SVoT: State-aware Visualization-of-Thought for Spatial Reasoning via Reinforcement Learning

Chao Lei Yanbei Jiang Markus Hiller Zhijian Zhou
Xunye Tian Krista A. Ehinger Nir Lipovetzky
School of Computing and Information Systems
The University of Melbourne
Melbourne, Australia
{clei1,yanbeij}@student.unimelb.edu.au
{zhijianzhou.ml, xunyetian.ml}@gmail.com
{m.hiller, nir.lipovetzky, kris.ehinger}@unimelb.edu.au

Abstract

Spatial reasoning remains a challenge for Multimodal Large Language Models (MLLMs), as it requires reliable multi-hop inference over both intermediate states and state transitions. Current studies often leave intermediate states unverified and treat state transitions as implicit processes, which limits reliability in multi-hop spatial reasoning. To address this, we propose State-aware Visualization-of-Thought (SVoT), a reinforcement learning framework that generates interleaved, verifiable intermediate states and visualizations. SVoT integrates transition reasoning chains into the generation processes, enabling the model to verify action preconditions and effects through interleaved textual and visual reasoning. We train SVoT via Group Relative Policy Optimization (GRPO), instantiating verification through reward design and evaluating the efficacy of different fine-grained rewards. As existing benchmarks reduce state transitions to single-variable updates, substantially simplifying the problems, we establish five domains by extending classical environments and introducing two novel domains, PACMAN and GATHER, that require multi-object interactions and numerical reasoning. These domains support systematic evaluation of multi-hop spatial reasoning with quantitative verification of generated intermediate states and transition reasoning. SVoT with transition-aware supervision achieves state-of-the-art performance across the introduced domains, yielding up to a 65% absolute accuracy gain on out-of-distribution test sets.

1 Introduction

Planning constitutes a core reasoning capability in Multimodal Large Language Models (MLLMs) [1]. A prominent evaluation setting for planning in MLLMs is spatial reasoning [2], a domain essential to navigation [3, 4], robotics [5], and autonomous driving [6, 7]. Spatial reasoning entails the ability to perceive and analyze object relationships, state transitions, and environmental interactions [8]. Although existing Visual Question Answering (VQA) benchmarks [9–12] have been proposed to evaluate spatial reasoning in MLLMs, they primarily emphasize static spatial relations, e.g., relative object positions, with limited coverage of planning-oriented capabilities. In contrast, multi-hop spatial reasoning requires sequential state tracking, i.e., accurate next-state predictions under a given action sequence [8, 13], which is central to planning. This dynamic nature compels the model to consistently identify objects, understand temporal evolution, and infer action-conditioned state transitions.

Grid-based environments provide a natural setting for evaluating sequential state tracking. They define discrete state variables (e.g., coordinates) and deterministic transition dynamics, which facilitate rigorous verification of state updates and validity constraints. Recent studies [8, 13] evaluate

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

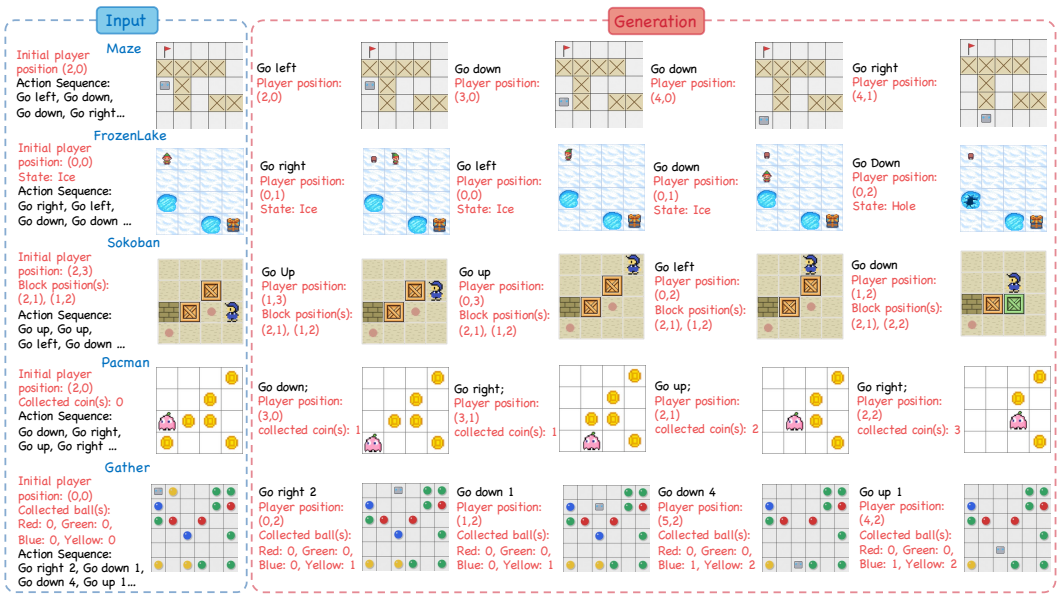


Figure 1: Illustration of the five domains used in SVoT. Coordinates are (row, column), starting from (0,0) at the top-left corner. For conciseness, we omit the domain description from the input.

sequential state tracking within grid-world benchmarks such as MAZE [14], FROZENLAKE [15], and POLYOMINO TILINGS [16]. Moreover, grid-based representations are widely used in real-world spatial applications, where continuous spaces are discretized into occupancy grids or cost maps for tractable collision checking, path planning, and multi-agent coordination in domains such as autonomous driving, robot navigation, and warehouse robotics [17–20]. Critically, accurate state tracking is essential for safe planning in these real-world applications. Thus, improving the reliability and verifiability of sequential state tracking in grid-based environments addresses a fundamental requirement for applying MLLMs to real-world spatial reasoning tasks.

Benefiting from the faculties of constructing mental images and performing mental simulation, humans demonstrate robust spatial awareness [21, 22]. Inspired by this, Wu et al. [8] proposed Visualization-of-Thought (VoT), which constructs a textual visualization after each reasoning step to explicitly track intermediate states. Notably, VoT demonstrates that accurate sequential state tracking is essential for multi-hop spatial reasoning. Advancing this paradigm, Multimodal Visualization-of-Thought (MVoT) [13] identifies native multimodal generation as the promising pathway for implementing sequential state tracking. By leveraging a multimodal-native backbone to generate actual visualizations, MVoT unifies text and vision within the reasoning traces, offering a visually grounded reasoning process. MVoT achieves state-of-the-art performance in sequential state tracking across multiple grid-based domains.

Despite this progress, VoT and MVoT suffer from two critical limitations. First, existing evaluations typically restrict goal specifications to a single state variable or a binary success flag. This reduces each state transition to a single-variable update and weakens the multi-object interactions that make multi-hop spatial reasoning genuinely challenging. Second, the absence of intermediate-state verification and explicit state-transition reasoning prevents reliable assessment of whether the model reasons correctly at each step or merely reaches the correct final state by chance, thereby limiting the reliability of sequential state tracking.

To address both limitations, we propose State-aware Visualization-of-Thought (SVoT), a reinforcement learning (RL)-based framework that improves MLLMs' sequential state tracking through explicit intermediate-state verification and state-transition reasoning. We further evaluate SVoT in richer dynamic environments beyond single-object interactions. In contrast to MVoT, which relies mainly on visualizations as the reasoning trace, SVoT augments the trace with an explicit, structured representation for each intermediate state to support verification and integrates transition reasoning chains into the generation of both intermediate states and visualizations. Transition reasoning chains enable the model to explicitly verify action preconditions and effects through interleaved textual

and visual reasoning, achieving both reliable next-state prediction and faithful visual synthesis. We further examine reward design as the verification mechanism in SVoT, adopting both an Outcome Reward Model (ORM) that supervises only the state description and visualization, and a fine-grained Process Reward Model (PRM) that additionally optimizes the reasoning trajectory.

To support rigorous evaluation of richer dynamics, we establish five grid-based domains, extending classical environments such as MAZE and FROZENLAKE, incorporating SOKOBAN, and introducing the novel PACMAN and GATHER domains, as illustrated in Figure 1. These tasks feature multiple interacting objects (e.g., walls, holes, blocks, coins, balls) with diverse attributes and constraints (e.g., collisions, hazards, positions, counts, colors), requiring explicit transition reasoning to support non-trivial numerical and causal inference. Their discrete, object-centric formulation provides structured representations of states and state transitions, allowing quantitative verification of both state-representation and transition-reasoning correctness. Detailed related work is provided in Appendix A.1.

