the K learnable embeddings and then by the target block $\mathbf{z}^*$, giving the sequence $[\mathbf{x}, \langle\text{BOT}\rangle, \mathbf{e}, \mathbf{z}^*, \langle\text{EOT}\rangle, \mathbf{c}, \ldots, \mathbf{y}]$. An attention mask lets the embeddings of a block attend to each other bidirectionally and see all preceding tokens except the embeddings of earlier blocks, and hides them from every other token. Later tokens therefore condition on $\mathbf{z}^*$, and all latent blocks of a trace are predicted in a single forward pass, the *predictor pass* (Appendix D.1). Since the predictor and the VLM are the same model, a second forward pass then generates the answer text from the latent blocks (Figure 3, bottom). We train the base VLM θ with the following objective:

$$\mathcal{L}_{\text{predict}}(\theta) = \lambda_{\text{latent}} \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{z}_k - \mathbf{z}_k^*\|_2^2 + \lambda_{\text{task}} \mathcal{L}_{\text{CE}}\big(\mathbf{y} \,\big|\, \mathbf{x}, \mathbf{z}^*\big), \tag{2}$$

where λ_{latent} and λ_{task} balance the two terms. The first term pulls $\mathbf{z}$ toward the optimized target $\mathbf{z}^*$ from the Scaffolding Phase; the second term trains the VLM to generate the answer from the teacher-forced target $\mathbf{z}^*$, which the predicted $\mathbf{z}$ replaces at inference. With several latent blocks, the first term is averaged over all blocks, and the $\langle\text{BOT}\rangle$ tokens are included in the cross-entropy loss. At inference, the model decodes text until it emits $\langle\text{BOT}\rangle$, predicts one latent block in place of $\mathbf{z}^*$, and resumes decoding.

3.3 Stage 2: Scaffolding RL

Prior latent-reasoning RL methods either optimize only the text trajectory or use a deterministic latent regularizer. Neither choice samples alternative latent actions, so neither directly explores latent reasoning trajectories. Scaffolding RL instead defines an explicit, input-adaptive policy over residual latent actions and samples from it during rollout, as illustrated in Figure 4.

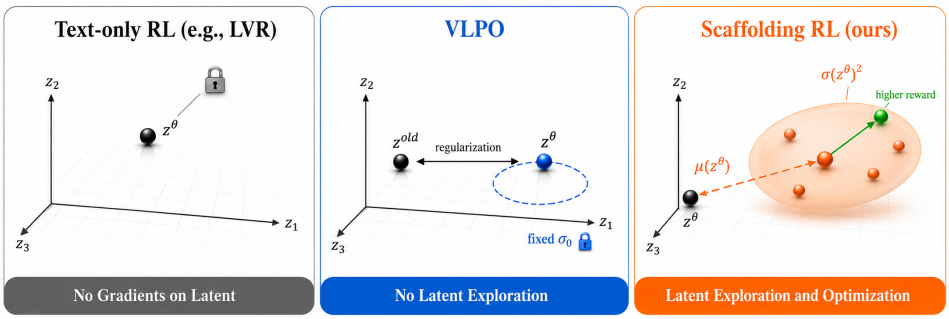


Figure 4: **Three Stage 2 RL paradigms in latent space. Left:** text-only RL provides no direct gradients on latent actions. **Middle:** VLPO applies fixed-variance Gaussian regularization to deterministic latent outputs but does not sample latent actions. **Right:** Scaffolding RL samples residual latent actions from a learned distribution and uses reward to optimize the latent tokens with exploration.

To solve this, we introduce **Scaffolding RL**. We treat the generated latent prior $\mathbf{z}^\theta = f_\theta(\mathbf{x}, \mathbf{c}_{\text{pre}})$ as a *base action* and learn adaptive, reward-guided adjustments $\mathbf{\Delta}_z^*$. Here, $\mathbf{c}_{\text{pre}}$ denotes the optional text reasoning generated before the latent block. Crucially, the mean and variance of the Gaussian

Table 1: **FrozenLake results (accuracy %)** on four representative levels; Avg. is their mean. Best in **bold**, second best underlined. Methods marked ◇ use image-generation architectures. △: gain of Scaffolding Encoder+RL over SFT.

Method	Lv. 8 (8×8)	Lv. 16 (16×16)	Lv. 24 (24×24)	Lv. 32 (32×32)	Avg.
Base model & supervised fine-tuning					
SFT	84.0	65.0	41.0	18.0	52.0
SFT+GRPO	85.0	67.0	43.0	20.0	53.8
Image-generation methods					
VPRL◇ (Xu et al., 2025)	88.0	62.0	48.0	35.0	58.3
DiffThinker◇ (He et al., 2025)	92.0	68.0	53.0	44.0	64.3
Latent reasoning methods					
LVR (Li et al., 2025b)	86.0	69.0	45.0	22.0	55.5
Mirage (Yang et al., 2025b)	87.0	71.0	47.0	24.0	57.3
CoVT (Qin et al., 2025)	88.0	73.0	50.0	27.0	59.5
Monet (Wang et al., 2026)	89.0	75.0	53.0	31.0	62.0
VaLR (Jeon et al., 2026)	91.0	78.0	57.0	36.0	65.5
Scaffolding Encoder	94.0	79.0	63.0	52.0	72.0
Scaffolding Encoder+RL	**96.0**	**82.0**	**67.0**	**55.0**	**75.0**
△ vs. SFT	**+12.0**	**+17.0**	**+26.0**	**+37.0**	**+23.0**

sampler are not fixed, but are learned as functions of the base action $\mathbf{z}^\theta$. We predict them using two MLP heads on top of the shared VLM hidden states:

$$\mathbf{\Delta}_z \sim \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{z}^\theta),\, \boldsymbol{\sigma}(\mathbf{z}^\theta)^2\big)\,, \tag{3}$$

yielding the perturbed latent block $\mathbf{z}^\theta + \mathbf{\Delta}_z$ that replaces $\mathbf{z}^\theta$ in the forward pass. Because $\boldsymbol{\mu}(\mathbf{z}^\theta)$ and $\boldsymbol{\sigma}(\mathbf{z}^\theta)$ are conditioned on the generated latent prior, both the center and scale of exploration can vary across inputs. To ensure stable initial exploration, the mean head is zero-initialized so that the initial distribution is centered on the generated latent prior, while the variance head begins at small positive values. The sampled latent block is treated as one more action of the policy: its probability ratio enters the GRPO objective (Shao et al., 2024b) alongside those of the text tokens, so the latent block and the text are updated together from the same reward (Appendix C). This dynamic sampling complements the Stage 1 scaffolding encoder and yields the gains reported in our experiments.

4 Experiments

We evaluate Scaffolding Minds in two complementary settings: **spatial planning** (Section 4.1), requiring multi-step reasoning in structured visual environments, and **visual-centric reasoning** (Section 4.2), testing fine-grained evidence extraction and reasoning on real-world benchmarks.

Implementation. We use Qwen2.5-VL (Bai et al., 2025) as the shared VLM backbone. In Stage 1 (Scaffolding Encoder), we first train the scaffolding encoder—which reuses the VLM vision encoder and compresses the helper image into $K{=}4$ latent tokens—while keeping the remaining VLM components frozen, and then freeze the scaffolding encoder and train the shared VLM to generate the target latent block from the standard input alone. In Stage 2 (Scaffolding RL), we apply reward-guided adjustments on top of the generated latent prior; the scaffolding encoder is no longer used; at inference, the VLM first generates the latent prior, and the two MLP heads then read its last hidden states to produce $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, from which the adjustment is sampled.

Baselines. We compare against three families of methods: (i) supervised baselines without latent tokens, *SFT* and *SFT+GRPO*; (ii) latent visual reasoning methods that derive their targets from off-the-shelf vision features — *LVR* (Li et al., 2025b), *Mirage* (Yang et al., 2025b), *CoVT* (Qin et al., 2025), *Monet* (Wang et al., 2026), and *VaLR* (Jeon et al., 2026); and (iii) image-generation methods that produce explicit intermediate images at inference time, *VPRL* (Xu et al., 2025) and *DiffThinker* (He et al., 2025). On the visual-centric reasoning benchmarks we additionally compare against the thinking-with-images systems *DeepEyes* (Zheng et al., 2025) and *Thyme* (Zhang et al., 2025b), and the latent baseline *SkiLa* (Tong et al., 2025).