We outline our contributions as follows: 1) We propose SVoT, a RL-based framework that unifies verifiable intermediate state with visual generation and integrates transition reasoning chains as explicit reasoning steps to enhance MLLMs' sequential state tracking in grid-based environments. 2) We introduce five grid-based domains that enable systematic evaluation of multi-hop spatial reasoning and provide explicit representations of intermediate states and state transitions, supporting quantitative verification of intermediate-state and transition-reasoning correctness. 3) We conduct extensive experiments across the introduced domains, demonstrating that SVoT achieves state-of-the-art performance in sequential state tracking, with up to a 65% absolute improvement on out-of-distribution test sets. We further analyze the impact of alternative reward models, and highlight failure cases and remaining challenges in multi-hop spatial reasoning.

2 State-aware Visualization-of-Thought

In this work, we study sequential state tracking for multi-hop spatial reasoning. We formulate the task as $\mathcal{P} = \langle \mathcal{X}, g \rangle$, where $\mathcal{X} = \langle d, v_0, \mathcal{A} \rangle$ denotes the task specification, comprising the domain description d, the initial-state visualization v_0, and the action sequence $\mathcal{A} = \{a_i\}_{i=1}^{n}$, and g is the ground-truth goal configuration. Given $\mathcal{X}$, a pre-trained MLLM with parameters θ, $\mathcal{M}_\theta$, is expected to generate a sequence of intermediate states $\mathcal{Z} = \{z_i\}_{i=1}^{n}$ and then predict the goal configuration g. Each z_i is generated autoregressively conditioned on $\mathcal{X}$ and the preceding states $z_{<i}$, while g is predicted conditioned on $\mathcal{X}$ and the complete trajectory $\mathcal{Z}$:

$$z_i \sim \mathcal{M}_\theta(\cdot \mid \mathcal{X}, \{z_j\}_{j=1}^{i-1}), \tag{1}$$

$$g \sim \mathcal{M}_\theta(\cdot \mid \mathcal{X}, \{z_j\}_{j=1}^{n}). \tag{2}$$

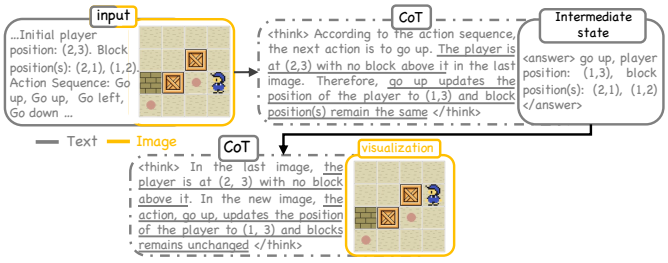


Figure 2: Examples of the CoT (transition reasoning chain) in SVoT used to guide the generation of intermediate state and visualization for the SOKOBAN problem in Figure 1

Although both VoT and MVoT emphasize the significance of intermediate state z_i for sequential state tracking, the definition of z_i remains underspecified. In tasks such as MAZE and FROZENLAKE, MVoT typically reduces z_i to merely the next action prediction, providing sparse information for visualization and subsequent state prediction. In contrast, SVoT formally defines the intermediate state as a tuple $z_i = \langle a_i, s_i \rangle$, containing the action a_i and the state description s_i, resulting from applying a_i to the previous state s_{i-1}. In SVoT, we generate the ground truth for each s_i by anchoring the process with an initial state s_0. Consequently, each s_i is derived via a deterministic transition function f, $s_i = f(s_{i-1}, a_i)$, based on the specified action preconditions and effects. For each domain, Figure 1 illustrates the intermediate state z_i (comprising the ground truth a_i and s_i), and the initial state description s_0 in the input, highlighted in red. The state descriptions s_i in SVoT capture precise object layout, relationships, and attributes to guide subsequent generations and enable the quantitative evaluation of s_i, e.g., verifying the player's coordinate (e.g., $(0, 2)$) or the exact number of collected coins (e.g., 2).

Central to SVoT is the transition reasoning chain c_i, which acts as the transition function, helping the model derive the intermediate state z_i and the visualization v_i by verifying action preconditions and applying action effects, closely aligning with classical planning assumptions. Unlike standard Chain-of-Thought (CoT) in LLMs, c_i encodes both semantic and spatial information (e.g., "the player is at (2,3) with no block above it"), as illustrated in Figure 2. This enables SVoT to move beyond the implicit state updates of MVoT toward a structured and verifiable reasoning process. By grounding each generation in a precise state-updating process, c_i bridges abstract spatial reasoning and concrete multimodal generation, ensuring that state descriptions and visualizations remain consistent with the underlying dynamics. For notational clarity, we use c_i^z and c_i^v to denote the transition reasoning chains for z_i and v_i, respectively, while using c_i as the general form (or simply "CoT" for brevity).

To unify reasoning and generation for sequential state tracking, SVoT augments the task specification $\mathcal{X}$ with an initial state description s_0, resulting in $\mathcal{X} = \langle d, s_0, v_0, \mathcal{A} \rangle$. SVoT represents each intermediate state as $z_i = \langle a_i, s_i \rangle$. In addition, SVoT produces a corresponding visualization v_i for each z_i. Crucially, the MLLM is instructed to generate a transition reasoning chain c_i in conjunction with z_i and v_i, yielding an interleaved multimodal generation process that naturally supports CoT reasoning. In SVoT, c_i, z_i, and v_i are predicted conditioned on $\mathcal{X}$ and the preceding generated states and visualizations, and the goal configuration g is then predicted from the complete multimodal trajectory. To satisfy context-window constraints while preserving the explicit trajectory, we exclude previously generated transition reasoning chains from the conditioning context in each generation. Equations (3)–(5) formalize the SVoT generation process:

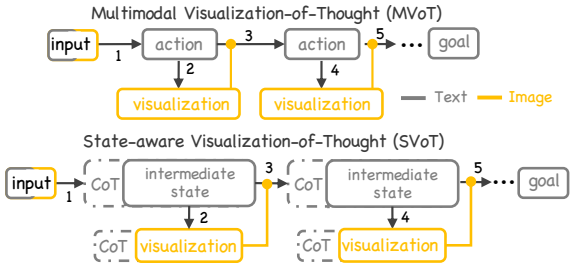


Figure 3: The architectures of MVoT and SVoT.

$$c_i^z, z_i \sim \mathcal{M}_\theta(\cdot \mid \mathcal{X}, \{z_j, v_j\}_{j=1}^{i-1}), \tag{3}$$

$$c_i^v, v_i \sim \mathcal{M}_\theta(\cdot \mid \mathcal{X}, \{z_j, v_j\}_{j=1}^{i-1}, z_i), \tag{4}$$

$$g \sim \mathcal{M}_\theta(\cdot \mid \mathcal{X}, \{z_j, v_j\}_{j=1}^{n}). \tag{5}$$

Figure 3 illustrates the architectural differences between SVoT and MVoT. In MVoT, the intermediate state z_i is typically reduced to an isolated action, e.g., $z_i = a_i$, without explicit transition reasoning to support subsequent state or visualization generation. For conciseness, we make no distinction between goal prediction and state prediction in SVoT unless explicitly noted.

3 Training Strategy

SVoT employs Anole-7B [23] as the backbone model, a unified autoregressive MLLM that encodes visual and textual inputs as a single stream of discrete tokens. This architecture supports joint text-and-visual generation within a unified autoregressive sequence, enabling end-to-end optimization. Anole builds on Chameleon [24] without explicitly optimizing CoT capabilities. We implement a two-stage training pipeline to develop CoT-based multi-hop spatial reasoning capabilities for this backbone. In both stages, each training prompt is constructed with the ground-truth multimodal trajectory prefix. In the first stage, we perform Supervised Fine-Tuning (SFT) to train the model to generate transition-reasoning-guided state outputs (e.g., $\{c_i^z, z_i\}$) and visual counterparts (e.g., $\{c_i^v, v_i\}$), utilizing the loss function introduced in MVoT (details in Appendix A.2). This stage establishes a robust initialization for the subsequent RL phase. In the second stage, we employ Group Relative Policy Optimization (GRPO) [25] to encourage exploration of flexible and effective transition reasoning chains for synthesizing z_i and v_i.

SVoT defines three reward functions for GRPO: (1) a state reward r_z for the correctness of each intermediate state z_i, (2) a visual reward r_v for the fidelity of each visualization v_i, and (3) a reasoning reward r_c for the faithfulness of each transition reasoning chain c_i. For the text modality rewards, r_z and r_c, SVoT parses the prediction $\hat{y}$ and the ground truth y into a set of semantic elements $P(y)$ and defines r_z and r_c as the average exact-match accuracy across all fields $p \in P(y)$, as formulated in

Equation (6):

$$r(\hat{y}, y) = \frac{1}{|P(y)|} \sum_{p \in P(y)} \mathbb{I}\left[p(\hat{y}) = p(y)\right]. \tag{6}$$

Here, $\mathbb{I}[\cdot]$ denotes the indicator function. For r_z (where $y = z_i$), $P(z_i)$ consists of discrete fields derived from the action and state description (e.g., action direction, player coordinates, and the number of collected coins in PACMAN). For r_c (where $y = c_i$), $P(c_i)$ contains keywords extracted from action preconditions and effects (e.g., "with no block above it" for preconditions, and "block position(s) remain the same" for effects in SOKOBAN as shown in Figure 2).

For the visual reward r_v, we partition the generated image $\hat{v}$ and the corresponding ground-truth image v into aligned $n \times n$ grids of cells, denoted by $\hat{v}_{ij}$ and v_{ij}, and compute a weighted cell-wise matching score that measures local visual consistency. Since reliable semantic labels are often unavailable, we employ a variance-based heuristic for automatic foreground detection, where a grid cell is classified as foreground if its pixel-wise color variance exceeds a threshold. This allows the r_v to assign higher weights to semantically salient regions (e.g., players and boxes) than to the background, where the heuristic tends to highlight texture-rich objects and treat relatively uniform cells as background. The visual reward r_v combines a cell-level matching term r_{str} with an image-quality term r_{qua}:

$$r_v(\hat{v}, v) = r_{\text{str}}(\hat{v}, v) \cdot r_{\text{qua}}(\hat{v}, v), \tag{7}$$

$$r_{\text{str}}(\hat{v}, v) = \sum_{(i,j)} w_{ij} \cdot \mathbb{I}\left[\mathcal{S}(\hat{v}_{ij}, v_{ij}) \geq \tau\right], \tag{8}$$

$$r_{\text{qua}}(\hat{v}, v) = \text{clip}\left(\frac{\phi(\hat{v}; \Omega)}{\phi(v; \Omega) + \epsilon}, 0, 1\right). \tag{9}$$

Here, r_{str} measures cell-level consistency using structural, edge, and color features, while r_{qua} penalizes blurry predictions by measuring foreground sharpness. In Eq. (8), a cell contributes to the reward only when its composite similarity score $\mathcal{S}$ exceeds the threshold τ. The weights w_{ij} are normalized such that $\sum w_{ij} = 1$ and assign higher weight to foreground cells than background cells to encourage semantic focus. $\mathcal{S}$ combines three features:

$$\mathcal{S}(\cdot) = \lambda_h s_{\text{hash}}(\cdot) + \lambda_e s_{\text{edge}}(\cdot) + \lambda_c s_{\text{color}}(\cdot), \tag{10}$$

where s_{hash} measures perceptual-hash similarity to capture global structure, s_{edge} computes edge-map IoU using Sobel operators to reflect local texture boundaries, and s_{color} evaluates joint RGB histogram correlation to assess color consistency. In Eq. (9), Ω denotes the foreground cells, and $\phi(\cdot; \Omega)$ calculates the average Laplacian variance within Ω to measure sharpness. Detailed visual reward formulation and hyperparameter settings are available in Appendices A.3 and A.13.

For each prompt, GRPO samples a group of G completions $\{o_g\}_{g=1}^{G}$. In SVoT, completions within a group belong to the same output type, either $\{c^z, z\}$ or $\{c^v, v\}$. The reward r_g for completion o_g is computed as a weighted sum of individual rewards (e.g., $r_g = 0.5\, r_c + 0.5\, r_z$). GRPO then normalizes rewards within each group to compute relative advantages, promoting higher-reward completions while suppressing suboptimal ones. SVoT optimizes the policy using the standard GRPO objective. Full details of GRPO in SVoT are available in Appendix A.4.

In SVoT, the reasoning reward r_c evaluates the fidelity of the transition reasoning chain against its ground truth, ensuring that the model's logic is sound and faithful rather than solely optimized for the output, an approach aligning with a Process Reward Model (PRM) [26, 27]. In contrast, we explore a simplified r_g that retains only r_z or r_v, while removing reasoning reward r_c akin to an Outcome Reward Model (ORM) [27]. We empirically investigate whether explicit supervision of the transition reasoning process via PRM yields superior sequential state tracking performance compared to sparse, outcome-only signals used by ORM. Figure 4 illustrates the GRPO pipeline in SVoT with PRM.

4 Spatial Reasoning Tasks

To support rigorous evaluation of multi-hop spatial reasoning, we establish five grid-based domains (Figure 1) of varying complexity, where the goal configuration g is defined as the final state description induced by the given action sequence $\mathcal{A}$. Actions are single-cell moves in most domains, and

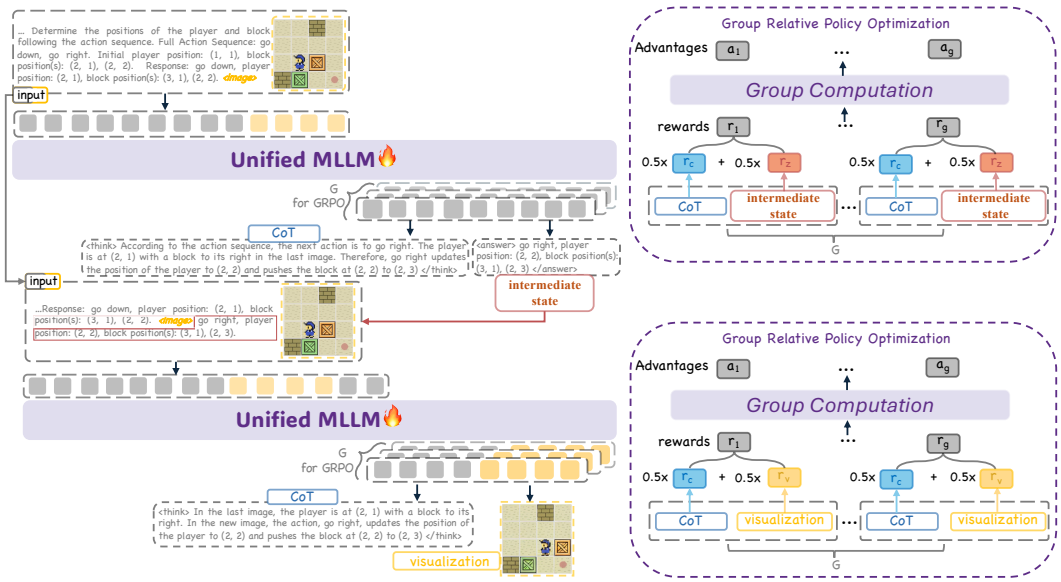


Figure 4: GRPO training in SVoT with PRM. Left: sampling CoT with intermediate state and CoT with visualization completions. Right: reward and advantage computation for GRPO updates.

multi-step moves in GATHER, restricted to the four cardinal directions (up/down/left/right), with domain-specific preconditions and effects. Brief introductions to each domain are provided below, with full construction details and descriptions available in Appendix A.5.

- MAZE: Building on the MAZE environment from MVoT, we additionally include invalid movements that require models to infer action effects, where boundary violations and wall collisions leave the agent's position unchanged. The objective is to predict the agent's final coordinate.
- FROZENLAKE: We adopt FROZENLAKE from MVoT, where tile distinctions (e.g., ice vs. holes vs. goal) demand precise perception. The task is to predict the final player coordinate and tile type.
- SOKOBAN: A classical planning puzzle with multi-object interactions, where player moves can push a block directly in front, resulting in block displacements. The task is to predict the final coordinates of the player and all blocks.
- PACMAN: The player navigates a grid to collect coins, integrating numerical reasoning into sequential state tracking under multi-object interactions. The task requires predicting the cumulative number of collected coins over the action sequence and the agent's final coordinate.
- GATHER: An advanced variant of PACMAN, introducing multi-step actions (e.g., "move down 4 steps") and color-specific objectives, where each movement collects colored balls distributed along the trajectory. The task requires predicting the number of colored balls collected and the agent's final position. This domain presents a substantial challenge, demanding long-horizon state tracking, numerical reasoning, and visual grounding.