4.1 SPATIAL PLANNING: FROZENLAKE

FrozenLake (Wu et al., 2024) presents grid-based environments with obstacles ("holes") requiring multi-step spatial planning: the agent outputs the complete path as directional moves (Up/Down/Left/Right), receiving zero reward for paths through holes. We train a single model on all even levels from 8 to 32, spanning 8×8 to 32×32 grids, with 4,550 training examples in total. All grid images are padded to a unified 32×32 resolution before being encoded. We evaluate on 100 unseen test layouts for each level from 8 to 32, including the odd levels not seen in training, and report four representative levels (8, 16, 24, 32). All baselines are trained and evaluated by us in the same setup; training and evaluation splits are detailed in the appendix.

Main results. Table 1 and Figure 5 show a consistent pattern across maze complexity. First, compared with base methods, our full model improves average accuracy by 23.0 points over the SFT baseline, indicating that latent visual reasoning remains valuable even on top of a strong base. Second, compared with latent reasoning baselines, our method gains 9.5 points over the strongest prior approach. This gap is important because the latent baselines are trained with the same intermediate helper images; the advantage therefore comes from learning a better latent target space rather than from access to extra information. Our latent representation is optimized directly through downstream task loss, which makes it more suitable for decision-relevant reasoning than targets inherited from general-purpose vision features. Third, compared with image-generation methods, our method improves by 10.7 points. This suggests that explicitly generating intermediate images is not necessary to obtain strong visual reasoning, as long as the latent space itself captures the right reasoning structure. Figure 5 further shows that this advantage grows with difficulty: our method degrades more gracefully as the maze becomes larger, reaching a +19-point gap over the strongest latent baseline at Level 32.

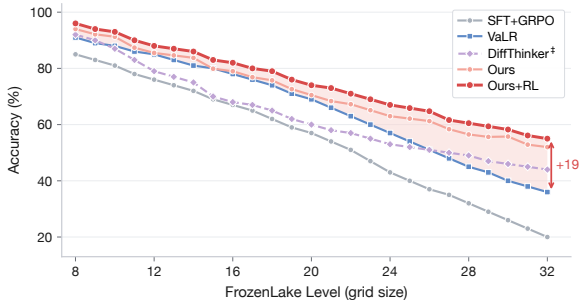


Figure 5: **Per-level FrozenLake accuracy.** Scaffolding Minds (red) degrades gracefully as grid size grows, while baselines collapse faster. The shaded band highlights the steadily widening advantage of Ours + RL over the strongest latent baseline (VaLR), reaching **+19** points at Level 32.

4.2 VISUAL-CENTRIC REASONING BENCHMARKS

To show that Scaffolding Minds generalizes beyond spatial planning and is not tied to a task-specific helper image such as the FrozenLake heatmap, we evaluate on nine established visual-centric reasoning benchmarks: V* (Wu & Xie, 2024) (fine-grained attribute/spatial reasoning), BLINK (Fu et al., 2024) (core perception), MMVP (Tong et al., 2024b) (visual shortcomings), MMStar (Chen et al., 2024a) (vision-indispensable), CVBench (Tong et al., 2024a) (2D/3D understanding), HRBench-4K and HRBench-8K (Wang et al., 2024a) (high-resolution), MME-RealWorld-Lite (Zhang et al., 2025c) (real-world scenarios), and Jigsaw (Lyu et al., 2025) (spatial reasoning). We train on Zebra-CoT (Li et al., 2025a), using 162K interleaved traces for Stage 1 and a disjoint 20K split for Stage 2. Each trace alternates text with a variable number of dataset-provided helper images, such as crops, bounding boxes, diagrams, and candidate placements. These helpers provide intermediate visual evidence rather than the solution, and each supervises one latent block, so the model learns to produce multiple interleaved blocks. Almost all key settings follow FrozenLake (Appendices D.2 and E.2).

Results. Table 2 and Figure 1 (right) show that the same latent-learning strategy transfers beyond FrozenLake. First, compared with the supervised fine-tuning baseline, our method improves the average score by 10.3 points, showing that the learned latent space remains useful even on visual-centric reasoning benchmarks without explicit planning outputs. Second, compared with prior latent reasoning methods, our method improves the strongest prior average by 5.6 points. The improvement is consistent across benchmarks that require spatial discrimination and fine-grained visual comparison, where our method also gains the most over the base model, e.g., $+20.7$ points on MMVP, $+17.6$ points on MME-RealWorld-Lite, and $+14.4$ points on V*. Third, compared with prior thinking-with-images style methods, our method improves the strongest prior average by 6.2 points, indicating that strong visual-reasoning gains do not require manipulating or generating additional images at inference time when the latent representation is already optimized for the downstream task.

Table 2: **Visual-centric reasoning benchmark results (accuracy %). Bold**: best; underlined: second best. †: reproduced via original checkpoint/code; ‡: our re-implementation. §: retrained by us under our setup. Unmarked baseline cells are taken from the original papers. CVB.=CVBench, HRB=HRBench, MME=MME-RealWorld-Lite, Jig.=Jigsaw.

Method	V*	BLINK	MMVP	MMStar	CVB.	HRB-4K	HRB-8K	MME	Jig.	Avg.
Closed-source models										
GPT-4o	66.0	60.0	70.7	61.6	80.1	59.0	55.5	52.0	53.0	62.0
GPT-4V	58.0	58.3	51.0	56.0	69.5	56.5	52.0	45.0	47.0	54.8
GPT-4o-mini	57.0	53.6	56.0	54.8	68.0	56.0	51.0	37.4	46.0	53.3
Claude 3.7-Sonnet	72.0	56.6	64.0	65.1	80.0	68.0	64.0	49.0	52.0	63.4
Base model & supervised fine-tuning										
Qwen2.5-VL-7B	76.2	55.7	56.0	63.9	74.5	68.6	64.9	39.7	50.4	61.1
Vanilla SFT	81.3	57.2	58.7	66.0	77.0	70.5	66.8	42.0	52.5	63.6
Tool-calling & think-with-image methods										
DeepEyes§ (Zheng et al., 2025)	83.5	59.5	61.7	67.8	80.6	75.3	72.8	53.4	54.6	67.7
Thyme (Zhang et al., 2025b)	81.9	56.1	62.4†	65.9	79.8†	77.0	72.0	55.2	54.8†	67.2
Latent reasoning methods										
Mirage (Yang et al., 2025b)	80.7‡	57.4‡	57.6‡	65.7‡	76.3‡	69.8‡	66.4‡	42.3‡	53.6‡	63.3
CoVT (Qin et al., 2025)	78.0	56.0	58.7	68.9†	80.0	72.9	69.4	51.4†	52.7†	65.3
Monet (Wang et al., 2026)	83.2	57.7†	60.3†	68.3†	79.7†	71.0	68.0	55.5	54.6†	66.5
LVR (Li et al., 2025b)	81.7	56.7†	71.7	67.2†	78.3†	71.4†	67.6†	44.3†	52.4†	65.7
VaLR§ (Jeon et al., 2026)	84.6	62.3	70.2	69.7	83.2	70.6	68.7	50.1	53.8	68.1
SkiLa§ (Tong et al., 2025)	84.8	57.5	75.7	65.4	82.1	72.8	67.3	54.6	54.3	68.3
Scaffolding Encoder	87.4	64.8	73.3	72.1	86.4	73.6	71.1	53.4	56.0	70.9
Scaffolding Encoder+RL	**90.6**	**67.2**	**76.7**	**73.5**	**88.4**	**77.5**	**74.1**	**57.3**	**59.7**	**73.9**
Δ vs. Qwen2.5-VL-7B	**+14.4**	**+11.5**	**+20.7**	**+9.6**	**+13.9**	**+8.9**	**+9.2**	**+17.6**	**+9.3**	**+12.8**