Across all domains, we vary grid sizes from 4 to 6 to maintain image quality, while extending to 5 to 7 for MAZE to ensure feasible paths. For each size, we scale the action sequence lengths to evaluate long-horizon state tracking, and adjust the number of interactive objects to ensure diversity. For each domain and grid size, we construct 500 deduplicated training instances, comprising 100 valid maps, each paired with 5 valid action sequences. To assess generalization, we define In-Distribution (ID) and Out-of-Distribution (OOD) test settings. ID instances match the training distribution, while OOD instances introduce longer action sequences and larger numbers of interactive objects, testing robustness to configurations unseen during training. We generate separate ID and OOD test sets by sampling 60 unseen maps per domain and grid size and pairing each map with 2 valid action

Size	Backbone	Method	Maze				FrozenLake				Sokoban				Pacman				Gather			
			ID		OOD		ID		OOD		ID		OOD		ID		OOD		ID		OOD	
			C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F
4	GPT-4o	T-CoT					100	83.3	90.0	56.7	63.3	48.3	56.7	36.7	90.0	26.7	83.3	10.0	53.8	23.3	46.7	13.3
	Anole	T-CoT	-	-	-	-	75.0	33.3	66.7	33.3	65.0	5.0	53.3	0.0	71.7	53.3	50.0	36.7	41.7	6.7	36.7	0.0
		MVoT	-	-	-	-	86.7	46.7	80.0	56.7	73.3	15.0	61.7	3.3	73.3	56.7	63.3	46.7	46.7	13.3	36.7	3.3
		$SVoT_o$	-	-	-	-	**100**	**86.7**	**88.3**	63.3	73.3	71.7	70.0	50.0	**100**	63.3	86.7	61.7	**60.0**	40.0	43.3	10.0
		$SVoT_p$	-	-	-	-	**100**	83.3	**88.3**	**68.3**	**86.7**	**78.3**	**80.0**	**68.3**	**100**	**70.0**	**93.3**	**66.7**	**60.0**	**50.0**	**46.7**	**16.7**
5	GPT-4o	T-CoT	66.7	23.3	56.7	26.7	90.0	80.0	86.7	56.7	63.3	40.0	43.3	20.0	83.3	23.3	78.3	10.0	40.0	0.0	33.3	0.0
	Anole	T-CoT	71.7	16.7	46.7	16.7	73.3	33.3	43.3	26.7	60.0	3.3	50.0	0.0	71.7	51.7	43.3	31.7	31.7	0.0	25.0	0.0
		MVoT	83.3	26.7	56.7	26.7	73.3	43.3	53.3	36.7	61.7	10.0	55.0	3.3	73.3	56.7	53.3	38.3	33.3	3.3	30.0	0.0
		$SVoT_o$	90.0	76.7	60.0	33.3	90.0	83.3	83.3	56.7	70.0	66.7	63.3	45.0	90.0	60.0	76.7	56.7	**46.7**	0.0	30.0	0.0
		$SVoT_p$	**93.3**	**80.0**	**70.0**	**41.7**	**91.7**	**86.7**	**85.0**	**58.3**	**78.3**	**71.7**	**73.3**	**53.3**	**93.3**	**66.7**	**83.3**	**60.0**	**46.7**	**3.3**	**33.3**	**1.7**
6	GPT-4o	T-CoT	63.3	23.3	40.0	16.7	86.7	60.0	76.7	53.3	45.0	36.7	35.0	16.7	76.7	10.0	60.0	8.3	36.7	3.3	26.7	0.0
	Anole	T-CoT	70.0	16.7	36.7	1.7	50.0	36.7	33.3	13.3	58.3	3.3	41.7	3.3	48.3	36.7	40.0	28.3	38.3	0.0	23.3	0.0
		MVoT	83.3	26.7	46.7	3.3	63.3	40.0	43.3	20.0	58.3	6.7	51.7	6.7	60.0	45.0	51.7	26.7	40.0	0.0	20.0	0.0
		$SVoT_o$	86.7	70.0	55.0	26.7	**93.3**	80.0	**76.7**	40.0	66.7	55.0	56.7	33.3	**83.3**	53.3	68.3	41.7	**46.7**	3.3	23.3	0.0
		$SVoT_p$	**90.0**	**81.7**	**66.7**	**36.7**	**93.3**	**83.3**	**76.7**	**55.0**	**71.7**	**68.3**	**65.0**	**48.3**	**83.3**	**60.0**	**75.0**	**56.7**	45.0	**6.7**	**30.0**	1.7
7	GPT-4o	T-CoT	66.7	26.7	36.7	10.0																
	Anole	T-CoT	60.0	3.3	30.0	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		MVoT	73.3	6.7	40.0	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		$SVoT_o$	76.7	63.3	51.7	20.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		$SVoT_p$	**80.0**	**70.0**	**65.0**	**33.3**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 1: Goal coverage accuracy (%) across five domains under ID and OOD settings, evaluated on classification (C) and free-response (F) output formats. SVoT variants are distinguished by subscripts o (ORM) and p (PRM). Best results among Anole-based methods are **bolded**, and results for $SVoT_p$ are highlighted in gray.

sequences under the corresponding configuration, yielding 120 test instances per domain–size pair in each setting. See Appendix A.6 for dataset collection details and domain statistics.

5 Experiments

We fine-tune Anole with LoRA [28], applying updates to all attention matrices and the unified output head (rank $r = 8$, scaling factor $\alpha = 16$, dropout 0.1). The training pipeline consists of 5 epochs of SFT (learning rate 1×10^{-4}), followed by 5 epochs of GRPO (learning rate 5×10^{-6}). In both stages, we maintain a batch size of 8. During GRPO, we sample $G = 4$ completions with a temperature of 1.0. We implement both SFT and GRPO with the TRL library [29] and optimize using AdamW [30]. The entire training process is run on 4 NVIDIA A100 GPUs and takes approximately 10 days.

We begin by evaluating goal prediction accuracy on ID and OOD test sets, each containing 120 instances for every domain–size pair. Formally, given the task specification $\mathcal{X}$, the model must predict the goal state g by autoregressively generating each intermediate state z_i and its corresponding visualization v_i in sequence, guided by the transition reasoning chain c_i. We report performance under two output formats: *Free Response*, where the model provides a textual description of g, and *Classification*, where it selects the correct g from four candidates (see Appendix A.7 for candidate construction details). A prediction is considered correct if it exactly matches the ground truth. We adopt the state-of-the-art solver MVoT as our primary baseline. Additionally, we include two textual CoT (T-CoT) baselines: GPT-4o [31], a strong non-finetuned MLLM, and Anole, which is instruction-tuned to reason step by step by explicitly describing coordinates and the environment layout in text before producing the final answer. For SVoT, we evaluate two variants: $SVoT_o$ (SVoT+ORM) and $SVoT_p$ (SVoT+PRM). For fair comparison, all baselines are provided with the same initial state description as SVoT.

Main Results. As shown in Table 1, free-response evaluation is markedly harder than classification, and OOD further degrades performance across all domains, underscoring the difficulty of long-horizon state tracking. While MVoT outperforms the Anole T-CoT baseline, both SVoT variants

substantially improve over MVoT across most domain–size pairs. By predicting structured transition reasoning chains, intermediate states, and aligned visualizations, SVoT makes state tracking explicitly verifiable and enables precise reward design for GRPO. SVoT_p outperforms both SVoT_o and MVoT in most settings, achieving state-of-the-art performance, particularly under the challenging OOD and free-response settings, with a largest absolute gain of 65% over MVoT on SOKOBAN at size 4. The superior performance of SVoT_p suggests that transition-aware rewards provide more effective guidance than outcome-only signals for complex state tracking tasks. Compared with GPT-4o, SVoT_p generally performs better, while achieving comparable performance on FROZENLAKE. GPT-4o shows limited free-response performance in PACMAN and GATHER, where numerical reasoning introduces additional difficulty, and also degrades on MAZE, where collision detection and boundary-constraint reasoning impose significant challenges. Notably, GATHER remains challenging for all methods despite the relative advantage of SVoT_p. To diagnose this, we analyze single-step prediction accuracy to isolate individual error sources, with results discussed below.