4.3 Ablation on Key Contributions

SFT visual target ablation. To identify which Stage 1 design choice drives the gain over prior latent reasoning methods, we compare four Stage 1 configurations under the same Qwen2.5-VL-7B backbone with no Stage 2 RL applied: *SFT* without latent tokens as a control; *Frozen target*, the prior-work objective that L2-regresses latent tokens onto frozen off-the-shelf vision-encoder features (Li et al., 2025b), mean-pooled to K tokens so that the target has no trainable parameters; and two *optimized targets*, trained through the downstream task loss, that differ only in how much of the target encoder is trained: the attention-pooling module alone (∼80M parameters; *Optimized target, frozen VE*) or also the vision encoder (∼0.76B in total; *Optimized target, tuned VE*, ours). Attention pooling is simply the smallest trainable module through which a target can be optimized. As shown in Figure 6, optimizing the target lifts accuracy by +4.3 points even when only the pooling module is trained, and letting the task loss also shape the vision encoder adds a further +10.0 points; together they account for the full +14.3-point gap over the frozen target used by prior work. Added capacity alone does not explain this: tuning the same vision encoder in plain SFT gains only +4.5 points (Appendix F.2). *These results suggest that optimizing the latent target, rather than regressing onto frozen features, drives the Stage 1 gain.*

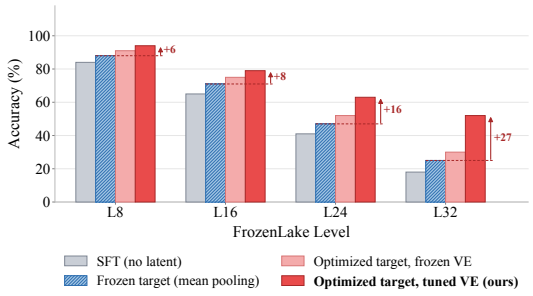


Figure 6: Stage 1 visual target ablation on FrozenLake.

RL method comparison. Table 3 compares Stage 2 recipes from the same Stage 1 checkpoint. Text-only GRPO leaves the latent block untouched and adds only +0.6 points. VLPO adds +0.9 points, but its training fluctuates (Figure 2) because its fixed-variance Gaussian regularizes the latent block without sampling alternative latent rollouts. Scaffolding RL gives the largest gain (+3.0 points), and its advantage widens on the frozen-target checkpoint (+5.2 vs. +1.2 points for VLPO), where the weaker latent leaves more room for latent exploration. Neither head suffices alone: a learned mean or a learned variance gains only +1.6 or +1.2 points, since the mean sets the direction of

Table 3: **Stage 2 RL ablation on FrozenLake.** We report the mean and standard deviation of accuracy over 3 seeds, averaged over the four representative Levels 8, 16, 24, and 32.

Stage 2 recipe	Latent sampling Mean	Latent sampling Variance	Stage 1 target Scaffolding enc.	Stage 1 target Frozen target
Stage 1 checkpoint (no RL)	–		*72.0*$\pm$0.6	*57.8*$\pm$0.6
Text-only GRPO	no latent sampling		72.6$\pm$0.5 (+0.6)	58.4$\pm$0.5 (+0.6)
VLPO (Wang et al., 2026)	latent regularization		72.9$\pm$0.6 (+0.9)	59.0$\pm$0.6 (+1.2)
Learned mean only	learned	fixed	73.6$\pm$0.5 (+1.6)	60.4$\pm$0.7 (+2.6)
Learned variance only	zero	learned	73.2$\pm$0.6 (+1.2)	59.8$\pm$0.5 (+2.0)
Scaffolding RL (ours)	**learned**	**learned**	**75.0**$\pm$0.7 (+**3.0**)	**63.0**$\pm$0.6 (+**5.2**)

exploration and the variance sets its width. *Sampling learned adjustments on top of the Stage 1 prior is essential for converting reward into reliable latent-space exploration.*

5 Related Work

Think about Image. This line of work treats the image as a static precondition and reasons *about* it purely in language space, extending textual chain-of-thought (Wei et al., 2022; Kojima et al., 2022) to vision-language models such as Qwen2.5-VL (Bai et al., 2025), LLaVA (Liu et al., 2023), and Gemini (Gemini Team, 2023) by verbalizing the visual evidence into a natural-language reasoning chain (Bavishi et al., 2023; Xu et al., 2024; Zhang et al., 2024; Dong et al., 2025; Mitra et al., 2024; Mondal et al., 2024; Shao et al., 2024a; Cheng et al., 2025; Wei et al., 2024; Du et al., 2025). *Unlike these text-only CoTs, which tend to lose fine-grained spatial and perceptual evidence, our method carries such evidence forward as continuous latent visual tokens in the reasoning chain.* A second line of work, *thinking with images*, instead manipulates or generates intermediate images during reasoning; we discuss it in Appendix B.

Latent Reasoning. A more recent line replaces explicit intermediate images with a compact block of continuous latent tokens inserted into the reasoning chain, building on observations that explicit chain-of-thought has its own limitations (Pinker, 1994; Silver et al., 2016; Chen et al., 2024b). On the text side, related work has explored pause and planning tokens (Goyal et al., 2024; Wang et al., 2024b), compressed CoT (Cheng & Van Durme, 2024; Shen et al., 2025), continuous reasoning spaces (Hao et al., 2024; Liu et al., 2024), latent internalization of CoT (Deng et al., 2023; 2024; Zelikman et al., 2022), latent diffusion (Kang et al., 2025; 2026), and looped architectures (Saunshi et al., 2025; Teoh et al., 2025). In the multimodal setting, prior methods derive the latent target from a frozen off-the-shelf vision encoder (e.g., SigLIP (Tschannen et al., 2025)) applied to a helper image, through variants including L2 feature regression, vision-aligned distillation, policy-optimized embeddings, and sketchpad-style decoders (Yang et al., 2025b; Wang et al., 2026; Qin et al., 2025; Li et al., 2025b; Tong et al., 2025; Zhang et al., 2025a; Chen et al., 2025; Jeon et al., 2026; Dong et al., 2026). Methodologically, this family is closely related to joint-embedding predictive architectures (LeCun, 2022) and learning-using-privileged-information (Vapnik & Vashist, 2009), and conceptually echoes the idea of fading instructional scaffolds in cognitive science (Vygotsky, 1978; Wood et al., 1976). *In contrast to prior multimodal latent methods that inherit a fixed general-purpose target from a frozen vision encoder, our method trains a scaffolding encoder end-to-end through the downstream task loss — turning the latent target itself into an optimized component of the system — and pairs it with an input-adaptive RL sampler whose mean and variance are learned functions of the latent prior.*

6 Conclusion

We present Scaffolding Minds, a two-stage framework that improves both how latent visual reasoning is supervised and how it is refined through RL. Stage 1 learns a scaffolding encoder whose latent target is optimized end-to-end for the reasoning task, in place of the fixed vision-encoder target used by prior work. Stage 2 learns a Gaussian latent policy that samples residual actions around the predicted latent, enabling reward-driven exploration directly in latent space. Together, the two stages improve the strongest prior latent baseline by $+9.5$ points on average on FrozenLake ($+19$ points at 32×32) and by $+5.6$ points on average across nine visual-centric reasoning benchmarks, showing that both the quality of the latent target and direct latent-space exploration matter.

AI USE STATEMENT

Generative AI tools were used only for writing assistance: drafting and editing the manuscript to improve grammar, wording, clarity, and readability. They were not used to generate synthetic datasets; develop theoretical models or conceptual frameworks; formulate mathematical claims or proofs; propose or refine hypotheses; design or provide feedback on the research methodology or experiments; implement the method; translate content; clean or reformat data; analyze data; or interpret results. The authors reviewed and verified all AI-assisted writing and take full responsibility for the final content of this work, including all text and claims.

REPRODUCIBILITY STATEMENT

Section 3 describes the method, Appendix C gives the full Scaffolding RL objective, and Appendix D.1 specifies the attention masks used in Stage 1. Appendix D lists the data, backbone, optimization hyperparameters, compute, and evaluation protocol for FrozenLake and for the visual-centric benchmarks, and Appendix E gives the code that generates the FrozenLake helper images. All experiments use publicly available models, data, and benchmarks: the Qwen2.5-VL backbone (Bai et al., 2025), the FrozenLake suite of VSP (Wu et al., 2024), the Zebra-CoT dataset (Li et al., 2025a), and the nine public benchmarks of Section 4.2. We will release our code and trained checkpoints upon publication.

REFERENCES

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Rohan Bavishi, Erich Elsen, Curtis Hawthorne, Maxwell Nye, Augustus Odena, Arushi Somani, and Sağnak Taşırlar. Fuyu-8b: A multimodal architecture for ai agents. https://www.adept.ai/blog/fuyu-8b, 2023.

Chao Chen, Zhixin Ma, Yongqi Li, Yupeng Hu, Yinwei Wei, Wenjie Li, and Liqiang Nie. Reasoning in the dark: Interleaved vision-text reasoning in latent space. *arXiv preprint arXiv:2510.12603*, 2025.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a.

Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Andersen. MELODI: Exploring memory compression for long contexts. *arXiv preprint arXiv:2410.03156*, 2024b.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

Zihui Cheng, Qiguang Chen, Xiao Xu, Jiaqi Wang, Weiyun Wang, Hao Fei, Yidong Wang, Alex Jinpeng Wang, Zhi Chen, Wanxiang Che, and Libo Qin. Visual thoughts: A unified perspective of understanding multimodal chain-of-thought. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024.

Shuai Dong, Siyuan Wang, Xingyu Liu, Chenglin Li, Haowen Hou, and Zhongyu Wei. Interleaved latent visual reasoning with selective perceptual modeling. In *Proceedings of the 64th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 29316–29335, 2026.

Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkang Yang, Winston Hu, Yongming Rao, and Ziwei Liu. Insight-V: Exploring long-chain visual reasoning with multimodal large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9062–9072, 2025.

Yifan Du, Kun Zhou, Yingqian Min, Yue Ling, Wayne Xin Zhao, and Youbin Wu. Revisiting the necessity of lengthy chain-of-thought in vision-centric reasoning generalization. *arXiv preprint arXiv:2511.22586*, 2025.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision (ECCV)*, 2024.

Gemini Team. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiawei Gu, Yunzhuo Hao, Huichen Will Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. ThinkMorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. *arXiv preprint arXiv:2510.27492*, 2025.

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. *arXiv preprint arXiv:2412.06769*, 2024.

Zefeng He, Xiaoye Qu, Yafu Li, Tong Zhu, Siyuan Huang, and Yu Cheng. DiffThinker: Towards generative multimodal reasoning with diffusion models. *arXiv preprint arXiv:2512.24165*, 2025.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Byungwoo Jeon, Yoonwoo Jeong, Hyunseok Lee, Minsu Cho, and Jinwoo Shin. Vision-aligned latent reasoning for multi-modal large language model. *arXiv preprint arXiv:2602.04476*, 2026.

Haoqiang Kang, Yizhe Zhang, Nikki Lijing Kuang, Nicklas Majamaki, Navdeep Jaitly, Yi-An Ma, and Lianhui Qin. LaDiR: Latent diffusion enhances LLMs for text reasoning. *arXiv preprint arXiv:2510.04573*, 2025.

Haoqiang Kang, Yizhe Zhang, Nikki Lijing Kuang, Yi-An Ma, and Lianhui Qin. Beyond mode elicitation: Diversity-preserving reinforcement learning via latent diffusion reasoner. *arXiv preprint arXiv:2602.01705*, 2026.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in Neural Information Processing Systems*, 35, 2022.

Yann LeCun. A path towards autonomous machine intelligence. *Open Review*, 2022. URL https://openreview.net/pdf?id=BZ5a1r-kVsf.

Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, Tom Goldstein, and Micah Goldblum. Zebra-CoT: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025a.

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. *arXiv preprint arXiv:2509.24251*, 2025b.

Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulic, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. *arXiv preprint arXiv:2501.07542*, 2025c.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

Luyang Liu, Jonas Pfeiffer, Jiaxing Wu, Jun Xie, and Arthur Szlam. Deliberation in latent space via differentiable cache augmentation. *arXiv preprint arXiv:2412.17747*, 2024.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *Advances in Neural Information Processing Systems*, 2023.

Zesen Lyu, Dandan Zhang, Wei Ye, Fangdi Li, Zhihang Jiang, and Yao Yang. Jigsaw-puzzles: From seeing to understanding to reasoning in vision-language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang, and Wenqi Shao. MM-Eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2503.07365*, 2025.

Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. Compositional chain-of-thought prompting for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14420–14431, 2024.

Debjyoti Mondal, Suraj Modi, Subhadarshi Panda, Rituraj Singh, and Godawari Sudhakar Rao. Kam-cot: Knowledge augmented multimodal chain-of-thoughts reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 2024.

Steven Pinker. *The Language Instinct*. William Morrow and Company, New York, 1994.

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. *arXiv preprint arXiv:2511.19418*, 2025.

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 (ICLR)*, 2025.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. *arXiv preprint arXiv:2403.16999*, 2024a.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024b.

Xuan Shen, Yizhou Wang, Xiangxi Shi, Yanzhi Wang, Pu Zhao, and Jiuxiang Gu. Efficient reasoning with hidden thinking. *arXiv preprint arXiv:2501.19201*, 2025.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, and Yu Cheng. OpenThinkIMG: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S. Hu, Pratyusha Sharma, Riashat Islam, Alex Lamb, and John Langford. Next-latent prediction transformers learn compact world models. *arXiv preprint arXiv:2511.05963*, 2025.

Jintao Tong, Jiaqi Gu, Yujing Lou, Lubin Fan, Yixiong Zou, Yue Wu, Jieping Ye, and Ruixuan Li. Sketch-in-latents: Eliciting unified reasoning in mllms. *arXiv preprint arXiv:2512.16584*, 2025.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Ziteng Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024b.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009.

Lev S. Vygotsky. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, 1978.

Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond images and language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. URL https://arxiv.org/abs/2511.21395.

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. *arXiv preprint arXiv:2408.15556*, 2024a.

Xinyi Wang, Lucas Caccia, Oleksiy Ostapenko, Xingdi Yuan, William Yang Wang, and Alessandro Sordoni. Guiding language model reasoning with planning tokens. *arXiv preprint arXiv:2310.05707*, 2024b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Lai Wei, Wenkai Wang, Xiaoyu Shen, Yu Xie, Zhihao Fan, Xiaojin Zhang, Zhongyu Wei, and Wei Chen. Mc-cot: A modular collaborative cot framework for zero-shot medical-vqa with llm and mllm integration. *arXiv preprint arXiv:2410.04521*, 2024.

David Wood, Jerome S. Bruner, and Gail Ross. The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2):89–100, 1976.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

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. *arXiv preprint arXiv:2407.01863*, 2024.

Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. Llava-cot: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2024.

Yi Xu, Chengzu Li, Han Zhou, Xingchen Wan, Caiqi Zhang, Anna Korhonen, and Ivan Vulić. Visual planning: Let's think only with images. *arXiv preprint arXiv:2505.11409*, 2025.

Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Zeyuan Yang, Xueyang Yu, Delin Chen, Maohao Shen, and Chuang Gan. Machine mental imagery: Empower multimodal reasoning with latent visual tokens. *arXiv preprint arXiv:2506.17218*, 2025b.

Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. MM-REACT: Prompting ChatGPT for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STar: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022.

Huanyu Zhang, Wenshan Wu, Chengzu Li, Ning Shang, Yan Xia, Yangyu Huang, Yifan Zhang, Li Dong, Zhang Zhang, Liang Wang, Tieniu Tan, and Furu Wei. Latent sketchpad: Sketching visual thoughts to elicit multimodal reasoning in MLLMs. *arXiv preprint arXiv:2510.24514*, 2025a.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, et al. Thyme: Think beyond images. *arXiv preprint arXiv:2508.11630*, 2025b.

Yi-Fan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. MME-RealWorld: Could your multimodal LLM challenge high-resolution real-world scenarios that are difficult for humans? In *International Conference on Learning Representations (ICLR)*, 2025c.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *Transactions on Machine Learning Research*, 2024.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*, 2025.

A Limitation and Future Work

In line with previous latent reasoning methods, our approach relies on the availability of an intermediate helper image during training; how to extend the framework to settings where such helper images are unavailable or expensive to obtain remains an open question. In addition, while we use value-function heatmaps as the natural helper-image choice for FrozenLake, we do not exhaustively search alternative helper-image designs, and automatically proposing helper images for new domains is left to future work. Finally, each latent block uses a fixed number of K tokens, and adapting the block length to the difficulty of each reasoning step is a natural extension.