Single-Step Prediction Accuracy. We further investigate the performance drop in GATHER by reporting single-step prediction accuracy for both transition reasoning chain c_i and intermediate state z_i, conditioning on ground-truth inputs. Results are measured over N state predictions across 120 ID and OOD test instances for each grid size. For c_i, we assess the player's position before a multi-step movement (PosOld), after the movement (PosNew), and the ball-collection reasoning (e.g., "collects 1 green ball and 1 blue ball"). For z_i, we evaluate the predicted action, the structured player position (Pos), and the record of collected balls (e.g., "Green: 1, Blue: 1"). As shown in Table 2, the main bottleneck lies in path-dependent ball collection rather than spatial tracking. SVoT maintains robust fidelity in player dynamics (action and position prediction) across all settings, with only a minor drop in position accuracy as grid size increases. In contrast, ball-tracking accuracy degrades sharply with increasing grid size and drops further under OOD shift for both SVoT_o and SVoT_p. SVoT_p maintains ball-tracking accuracy above 40% in both c_i and z_i predictions. Nevertheless, multi-step collection of multi-colored balls introduces ambiguity and amplifies error accumulation over long horizons, contributing to the low performance reported in Table 1.

Size	Setting (N)	Method	Transition Reasoning Chain			Intermediate State		
			PosOld	PosNew	Ball	Action	Pos	Ball
4	ID (646)	SVoT_o	97.83	**98.92**	72.76	**100.0**	**97.68**	68.27
		SVoT_p	**98.92**	93.65	**74.46**	**100.0**	93.34	**74.92**
	OOD (1196)	SVoT_o	96.74	**93.48**	40.89	87.79	90.89	40.89
		SVoT_p	**97.83**	92.39	**57.36**	**95.07**	**91.39**	**51.09**
5	ID (743)	SVoT_o	**97.98**	94.75	50.87	**95.42**	92.19	48.05
		SVoT_p	**97.98**	**95.83**	**55.99**	**95.42**	**95.42**	**55.99**
	OOD (1275)	SVoT_o	**96.78**	93.49	40.47	93.80	92.55	40.47
		SVoT_p	**96.78**	**94.67**	**47.84**	**95.84**	**93.80**	**44.78**
6	ID (951)	SVoT_o	96.85	90.43	40.90	94.01	90.22	35.12
		SVoT_p	**98.95**	**95.69**	**45.95**	**98.00**	**94.64**	**45.95**
	OOD (1418)	SVoT_o	94.64	**94.64**	31.73	88.58	83.29	31.73
		SVoT_p	**95.70**	**94.64**	**44.71**	**92.88**	**89.07**	**41.47**

Table 2: Single-step prediction accuracy (%) of the transition reasoning chain and intermediate state for SVoT_o and SVoT_p within the GATHER domain.

Visualization Qualitative Analysis. Figure 5 illustrates representative visualizations generated under the OOD and free-response settings in Table 1. MVoT-generated images are prone to blurring as pattern complexity increases (e.g., in FROZENLAKE and PACMAN) and often exhibit inaccuracies (e.g., in GATHER). In contrast, SVoT mitigates blur by penalizing vague generations via the visual reward. Notably, SVoT_p produces high-fidelity visualizations; for instance, in GATHER, its transition reasoning chain correctly tracks the ball-collection event (e.g., "collects 1 red ball and 1 blue ball"), and the visualization accurately reflects this collection dynamic. Conversely, SVoT_o suffers from an under-optimized reasoning chain (e.g., "balls remain unchanged"), resulting in a generated image that updates the agent's position but omits the corresponding ball collection.

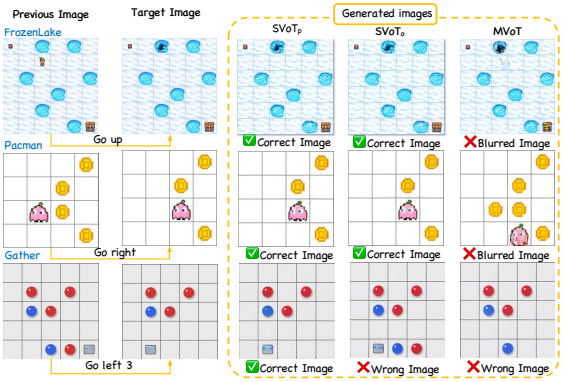


Figure 5: Example visualizations generated in Table 1 under the OOD and free-response settings.

Visualization Quantitative Analysis. Table 3 reports quantitative evaluation of generated images in Table 1 under OOD and free-response settings, computed using the visual reward r_v (Eq. (7)). Across all domains, SVoT achieves higher r_v scores than MVoT, reflecting the effectiveness of r_v in guiding generation. Between the two variants, SVoT_p consistently outperforms SVoT_o despite sharing the same visual reward. This suggests that PRM produces more faithful and generalizable visualizations by promoting accurate state-transition reasoning. In GATHER, the lower scores of both SVoT variants align with the reduced transition-reasoning and intermediate-state prediction accuracy in Table 2, indicating that visual quality is closely tied to the correctness of the underlying transition reasoning and state representation.

Method	Maze	FrozenLake	Sokoban	Pacman	Gather
MVoT	0.8056	0.8361	0.7962	0.7954	0.7414
SVoT_o	0.8243	0.8464	0.8238	0.8149	0.7482
SVoT_p	**0.8571**	**0.8576**	**0.8578**	**0.8449**	**0.7716**

Table 3: Quantitative analysis of generated images in Table 1 aggregated by size under OOD and free-response settings. Scores are computed using Eq. (7). See Appendix A.8 for additional experiments.

Ablation Study. Table 4 details the ablation study of SVoT_p under the same experimental settings as Table 1. We report free-response results for grid sizes 4 to 6 to save space. The removal of visualizations (*w/o*-V) leads to a clear performance decline, suggesting that interleaved images provide essential visual cues to facilitate state tracking. Replacing GRPO with an extended SFT stage (*w/o*-RL), while keeping the same training-epoch budget, results in a substantial performance drop. This is evident in the spatially constrained MAZE domain and the interaction-rich SOKOBAN domain, where detecting invalid moves is critical and multiple player–block interactions frequently occur. These results underscore the advantage of GRPO over SFT in reasoning under complex constraints and rich interaction dynamics. Finally, after further removing transition reasoning chains from *w/o*-RL, *w/o*-RL-C yields a marginal decrease, indicating that SFT alone is insufficient to fully exploit transition reasoning chains without RL-based reward optimization.

Size	Method	Maze		FrozenLake		Sokoban		Pacman		Gather	
		ID	OOD	ID	OOD	ID	OOD	ID	OOD	ID	OOD
4	SVoT_p	-	-	83.3	68.3	78.3	68.3	70.0	66.7	50.0	16.7
	w/o-V	-	-	73.3	63.3	58.3	43.3	65.0	53.3	35.0	3.3
	w/o-RL	-	-	78.3	65.0	35.0	25.0	61.7	48.3	23.3	8.3
	w/o-RL-C	-	-	73.3	60.0	30.0	26.7	58.3	46.7	23.3	6.7
	MVoT	-	-	46.7	56.7	15.0	3.3	56.7	46.7	13.3	3.3
5	SVoT_p	80.0	41.7	86.7	58.3	71.7	53.3	66.7	60.0	3.3	1.7
	w/o-V	68.3	35.0	70.0	53.3	53.3	40.0	61.7	50.0	0.0	0.0
	w/o-RL	33.3	28.3	65.0	50.0	33.3	21.7	55.0	46.7	3.3	1.7
	w/o-RL-C	36.7	23.3	50.0	36.7	23.3	16.7	53.3	41.7	1.7	0.0
	MVoT	26.7	26.7	43.3	36.7	10.0	3.3	56.7	38.3	3.3	0.0
6	SVoT_p	81.7	36.7	83.3	55.0	68.3	48.3	60.0	56.7	6.7	1.7
	w/o-V	70.0	25.0	63.3	45.0	45.0	26.7	55.0	45.0	0.0	0.0
	w/o-RL	38.3	16.7	51.7	38.3	26.7	16.7	51.7	40.0	6.7	1.7
	w/o-RL-C	33.3	16.7	40.0	28.3	20.0	16.7	50.0	30.0	3.3	0.0
	MVoT	26.7	3.3	40.0	20.0	6.7	6.7	45.0	26.7	0.0	0.0

Table 4: Ablation study of SVoT_p evaluated on goal coverage accuracy, using the same settings as Table 1. *w/o*-V denotes SVoT_p without visualizations; *w/o*-RL denotes SVoT_p without GRPO (trained using only SFT for 10 epochs); and *w/o*-RL-C represents SFT with only state representations and visualizations without transition reasoning chains. See Appendix A.9 for the whole table.

Efficiency Analysis. We further evaluate the inference efficiency of SVoT_p under the OOD and free-response settings. As shown in Table 5, SVoT_p achieves an absolute improvement of 26.3% in goal coverage accuracy over MVoT, with a 14.4% increase in token usage and a 19.0% increase in inference latency per action. Visualization generation accounts for 90.6% of the total inference latency in SVoT_p, constituting the primary computational bottleneck. Removing intermediate visualizations from SVoT_p (*w/o*-V) reduces inference latency by 156.2 seconds per action, while resulting in an absolute decrease of 10.8% in goal coverage accuracy. These results indicate that intermediate visualizations improve sequential state tracking at the expense of additional computational cost. Furthermore, inference cost increases approximately linearly with trajectory length.