B Additional Related Work

Think with Image. A second line of work lets the model actively manipulate or generate intermediate images *during* reasoning. Tool-augmented systems call external tools or code to crop, zoom, or annotate the input (Yao et al., 2023; Lu et al., 2023; Hu et al., 2024; Yang et al., 2023); image-generation systems autoregressively visualize new images (Li et al., 2025c; Gu et al., 2025; Xu et al., 2025; Yang et al., 2025a; Zhang et al., 2025b) or use diffusion-based generation (He et al., 2025); and RL-driven variants train the VLM to decide when to invoke a tool or emit a new image (Su et al., 2025; Zheng et al., 2025; Meng et al., 2025). These methods preserve visual evidence across reasoning steps but pay a substantial inference-time cost from explicit image manipulation or generation. *In contrast, our method retains a similar multi-step visual reasoning capability while operating entirely in latent space, at near-base-VLM inference cost.*

C Scaffolding RL: Detailed Derivation

This appendix provides the full mathematical derivation of the Scaffolding RL objective summarized in Section 3.3.

Setup. Let $\mathbf{x}$ denote the model input (image and query), and let $\mathbf{z}^{\text{old}} = f_{\theta_{\text{old}}}(\mathbf{x})$ and $\mathbf{z}^{\theta} = f_{\theta}(\mathbf{x})$ denote the Stage 1 latent priors produced by the rollout and current policies. Their Gaussian heads produce $\boldsymbol{\mu}_k^{\text{old}}, \boldsymbol{\sigma}_k^{\text{old}}$ and $\boldsymbol{\mu}_k^{\theta}, \boldsymbol{\sigma}_k^{\theta}$, respectively. For rollout i, the old latent policy samples a residual adjustment at each latent position k:

$$\boldsymbol{\delta}_{i,k}^{\text{old}} \sim \mathcal{N}\big(\boldsymbol{\mu}_k^{\text{old}}, (\boldsymbol{\sigma}_k^{\text{old}})^2\big), \qquad \mathbf{a}_{i,k}^{\text{old}} = \mathbf{z}_k^{\text{old}} + \boldsymbol{\delta}_{i,k}^{\text{old}}. \tag{4}$$

Stacking all positions gives the rollout latent action $\mathbf{a}_i^{\text{old}} = [\mathbf{a}_{i,1}^{\text{old}}, \ldots, \mathbf{a}_{i,K}^{\text{old}}]$. Conditioned on $\mathbf{x}$ and $\mathbf{a}_i^{\text{old}}$, the model samples a text answer $y_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid \mathbf{x}, \mathbf{a}_i^{\text{old}})$ of length T_i.

Block-level latent importance ratio. We evaluate the same sampled latent action under the rollout and current factorized diagonal Gaussians:

$$p_{\vartheta}(\mathbf{a}_i^{\text{old}} \mid \mathbf{x}) = \prod_{k=1}^{K} \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi}\,\sigma_{k,j}^{\vartheta}} \exp\left[-\frac{1}{2}\left(\frac{a_{i,k,j}^{\text{old}} - z_{k,j}^{\vartheta} - \mu_{k,j}^{\vartheta}}{\sigma_{k,j}^{\vartheta}}\right)^2\right], \qquad \vartheta \in \{\theta, \theta_{\text{old}}\}, \tag{5}$$

where j indexes the d latent dimensions. The block-level latent importance ratio is $\rho_i^{\text{lat}}(\theta) = p_{\theta}(\mathbf{a}_i^{\text{old}} \mid \mathbf{x}) \,/\, p_{\theta_{\text{old}}}(\mathbf{a}_i^{\text{old}} \mid \mathbf{x})$. If a rollout contains several latent blocks, the ratio is computed for each block, and the latent term below is averaged over the blocks.

Token-level text importance ratio. For the discrete answer trajectory we use the standard per-token ratio: $\rho_{i,t}^{\text{text}}(\theta) = \pi_{\theta}(y_{i,t} \mid y_{i,<t}, \mathbf{x}, \mathbf{a}_i^{\text{old}}) \,/\, \pi_{\theta_{\text{old}}}(y_{i,t} \mid y_{i,<t}, \mathbf{x}, \mathbf{a}_i^{\text{old}})$.

Combined clipped objective. Scaffolding RL optimizes a GRPO-style clipped objective that sums the latent term and the text term:

$$\mathcal{L}_{\mathrm{RL}} = -\frac{1}{G}\sum_{i=1}^{G}\Bigg[\min\Big(\rho_i^{\mathrm{lat}}(\theta)A_i,\ \mathrm{clip}\big(\rho_i^{\mathrm{lat}}(\theta),1-\epsilon,1+\epsilon\big)A_i\Big) + \frac{1}{T_i}\sum_{t=1}^{T_i}\min\Big(\rho_{i,t}^{\mathrm{text}}(\theta)A_i,\ \mathrm{clip}\big(\rho_{i,t}^{\mathrm{text}}(\theta),1-\epsilon,1+\epsilon\big)A_i\Big)\Bigg] + \frac{\beta_{\mathrm{KL}}}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i} D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s_{i,t}) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s_{i,t}))\,, \tag{6}$$

where ϵ is the GRPO clipping threshold, G is the rollout group size, A_i is the group-relative advantage, $s_{i,t} = (y_{i,<t}, \mathbf{x}, \mathbf{a}_i^{\mathrm{old}})$ is the token-generation state, π_{ref} is the fixed Stage 1 reference policy, and β_{KL} is the KL coefficient. Since the K latent tokens are predicted jointly, the latent term acts on the sampled latent block as a whole rather than on individual autoregressive steps. During Stage 2, f_θ and both Gaussian heads remain trainable; no gradient is stopped or detached through $\mathbf{z}^\theta$.

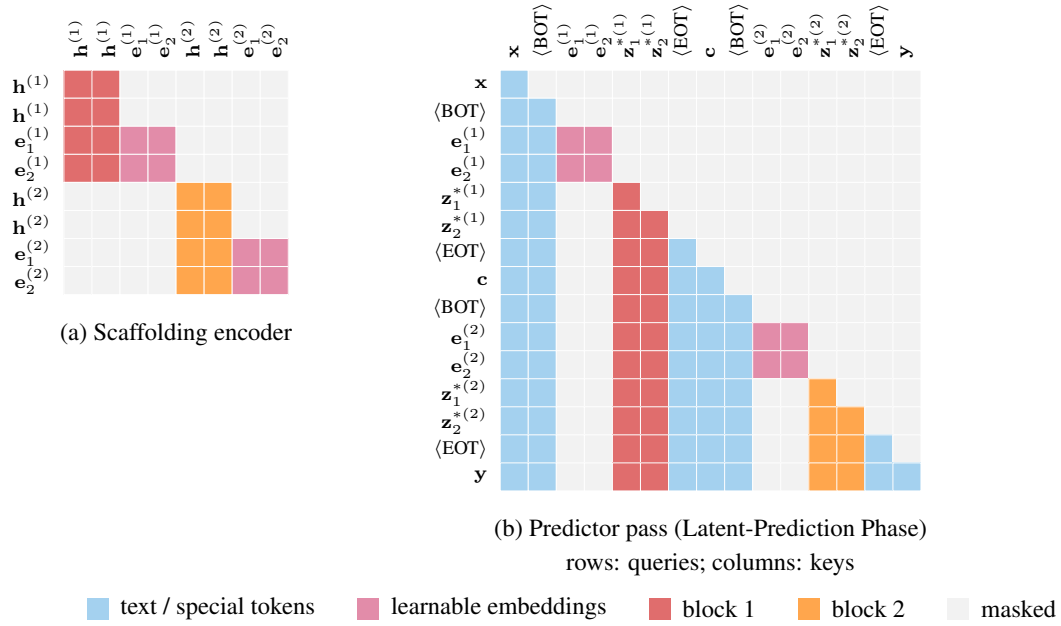
D Experimental Setup Details

D.1 FrozenLake

Data. We use the FrozenLake suite from VSP (Wu et al., 2024). Training data covers even levels from 8 to 32 (8×8 to 32×32 grids), with the per-level training set growing from 50 examples at Level 8 to 650 at Level 32 in steps of 50, for a total of 4,550 training examples. Each level has a held-out test set of 100 examples that we use for evaluation. Each of the four representative levels (8, 16, 24, 32) also has a held-out validation set of 50 examples (half the size of the test set), disjoint from both the training and test sets; we select K, the loss weights $(\lambda_{\mathrm{latent}}, \lambda_{\mathrm{task}})$, and the helper image on this validation set, and all reported numbers are test accuracy. All grid images are padded to a unified 32×32 resolution before being passed to the vision encoder, so that token counts and positional encodings are constant across difficulty levels. The four representative levels reported in Table 1 (8, 16, 24, 32) are chosen to span the full difficulty range.