Method	Acc. (%)	Tokens/action	Latency/action
SVoT_p	**44.4**	1183.0	172.5 s
MVoT	18.1	1033.7	145.0 s
w/o-V	33.7	103.6	16.3 s

Table 5: Goal coverage accuracy (Acc., %) and inference cost under the OOD and free-response settings, averaged across the domains and grid sizes evaluated in Table 1. Inference cost is reported per action.

Reward Analysis. Figure 6 illustrates the validation reward curves of SVoT_p and SVoT_o for single-step prediction on SOKOBAN and GATHER with grid size 5, including the state reward r_z, visual reward r_v, and reasoning reward r_c. Under SVoT_o, the total reward is $0.5\,r_z + 0.5\,r_v$; under SVoT_p, the total reward averages $0.5\,r_c + 0.5\,r_z$ and $0.5\,r_c + 0.5\,r_v$. Overall, SVoT with PRM yields faster and stable optimization than ORM, due to its denser, reasoning-aware supervision. In SOKOBAN, SVoT_p reaches peak total reward at around epoch 3, while SVoT_o attains a comparable level around epoch 4. For the challenging GATHER domain, the lower peak total reward under SVoT_p (0.87) reflects its higher task complexity. Under SVoT_o, the state reward r_z and visual reward r_v diverge post-peak, suggesting a decoupling between textual and visual representations, where the model appears to optimize textual patterns independently of visual consistency.

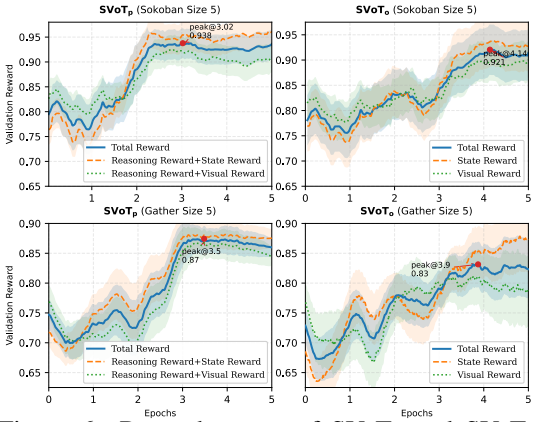


Figure 6: Reward curves of SVoT_p and SVoT_o on the SOKOBAN and GATHER validation sets for single-step prediction (876 and 742 samples, respectively). Shaded regions indicate standard deviation.

6 Conclusion

In this work, we introduced SVoT, an RL-based framework that generates interleaved, verifiable intermediate states and visualizations and treats state transitions as explicit reasoning steps to improve multi-hop spatial reasoning in MLLMs. This design bridges perception-driven spatial understanding with deterministic transition reasoning for reliable sequential state tracking. By establishing five grid-based domains that support precise verification of intermediate-state and transition-reasoning correctness, we enabled fine-grained reward optimization. Our results show that accurate intermediate-state prediction and faithful transition reasoning are central to reliable sequential state tracking, with transition-aware supervision yielding state-of-the-art performance across all evaluated domains.

7 Acknowledgements

This research was supported by the University of Melbourne's Research Computing Services and the Petascale Campus Initiative. Chao Lei was supported by a Melbourne Research Scholarship awarded by the University of Melbourne. Markus Hiller was supported by the Australian Research Council (ARC) under grant DP230102775.

References

[1] Qiucheng Wu, Handong Zhao, Michael Saxon, Trung Bui, William Yang Wang, Yang Zhang, and Shiyu Chang. Vsp: Diagnosing the dual challenges of perception and reasoning in spatial planning tasks for mllms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2270–2280, 2025.

[2] Wanyue Zhang, Yibin Huang, Yangbin Xu, JingJing Huang, Helu Zhi, Shuo Ren, Wang Xu, and Jiajun Zhang. Why do mllms struggle with spatial understanding? a systematic analysis from data to architecture. *arXiv preprint arXiv:2509.02359*, 2025.

[3] Yanyuan Qiao, Haodong Hong, Wenqi Lyu, Dong An, Siqi Zhang, Yutong Xie, Xinyu Wang, and Qi Wu. Navbench: Probing multimodal large language models for embodied navigation. In *Proceedings of the Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[4] Yunzhe Xu, Yiyuan Pan, Zhe Liu, and Hesheng Wang. Flame: Learning to navigate with multimodal llm in urban environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 9005–9013, 2025.

[5] Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.

[6] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

[7] Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. Dolphins: Multimodal language model for driving. In *European Conference on Computer Vision*, pages 403–420. Springer, 2024.

[8] Wenshan Wu, Shaoguang Mao, Yadong Zhang, Yan Xia, Li Dong, Lei Cui, and Furu Wei. Mind's eye of LLMs: Visualization-of-thought elicits spatial reasoning in large language models. In *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[9] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024.

[10] Fangyu Liu, Guy Emerson, and Nigel Collier. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 11:635–651, 2023.

[11] Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Sharon Li, and Neel Joshi. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. In *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems*, volume 37, pages 75392–75421, 2024.

[12] Chengzu Li, Caiqi Zhang, Han Zhou, Nigel Collier, Anna Korhonen, and Ivan Vulić. TopViewRS: Vision-language models as top-view spatial reasoners. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1786–1807, 2024.

[13] Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025.

[14] Michael Igorevich Ivanitskiy, Rusheb Shah, Alex F Spies, Tilman Räuker, Dan Valentine, Can Rager, Lucia Quirke, Chris Mathwin, Guillaume Corlouer, Cecilia Diniz Behn, et al. A configurable library for generating and manipulating maze datasets. *arXiv preprint arXiv:2309.10498*, 2023.

[15] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[16] Solomon W Golomb. Tiling with polyominoes. *Journal of Combinatorial Theory*, 1(2):280–296, 1966.

[17] Nima Mohajerin and Mohsen Rohani. Multi-step prediction of occupancy grid maps with recurrent neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10600–10608, 2019.

[18] Jiayang Liu, Xieyuanli Chen, Junhao Xiao, Sichao Lin, Zhiqiang Zheng, and Huimin Lu. Hybrid map-based path planning for robot navigation in unstructured environments. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2216–2223. IEEE, 2023.

[19] Roni Stern, Nathan Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, TK Kumar, et al. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the international symposium on combinatorial search*, volume 10, pages 151–158, 2019.

[20] Oren Salzman and Roni Stern. Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1711–1715, 2020.

[21] Roger N Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971.

[22] Edward C Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4):189, 1948.

[23] Ethan Chern, Jiadi Su, Yan Ma, and Pengfei Liu. Anole: An open, autoregressive, native large multimodal models for interleaved image-text generation. *arXiv preprint arXiv:2407.06135*, 2024.

[24] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

[25] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[26] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of the Twelfth International Conference on Learning Representations*, 2024.

[27] Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

[28] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

[29] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning, 2020.

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the Seventh International Conference on Learning Representations*, 2019.

[31] OpenAI. Hello gpt-4o. *OpenAI*, 2024. URL `https://www.openai.com/index/hello-gpt-4o`. Accessed: 2026-04-18.

[32] Jianrui Zhang, Mu Cai, Tengyang Xie, and Yong Jae Lee. CounterCurate: Enhancing physical and semantic visio-linguistic compositional reasoning via counterfactual examples. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15481–15495, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[33] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[34] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *Proceedings of Thirty-Eighth Advances in Neural Information Processing Systems*, volume 37, 2024.

[35] Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. In *Proceedings of the Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[36] Yufeng Cui, Honghao Chen, Haoge Deng, Xu Huang, Xinghang Li, Jirong Liu, Yang Liu, Zhuoyan Luo, Jinsheng Wang, Wenxuan Wang, et al. Emu3. 5: Native multimodal models are world learners. *arXiv preprint arXiv:2510.26583*, 2025.

[37] Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved native unified multimodal models. *arXiv preprint arXiv:2506.15564*, 2025.

[38] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[39] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[40] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[41] Yanbei Jiang, Chao Lei, Yihao Ding, Krista Ehinger, and Jey Han Lau. Propa: Toward process-level optimization in visual reasoning via reinforcement learning. *arXiv preprint arXiv:2511.10279*, 2025.