Backbone and architecture. Unless noted otherwise, all results use Qwen2.5-VL-7B (Bai et al., 2025) as the shared VLM backbone. The scaffolding encoder is implemented as a trainable copy of the VLM's vision encoder followed by a cross-attention pooling module that compresses the resulting visual tokens into $K{=}4$ latent tokens. Latent generation reuses the full VLM weights with $K{=}4$ learnable embeddings that attend to the shared representation to produce the latent block in a single forward pass. In Stage 2, the mean and variance heads are two-layer MLPs operating on the shared VLM hidden states at the latent positions.

Attention masks. A trace can contain several latent blocks, and computing them one after another would require one forward pass per block. We instead process all blocks of a trace in a single forward pass and use attention masks to preserve exactly the dependencies of the sequential computation (Figure 7). The masks follow two principles. *Independence of targets:* in the scaffolding encoder, the helper images and learnable embeddings are packed as $[\mathbf{h}^{(1)}, \mathbf{e}^{(1)}, \mathbf{h}^{(2)}, \mathbf{e}^{(2)}, \ldots]$; each helper image attends only to itself, and each group of embeddings attends bidirectionally to itself and to the helper image right before it. Each target block $\mathbf{z}^{*(b)}$ therefore summarizes only its own helper image, exactly as if the images were encoded one at a time. *No information beyond inference:* in the Latent-Prediction Phase, the model runs two forward passes. In the *predictor pass*, it reads $[\mathbf{x}, \langle\mathrm{BOT}\rangle, \mathbf{e}^{(1)}, \mathbf{z}^{*(1)}, \langle\mathrm{EOT}\rangle, \mathbf{c}, \langle\mathrm{BOT}\rangle, \mathbf{e}^{(2)}, \mathbf{z}^{*(2)}, \langle\mathrm{EOT}\rangle, \mathbf{y}]$. Each group of embeddings attends bidirectionally within the group and sees the input and all preceding text and target blocks, but neither helper images nor the embeddings of earlier blocks, which do not exist at inference. No other token attends to the embeddings, so later blocks condition on the teacher-forced targets $\mathbf{z}^{*(b)}$, which the predicted blocks replace at inference (Appendix F.3). In the second, *answer pass*, the same model reads $[\mathbf{x}, \langle\mathrm{BOT}\rangle, \mathbf{z}^{*(1)}, \langle\mathrm{EOT}\rangle, \mathbf{c}, \ldots, \mathbf{y}]$ with its standard causal mask and generates the text from the latent blocks. In the Scaffolding Phase, the frozen VLM reads the same sequence with its standard causal mask.



(a) Scaffolding encoder

(b) Predictor pass (Latent-Prediction Phase)
rows: queries; columns: keys

Figure 7: **Attention masks for processing all latent blocks of a trace in one forward pass** (two blocks, K=2, two tokens per helper image). **(a)** Scaffolding encoder. **(b)** Predictor pass in the Latent-Prediction Phase. $\mathbf{x}$: input; $\mathbf{c}$: text reasoning; $\mathbf{y}$: answer; $\mathbf{h}^{(b)}$: helper-image tokens; $\mathbf{e}^{(b)}$: learnable embeddings; $\mathbf{z}^{*(b)}$: target latent tokens. Colored cells are attended; gray cells are masked.

Optimization. Stage 1 (scaffolding encoder + latent generation) uses AdamW with learning rate 2×10^{-5} for latent generation and 5×10^{-5} for the scaffolding encoder, weight decay 0.01, batch size 32, cosine schedule with linear warmup over the first 5% of steps, and BF16 mixed precision. We train for 3 epochs over the FrozenLake training set. Stage 2 (Scaffolding RL) uses GRPO with rollout group size G=8, clipping ϵ=0.2, learning rate 1×10^{-6}, and KL penalty β_{KL}=0.001 against the Stage 1 model; the mean head is zero-initialized and the variance head is initialized to σ=0.05. Latent generation loss weights are λ_{latent}=1.0 and λ_{task}=0.5 (default; see Appendix F.5).

Compute. All experiments use NVIDIA A100 (80GB) GPUs. On FrozenLake, Stage 1 takes approximately 7 GPU-hours per run and Stage 2 approximately 24 GPU-hours per run, giving a total of ∼31 GPU-hours per configuration. Inference is performed on a single A100.

Evaluation. We report exact-match accuracy: a generated path is correct only if it both reaches the goal and avoids every hole. The model outputs the full path as a sequence of directional moves (Up/Down/Left/Right); paths that pass through a hole are scored as zero reward, consistent with the VSP setup. All baselines are trained and evaluated by us under the identical data, backbone, and evaluation protocol.

D.2 Visual-Centric Reasoning Benchmarks

Training data. We train on Zebra-CoT (Li et al., 2025a): Stage 1 uses 162K interleaved traces and Stage 2 uses a disjoint 20K-trace split. Each trace already includes its intermediate helper images, which the scaffolding encoder consumes directly; we do not generate separate helper images. All nine benchmarks are evaluated on their full official test sets. No FrozenLake data are used in this experiment, and no Zebra-CoT data are used in the FrozenLake experiments.

Hyperparameters. We reuse the same hyperparameters as for FrozenLake without further tuning: K=4 latent tokens, λ_{latent}=1.0, λ_{task}=0.5, AdamW with the same schedule, and the same Stage 2 GRPO setup. Training runs for 2 epochs over the Zebra-CoT training data. The full two-stage training takes approximately 200 A100 GPU-hours.

Evaluation. We follow each benchmark's standard protocol and report multiple-choice accuracy on all nine benchmarks.

E HELPER IMAGE GENERATION AND ANALYSIS

This appendix describes how the helper images used at training time are generated for each task family, lists the per-benchmark strategies, and analyzes the effect of helper image choice on FrozenLake.

E.1 FROZENLAKE: VALUE-FUNCTION HEATMAP

```
import numpy as np

def value_iteration(grid, gamma=0.85, theta=1e-7):
    """Optimal state-value function for Frozen-Lake grid.
    Codes: 1=start, 2=goal, -1=hole, 0=safe."""
    n_rows, n_cols = grid.shape
    V = np.zeros_like(grid, dtype=float)
    actions = [(-1,0), (1,0), (0,-1), (0,1)]
    is_terminal = lambda r,c: grid[r,c] in (-1,2)
    while True:
        delta = 0.0
        for r in range(n_rows):
            for c in range(n_cols):
                if is_terminal(r,c): continue
                v_old = V[r,c]
                V[r,c] = max(
                    ((0. if grid[nr,nc]==-1 else
                      1. if grid[nr,nc]==2 else 0.)
                     + gamma * (0. if grid[nr,nc] in (-1,2)
                                else V[nr,nc]))
                    if 0<=nr<n_rows and 0<=nc<n_cols
                    else gamma * V[r,c]
                    for dr,dc in actions
                    for nr,nc in [(r+dr, c+dc)])
                delta = max(delta, abs(v_old - V[r,c]))
        if delta < theta: break
    V[grid == -1] = 0.0; V[grid == 2] = 1.0
    return V
```

E.2 VISUAL-CENTRIC REASONING BENCHMARKS

For visual-centric reasoning benchmarks, every Zebra-CoT training example already contains one or more intermediate helper images paired with its query (Li et al., 2025a). The scaffolding encoder consumes these provided helper images directly during Stage 1 training; we do not generate a separate helper image for these tasks.

F Additional Experimental Results

F.1 Ablation on Hyperparameters

Table 4: **Number of latent tokens** K on FrozenLake (Stage 1 only, accuracy %).