[42] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *arXiv preprint arXiv:2503.17352*, 2025.

[43] 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.

[44] Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1859–1869, 2025.

[45] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[46] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Proceedings of the Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[47] Tom Silver and Rohan Chitnis. Pddlgym: Gym environments from pddl problems. *arXiv preprint arXiv:2002.06432*, 2020.

[48] Patrik Haslum, Nir Lipovetzky, Daniele Magazzeni, Christian Muise, Ronald Brachman, Francesca Rossi, and Peter Stone. *An introduction to the planning domain definition language*, volume 13. Springer, 2019.

A Appendix

A.1 Related Work

Static Spatial Reasoning. Despite significant advancements in Large Language Models (LLMs), Vision-Language Models (VLMs), and Multimodal Large Language Models (MLLMs), spatial reasoning remains a persistent challenge and a growing area of interest [10]. Previous research attributes the limited spatial capabilities of current models to a scarcity of explicit 3D and 2D spatial knowledge in training data. To address this, recent works have introduced comprehensive data generation pipelines [32, 9, 33, 34] and datasets [11, 12] to enable large-scale training on spatially-aware Visual Question Answering (VQA) tasks. Beyond data augmentation strategies, recent architectural advancements have focused on enhancing static spatial perception. SpatialRGPT [34] introduces a relative-depth injection module and region proposals to ground objects in 3D space, while Spatial-MLLM [35] leverages a dual-encoder architecture to recover implicit structural priors from 2D inputs without external depth data. However, these approaches primarily target instantaneous spatial understanding, localizing objects and inferring relations within a single, static frame. They rarely model environment dynamics during action execution, limiting their effectiveness for complex multi-hop spatial reasoning.

Dynamic Spatial Reasoning. To enhance dynamic spatial reasoning in LLMs, Wu et al. [8] proposed Visualization-of-Thought (VoT), a zero-shot prompting strategy that encourages models to internally simulate spatial layouts and their evolution over time. Inspired by human mental navigation, VoT prompts the model to generate a text-based state description after each reasoning step to guide multi-step inference. Recent strides in multimodal-native foundation models, such as Chameleon [24], Emu [36], and Show-o [37], have expanded capabilities from passive understanding to active multimodal generation. Building on the unified multimodal backbone, Multimodal Visualization-of-Thought (MVoT) [13] advances VoT by moving beyond purely textual reasoning. By synthesizing visual tokens directly within the reasoning chain, MVoT integrates text and vision into an interleaved reasoning format. This provides a visually grounded reasoning trace, making the intermediate decision-making process more explicit. However, a key limitation of both VoT and MVoT is the absence of a principled mechanism to verify intermediate steps. As a result, errors can accumulate over time, undermining reliable sequential state tracking in multi-hop spatial reasoning. In contrast, SVoT introduces verifiable structured state representations and explicit transition reasoning chains, allowing intermediate states and reasoning chains to be optimized via reinforcement learning. By tightly coupling transition reasoning, state prediction, and visualization generation, SVoT mitigates error propagation and improves the consistency of long-horizon state tracking.

Reinforcement Learning for Reasoning. Reinforcement learning (RL) has emerged as an effective approach for enhancing reasoning in large models, as demonstrated by OpenAI o1 [38], DeepSeek-R1 [39], and Qwen3 [40]. Among recent algorithms, Group Relative Policy Optimization (GRPO) [25] provides a computationally efficient approach by normalizing rewards over a group of sampled candidate outputs, yielding stable learning signals without requiring a separate critic model. This paradigm has recently been extended to multimodal applications [41–44]. Building on these advances, SVoT adapts GRPO to improve sequential state tracking in multi-hop spatial reasoning by optimizing an interleaved generation process that produces structured intermediate states and visualizations guided by transition reasoning chains. With fine-grained rewards, SVoT verifies predicted intermediate states and transition reasoning chains while evaluating visualization fidelity, achieving state-of-the-art performance on challenging sequential state tracking tasks with strong out-of-distribution (OOD) robustness.

A.2 MVoT

The supervised fine-tuning (SFT) stage of SVoT follows the same training pipeline as MVoT. MVoT builds on Chameleon's unified autoregressive framework [24], where image and text tokens are jointly processed by a single causal Transformer. Images are encoded by a frozen visual tokenizer with codebook $C \in \mathbb{R}^{N \times D}$, where N denotes the number of codebook entries and D denotes the embedding dimensionality. We denote the i-th visual codebook index as $t_i^{\text{vis}} \in \{1, \ldots, N\}$ and the embedding associated with index j as $e_j^{\text{vis}} \in \mathbb{R}^D$. Text is mapped to discrete token indices by a separate frozen text tokenizer, and the resulting text and image token sequences are concatenated before being fed into the Transformer.

In addition to the standard next-token cross-entropy objective, MVoT incorporates a *token discrepancy loss* that explicitly regularizes the model in the visual embedding space. For each ground-truth visual index t_i^{vis}, MVoT first computes the distance between its corresponding embedding $e_{t_i^{\text{vis}}}^{\text{vis}}$ and all visual codebook entries to obtain

$$S_{t_i^{\text{vis}}} = \left[\text{MSE}(e_{t_i^{\text{vis}}}^{\text{vis}}, e_1^{\text{vis}}), \ldots, \text{MSE}(e_{t_i^{\text{vis}}}^{\text{vis}}, e_N^{\text{vis}})\right] \in \mathbb{R}^{1\times N}, \tag{11}$$

where $\text{MSE}(\cdot,\cdot)$ denotes the mean squared error and larger values indicate that a codebook entry is farther from the ground-truth visual embedding. At position i, the Transformer outputs a probability distribution $P(t_i) \in \mathbb{R}^{1\times N}$ over the visual vocabulary. The token discrepancy loss is then defined as

$$\mathcal{L}_D = \sum_{i=1}^{N} S_{t_i^{\text{vis}}} \cdot P(t_i), \tag{12}$$

where $\cdot$ denotes the inner product. Intuitively, $\mathcal{L}_D$ discourages the model from assigning high probability to codebook entries that are associated with large distances to the ground-truth embedding in the visual space.

The causal Transformer is fine-tuned with a combination of the standard next-token cross-entropy loss $\mathcal{L}_C$ over both text and image tokens and the token discrepancy loss $\mathcal{L}_D$ on image tokens, while the image and text tokenizers remain frozen. The overall training objective is

$$\mathcal{L} = \mathcal{L}_C + \mathcal{L}_D. \tag{13}$$

A.3 Detailed Formulation of Visual Reward

The visual reward r_v is defined as

$$r_v(\hat{v}, v) = r_{\text{str}}(\hat{v}, v) \cdot r_{\text{qua}}(\hat{v}, v), \tag{14}$$

$$r_{\text{str}}(\hat{v}, v) = \sum_{(i,j)} w_{ij}\, \mathbb{I}\left[\mathcal{S}(\hat{v}_{ij}, v_{ij}) \geq \tau\right], \tag{15}$$

$$r_{\text{qua}}(\hat{v}, v) = \text{clip}\left(\frac{\phi(\hat{v};\Omega)}{\phi(v;\Omega)+\epsilon}, 0, 1\right). \tag{16}$$

The term r_{str} evaluates cell-level matching, while r_{qua} measures perceptual quality through foreground sharpness and penalizes blurry predictions. Foreground cells are identified using a variance-based heuristic. For each cell, RGB values are normalized to the range $[0, 1]$, and the cell is classified as foreground if the average per-channel variance of the normalized RGB values exceeds a threshold $\delta = 0.002$. The foreground mask Ω is extracted from the ground-truth image v and kept fixed when evaluating both $\hat{v}$ and v. It is used both to assign higher weights to semantically salient cells in r_{str} and to define the sharpness region in r_{qua}.