K	L8	L16	L24	L32	Avg.
1	85.0	70.0	54.0	45.0	63.5
2	88.0	74.0	57.0	48.0	66.8
4	**94.0**	**79.0**	**63.0**	**52.0**	**72.0**
8	93.0	78.0	61.0	51.0	70.8
16	91.0	76.0	59.0	50.0	69.0

Table 5: **Helper image choice on FrozenLake** (Stage 1 only, accuracy %). *Scaffolding encoder* ✓: our scaffolding encoder is used; ×: a frozen off-the-shelf vision encoder is used as in prior work.

Helper image	Scaffold. enc.	L8	L16	L24	L32	Avg.
Red-arrow overlay	×	85.0	67.0	52.0	40.0	61.0
	✓	92.0	76.0	58.0	46.0	68.0
Value-function heatmap	×	88.0	71.0	47.0	25.0	57.8
	✓	**94.0**	**79.0**	**63.0**	**52.0**	**72.0**

Number of latent tokens K. We sweep $K \in \{1, 2, 4, 8, 16\}$ on FrozenLake (Levels 8, 16, 24, and 32) while keeping all other Stage 1 settings fixed (Table 4). Accuracy rises sharply from $K{=}1$ (63.5%) to $K{=}4$ (72.0%), then falls back at $K{=}8$ (70.8%) and $K{=}16$ (69.0%); $K{=}4$ is the best setting at every level. This non-monotonic pattern reflects two opposing pressures: too few tokens cannot encode a multi-step spatial plan, while too many tokens add redundancy that latent generation must learn to match. $K{=}4$ strikes the balance between expressiveness and learnability.

Helper image choice. We cross two helper images on FrozenLake — the dense *value-function heatmap* (our default) and a sparse *red-arrow overlay* that draws the optimal path on the raw grid — with the choice between our scaffolding encoder and a frozen off-the-shelf encoder (Table 5, 2×2). **First**, a frozen off-the-shelf encoder cannot exploit a richer helper image: with this encoder, the dense heatmap is actually worse than the sparse red-arrow overlay (57.8 vs. 61.0), because an encoder trained on natural images cannot readily use the non-natural heatmap. **Second**, the scaffolding encoder fixes this. It beats the off-the-shelf encoder for both helpers, with the largest gain on the non-natural heatmap (+14.3 points, 57.8 → 72.0) versus only +7.0 points on the red-arrow overlay (61.0 → 68.0); the ordering between helpers also flips, with the heatmap now beating the red-arrow overlay (72.0 vs. 68.0).

F.2 Additional Ablations

Parameter-matched control. The scaffolding encoder tunes a copy of the vision encoder, so its gain could come from added trainable parameters rather than from a better latent target. To test this on FrozenLake, we fine-tune the same vision encoder in plain SFT without any latent target. Tuning the encoder alone raises SFT from 52.0% to 56.5% (90/72/44/20 on L8/16/24/32), whereas learning the latent target with the same tuned encoder reaches 72.0%. Added capacity therefore explains only a small part of the gain. Consistently, a frozen encoder cannot exploit a richer helper image (Table 5). The scaffolding encoder, a ∼677M-parameter copy of the vision tower plus a ∼80M-parameter attention-pooling module (∼0.76B in total, ∼9.1% of the 8.29B backbone), is used only during training and removed at inference.

Stage 1 and Stage 2 are complementary. To test whether Stage 2 can compensate for a weak latent target, we apply the three Stage 2 recipes to two Stage 1 checkpoints: ours (72.0%) and a prior-work checkpoint with a frozen target (mean pooling) (57.8%) (Figure 2 and Table 3). On the weak checkpoint, Scaffolding RL still gives the largest gain (+5.2 points, versus +1.2 points for VLPO and +0.6 points for text-only GRPO), even larger than its +3.0 points on our checkpoint because there is more headroom. Yet it reaches only 63.0%, below even the weakest recipe on top of the scaffolding encoder (72.6%). Stage 2 therefore refines the latent target that Stage 1 provides but cannot replace it, and the two stages are complementary.

Ablation on visual-centric benchmarks. To test whether both components help beyond FrozenLake, we repeat the two key FrozenLake ablations on the nine visual-centric benchmarks. All variants use Qwen2.5-VL-7B, the Zebra-CoT data of Section 4.2, $K{=}4$, and the hyperparameters

of Appendix D.2. In Stage 1, mirroring the SFT visual target ablation (Figure 6), both variants are trained on the same 162K SFT traces with the same latent predictor and differ only in the latent target: a frozen target from mean-pooled vision-encoder features, as in prior work, or targets from our scaffolding encoder. In Stage 2, mirroring Table 3, both variants start from our Stage 1 checkpoint (70.9% average; Table 2), use the same 20K RL traces and rollout budget, and differ only in the RL recipe. The scaffolding encoder raises the Stage 1 average from 66.3% to 70.9%, and Scaffolding RL raises the Stage 2 average from 72.7% (VLPO) to 73.9%, improving every benchmark in both cases (Table 6). Both components therefore transfer beyond spatial planning.

Table 6: **Ablation on the visual-centric reasoning benchmarks** (accuracy %). All rows use the same backbone and Zebra-CoT training data. Stage 1 rows differ only in the latent target; Stage 2 rows start from the same Scaffolding Encoder checkpoint and differ only in the RL recipe.

	Variant	V*	BLINK	MMVP	MMStar	CVB.	HRB-4K	HRB-8K	MME	Jig.	Avg.
Stage 1	Frozen target	82.4	57.5	71.3	67.8	79.0	72.0	68.3	45.2	52.8	66.3
	Scaffolding encoder	**87.4**	**64.8**	**73.3**	**72.1**	**86.4**	**73.6**	**71.1**	**53.4**	**56.0**	**70.9**
Stage 2	VLPO	89.3	66.2	75.3	72.9	87.6	75.9	72.9	55.7	58.2	72.7
	Scaffolding RL	**90.6**	**67.2**	**76.7**	**73.5**	**88.4**	**77.5**	**74.1**	**57.3**	**59.7**	**73.9**

F.3 Teacher-Forced vs. Self-Predicted Latents

In the Latent-Prediction Phase of Stage 1, the model runs two forward passes per trace: a *predictor pass* that generates all latent blocks, and an *answer pass* of the same model that generates the text from them. We train both with teacher forcing. This is convenient, but it raises an obvious question. During training, the predictor pass computes each latent block from a context that contains the preceding *target* blocks $\mathbf{z}^*$, and the answer pass conditions on these targets as well; at inference, both only see the blocks the model predicted itself. If the two differ noticeably, the model may learn to rely on context it never gets at test time. We test this directly. As a reference, we train a self-predicted variant in which later tokens condition on the preceding *predicted* blocks $\hat{\mathbf{z}}$ during training as well, with a stop-gradient $\mathrm{sg}(\hat{\mathbf{z}})$ so that no gradient flows back through this context. We run the comparison in the interleaved visual-centric setting, where each latent block builds on the ones before it, so any mismatch would accumulate across blocks. Everything else (the Zebra-CoT data, the backbone, and the hyperparameters) stays the same, and we evaluate the resulting Stage 1 models on four benchmarks. In terms of accuracy, the two variants end up almost identical: on every benchmark the difference is at most 0.4 points (Table 7). The cost, however, is quite different. The self-predicted variant has to finish each latent block before it can predict the next one, so it needs B sequential predictor passes instead of one, that is, $B{+}1$ forward passes per trace in total instead of 2, and in practice it trains about $3.2\times$ slower. In short, teacher forcing introduces no noticeable mismatch here, since the matching loss already keeps the predicted blocks close to their targets, and it is much cheaper to train. We therefore use it in all experiments.

Table 7: **Teacher-forced vs. self-predicted latent context** (Stage 1, accuracy %). *Forward passes*: predictor passes + answer pass per training trace with B latent blocks. *Training time*: Stage 1 training time relative to teacher forcing.

Preceding latent context	V*	MMVP	CVB.	HRB-4K	Mean	Forward passes	Training time
Target $\mathbf{z}^*$ (teacher forcing, ours)	87.4	73.3	86.4	73.6	80.2	$1{+}1$	$1.0\times$
Predicted $\mathrm{sg}(\hat{\mathbf{z}})$ (self-predicted)	87.1	72.9	86.2	73.3	79.9	$B{+}1$	$3.2\times$

F.4 RL Exploration Analysis

Table 3 and Figure 2 show that Scaffolding RL gains more, and more stably, than VLPO and text-only GRPO. Here we examine how the three recipes explore during training to identify where this advantage comes from. All recipes start from the same Stage 1 checkpoint on FrozenLake (72.0%) with the same rollout budget, and we track three metrics (Table 8). The *reward standard deviation within a rollout group* measures how different the outcomes of the rollouts for the same prompt are;

GRPO's group-relative advantage vanishes when all rollouts receive the same reward, so a larger spread gives a more informative update. *Pass@k* is the percentage of training prompts for which at least one of k rollouts is correct, and measures whether exploration reaches correct solutions. The *latent Gaussian scale* σ indicates how far latent samples move from the predicted latent. Text-only GRPO explores only through text sampling and has the lowest reward spread (0.15) and pass@16 (80). VLPO also samples only text, since it regularizes the latent block with a fixed scale instead of sampling it; its spread (0.32) is higher than that of text-only GRPO, because updating the latent block also shifts the text rollouts that condition on it, but it has no latent exploration to adapt. Scaffolding RL samples latent adjustments with a learned, input-dependent scale (0.04–0.15), which gives the largest reward spread (0.42) and the highest pass@k for every k. The advantage of Scaffolding RL therefore comes from latent exploration: sampling alternative latent reasoning paths yields more diverse and more often successful rollouts, which gives GRPO a stronger learning signal, and the learned scale keeps this exploration adaptive.

Table 8: **RL exploration metrics** on FrozenLake, all from the same Stage 1 checkpoint (72.0%). *VLPO does not sample latents; 0.05 is the fixed scale σ_0 of its Gaussian regularizer (Section 2).

Metric	Text-only GRPO	VLPO	Scaffolding RL
Reward s.d. within a rollout group	0.15	0.32	**0.42**
Pass@4 / @8 / @16 (%)	74 / 76 / 80	75 / 79 / 82	**79 / 82 / 85**
Latent Gaussian scale σ	N/A	0.05* (fixed)	0.04–0.15 (learned)

F.5 Latent Generation Loss Weights

We sweep the matching weight λ_{latent} and the task weight λ_{task} of the Latent-Prediction Phase objective on FrozenLake (Levels 8, 16, 24, and 32; Table 9); λ_{latent} is kept positive, since the matching term is what trains the latent prediction. Without the task term (λ_{task}=0, 64.8%), the model only learns to reproduce $\mathbf{z}^*$ and not to answer from it, which is the weakest setting. Adding the task term improves every level, and the default λ_{latent}=1.0, λ_{task}=0.5 gives the best average accuracy (72.0%): the matching term keeps the predicted latent close to its target, and the task term trains the same model to answer from the latent blocks.

Table 9: **Latent generation loss weights** on FrozenLake (accuracy %; Avg. over the four levels shown). λ_{latent} and λ_{task} weight the latent matching term and the task loss term, respectively.

λ_{latent}	λ_{task}	L8	L16	L24	L32	Avg.
1.0	0.0	88.0	72.0	54.0	45.0	64.8
1.0	**0.5**	**94.0**	**79.0**	**63.0**	**52.0**	**72.0**
1.0	1.0	92.0	77.0	60.0	51.0	70.0
0.5	1.0	93.0	78.0	62.0	52.0	71.3

F.6 Effect of Model Scale and LoRA Fine-tuning

To check that the framework's gains are not specific to the 7B backbone or to full fine-tuning, we repeat the FrozenLake experiment under two variants: a smaller Qwen2.5-VL-3B backbone, and a parameter-efficient LoRA setup on the same Qwen2.5-VL-7B backbone (rank r=32, α=64, dropout 0.05, applied to all attention and MLP projection matrices). Both runs use the same data, helper images, hyperparameters, and evaluation protocol as the main run; only the backbone size or trainability scope changes. Table 10 reports the joint result. At 3B scale, Scaffolding Encoder+RL improves over the 3B SFT baseline by $+17.2$ points on average, and the gap is largest on the harder levels ($+22$ points at Level 24 and $+18$ points at Level 32); its $+5.7$-point improvement over the strongest prior latent baseline is preserved. The LoRA setup also improves by $+17.0$ points over LoRA SFT, although its absolute average remains below 3B full fine-tuning (58.8% vs. 64.0%). Full fine-tuning at 7B remains the strongest configuration.

Table 10: **Effect of model scale and LoRA fine-tuning on FrozenLake** (accuracy %). Left: Qwen2.5-VL-3B with full fine-tuning. Right: Qwen2.5-VL-7B with LoRA r=32.

	Qwen2.5-VL-3B (full FT)					Qwen2.5-VL-7B (LoRA r=32)				
Method	**Lv. 8**	**Lv. 16**	**Lv. 24**	**Lv. 32**	**Avg.**	**Lv. 8**	**Lv. 16**	**Lv. 24**	**Lv. 32**	**Avg.**
SFT	75.0	60.0	38.0	14.0	46.8	70.0	52.0	32.0	13.0	41.8
SFT+GRPO	77.0	62.0	40.0	16.0	48.8	71.0	54.0	34.0	14.0	43.3
LVR (Li et al., 2025b)	78.0	63.0	41.0	18.0	50.0	73.0	56.0	36.0	16.0	45.3
Mirage (Yang et al., 2025b)	79.0	65.0	43.0	20.0	51.8	74.0	58.0	38.0	18.0	47.0
VaLR (Jeon et al., 2026)	83.0	71.0	51.0	28.0	58.3	80.0	65.0	44.0	24.0	53.3
Scaffolding Encoder	86.0	73.0	56.0	30.0	61.3	**84.0**	64.0	48.0	32.0	57.0
Scaffolding Encoder+RL	**88.0**	**76.0**	**60.0**	**32.0**	**64.0**	82.0	**67.0**	**51.0**	**35.0**	**58.8**

F.7 Inference Efficiency

We measure whether latent reasoning keeps inference close to the cost of the base VLM, both when a trace has a single latent block (FrozenLake) and when latent blocks are interleaved with text (the visual-centric benchmarks). We report single-query latency on one A100 in BF16, averaged over 100 queries (Table 11). At inference, the scaffolding encoder is removed; each latent block adds only K=4 positions computed in one forward pass, and the two Gaussian heads are small MLPs. Our method adds +4.6% latency on FrozenLake and +5.9% on the visual-centric benchmarks, within the range of other latent reasoning methods (+2.3% to +8.4%). Tool-calling and image-generation methods add +76% to +277% because they process or generate explicit images at each reasoning step. Our method thus stays near base-VLM cost in both single-block and interleaved settings.

Table 11: **Single-query latency** (single A100, BF16, averaged over 100 queries). K: latent tokens per block (— for methods that process or generate images). –: not measured.

		FrozenLake (single block)		Visual-centric (interleaved)	
Method	***K***	**Latency (ms)**	**Overhead**	**Latency (ms)**	**Overhead**
Qwen2.5-VL-7B (base)	—	260	—	740	—
Latent reasoning methods					
Mirage (Yang et al., 2025b)	4	266	+2.3%	–	–
LVR (Li et al., 2025b)	4	267	+2.7%	794	+7.3%
VaLR (Jeon et al., 2026)	4	268	+3.1%	802	+8.4%
SkiLa (Tong et al., 2025)	4	269	+3.5%	–	–
CoVT (Qin et al., 2025)	8	274	+5.4%	–	–
Monet (Wang et al., 2026)	8	276	+6.2%	–	–
Scaffolding Encoder+RL (Ours)	4	272	+4.6%	784	+5.9%
Tool-calling & image-generation methods					
DeepEyes (Zheng et al., 2025)	—	∼540	+108%	1358	+83.5%
Thyme (Zhang et al., 2025b)	—	∼680	+162%	1305	+76.4%
VPRL (Xu et al., 2025)	—	∼820	+215%	–	–
DiffThinker (He et al., 2025)	—	∼910	+250%	–	–
MVoT (Li et al., 2025c)	—	∼980	+277%	–	–