In the structural term r_{str}, a cell is counted as matched only when its composite similarity $\mathcal{S}$ exceeds the threshold $\tau = 0.8$, making the matching criterion relatively strict and reducing reward for coarse but inaccurate alignments. Before normalization, foreground cells are assigned weights ten times larger than those of background cells, i.e.,

$$w_{ij} \propto \begin{cases} 10, & (i,j) \in \Omega, \\ 1, & (i,j) \notin \Omega, \end{cases} \qquad \text{with} \qquad \sum_{i,j} w_{ij} = 1. \tag{17}$$

We next detail the calculation of the composite similarity score $\mathcal{S}(\hat{v}_{ij}, v_{ij})$. For notational brevity, let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{H_c \times W_c \times 3}$ denote the pixel arrays of the generated cell $\hat{v}_{ij}$ and the ground-truth cell v_{ij}, respectively, where H_c and W_c are the height and width of each cell patch. $\mathcal{S}$ comprises three components:

Perceptual Hash Similarity (s_{hash}). This component captures the coarse geometric structure of the image patch. We first convert the patches $\mathbf{x}$ and $\mathbf{y}$ to grayscale and downsample them to a fixed size of $K \times K$ (with $K = 8$). Let $\mathbf{H}(\mathbf{x}) \in \{0,1\}^{K^2}$ be the binary hash vector, where the k-th bit is set to 1 if the pixel intensity exceeds the patch's mean intensity, and 0 otherwise. The similarity is

calculated using the normalized Hamming distance:

$$s_{\text{hash}}(\mathbf{x}, \mathbf{y}) = 1 - \frac{1}{K^2} \sum_{k=1}^{K^2} \mathbb{I}\left[\mathbf{H}(\mathbf{x})_k \neq \mathbf{H}(\mathbf{y})_k\right], \tag{18}$$

where $\mathbb{I}[\cdot]$ is the indicator function.

Edge Alignment (s_{edge}). To ensure the alignment of local boundaries and textures, we extract edge maps using the Sobel operator. Let G_x and G_y be the horizontal and vertical gradients of the grayscale patch, computed via convolution. We define a binary edge map $\mathbf{E}(\mathbf{x})$ by thresholding the gradient magnitude:

$$\mathbf{E}(\mathbf{x}) = \mathbb{I}\left[\sqrt{G_x^2 + G_y^2} > \mu_{\mathbf{x}}\right], \tag{19}$$

where $\mu_{\mathbf{x}}$ is the mean gradient magnitude of the patch. The similarity s_{edge} is computed as the Intersection over Union (IoU) of the binary edge maps:

$$s_{\text{edge}}(\mathbf{x}, \mathbf{y}) = \begin{cases} 1, & \text{if } |\mathbf{E}(\mathbf{x}) \cup \mathbf{E}(\mathbf{y})| = 0, \\ \dfrac{|\mathbf{E}(\mathbf{x}) \cap \mathbf{E}(\mathbf{y})|}{|\mathbf{E}(\mathbf{x}) \cup \mathbf{E}(\mathbf{y})| + \epsilon}, & \text{otherwise.} \end{cases} \tag{20}$$

where ϵ is a small constant for numerical stability. When both edge maps are empty, we set $s_{\text{edge}} = 1$, treating the shared absence of edges as perfect edge-level agreement.

Color Consistency (s_{color}). To capture chromatic correlations, we compute a joint RGB histogram. We discretize the color space into $8 \times 8 \times 8$ bins and flatten it into a vector $\mathbf{h} \in \mathbb{R}^{512}$. Let $\tilde{\mathbf{h}} = \mathbf{h} - \bar{h}\mathbf{1}$ denote the mean-centered histogram vector, where $\bar{h}$ is the mean histogram value and $\mathbf{1}$ is the all-ones vector. The similarity is measured using the Pearson correlation coefficient, formulated as the cosine similarity of the centered vectors:

$$s_{\text{color}}(\mathbf{x}, \mathbf{y}) = \text{clip}\left(\frac{\tilde{\mathbf{h}}_{\mathbf{x}}^\top \tilde{\mathbf{h}}_{\mathbf{y}}}{\|\tilde{\mathbf{h}}_{\mathbf{x}}\|_2 \cdot \|\tilde{\mathbf{h}}_{\mathbf{y}}\|_2}, 0, 1\right). \tag{21}$$

The composite similarity score $\mathcal{S}$ is computed as a weighted combination of the s_{hash}, s_{edge}, and s_{color}, with $\lambda_h = 0.4$, $\lambda_e = 0.3$, and $\lambda_c = 0.3$, respectively:

$$\mathcal{S}(\mathbf{x}, \mathbf{y}) = \lambda_h s_{\text{hash}}(\mathbf{x}, \mathbf{y}) + \lambda_e s_{\text{edge}}(\mathbf{x}, \mathbf{y}) + \lambda_c s_{\text{color}}(\mathbf{x}, \mathbf{y}). \tag{22}$$

A.4 Group Relative Policy Optimization (GRPO)

GRPO is a PPO-style policy optimization method that avoids training a separate value critic, and instead estimates advantages by normalizing rewards within a group of sampled completions. For each prompt q, GRPO samples a set of G candidate completions $\{o_g\}_{g=1}^G$ from the current (old) policy $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ and evaluates each candidate with a reward function to obtain $\{r_g\}_{g=1}^G$. The advantage of a candidate o_g is computed via within group normalization:

$$A_g = \frac{r_g - \text{mean}(\{r_j\}_{j=1}^G)}{\text{std}(\{r_j\}_{j=1}^G)}. \tag{23}$$

The policy π_θ is then updated by optimizing the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim \mathcal{D},\, \{o_g\}_{g=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{G} \sum_{g=1}^{G} \frac{1}{|o_g|} \sum_{t=1}^{|o_g|} \min\Big(\rho_{g,t}(\theta) A_g,\, \text{clip}(\rho_{g,t}(\theta), 1-\epsilon, 1+\epsilon) A_g\Big) - \beta\, D_{\text{KL}}\big(\pi_\theta \,\|\, \pi_{\text{ref}}\big) \right]. \tag{24}$$

where $\rho_{g,t}(\theta) = \frac{\pi_\theta(o_{g,t}|q, o_{g,<t})}{\pi_{\theta_{\text{old}}}(o_{g,t}|q, o_{g,<t})}$ is the token-level probability ratio, $o_{g,t}$ denotes the t-th token of completion o_g, and $|o_g|$ denotes the completion length. ϵ is the clipping parameter, and β controls

the KL penalty. By normalizing rewards within each group, GRPO derives relative advantages that provide stable training signals and encourage the policy to assign higher probability to higher-advantage completions. In our experiments, we set $\epsilon = 0.2$ and $\beta = 0$ following prior studies [45, 46].

A.5 Domain Generation

In this work, all domains are instantiated as OpenAI Gym environments [15]. FROZENLAKE is based on the built-in Gym implementation, while the other four domains are implemented using PDDLGym [47]. PDDLGym is a Python library that automatically builds OpenAI Gym environments from PDDL [48] domain and problem files. This provides a convenient and scalable mechanism for constructing relational benchmarks for both reinforcement learning and planning-based sequential decision making. In PDDL, the domain file specifies the lifted transition model through predicate symbols and parameterized action schemas, each defined by precondition and effect formulas. The problem file specifies the concrete objects used to instantiate lifted predicates into ground atoms and action schemas into ground actions, together with the initial state, represented as a set of ground atoms, and the goal condition. When constructing datasets in our experiments, we manually design a PDDL domain file for each domain and automatically generate multiple problem files, which are used to construct the training, validation, and test sets.

For the MAZE domain, we generate PDDL problem files by sampling the initial player position, goal position, and wall positions under different random seeds. We discard problem files when breadth-first search (BFS) finds no feasible path from the initial player position to the goal. For each solvable instance, we construct action sequences that include both a ground-truth, goal-achieving plan obtained via BFS for the training set and randomly sampled action sequences, which may contain inapplicable moves corresponding to wall collisions or boundary violations. The wall density for each grid size is summarized in Table 6.

For the FROZENLAKE domain, we use the built-in OpenAI Gym generator to sample distinct maps by varying the hole density as specified in Table 6. We then perform a BFS-based reachability check to discard any map whose goal state is not reachable from the initial player position. For each solvable map, we extract a goal-reaching plan via BFS for the training set and generate additional action sequences by sampling random actions under deterministic grid dynamics (i.e., ignoring the stochastic "slippery" transitions); these trajectories may terminate early when the agent falls into a hole. The hole density for each grid size is reported in Table 6.

For the SOKOBAN domain, we generate PDDL problem files by sampling the initial player position, the number of blocks, the initial and goal positions of each block, and the wall positions under different random seeds and grid sizes. Similar to the previous domains, we discard any problem file that is unsolvable and, for each solvable instance, extract a goal-achieving plan to use as supervised action sequences for training, while additionally generating random action sequences. The configuration of the number of blocks and the wall density for each grid size is reported in Table 6.

For the PACMAN domain, we generate PDDL problem files by sampling the initial player position, number of coins and their locations, controlled by the coin-density configuration. Similar to the previous domains, we discard any problem file that is unsolvable and, for each solvable instance, extract a goal-achieving plan (collecting all coins) for training, while additionally generating random action sequences. The coin density for each grid size is reported in Table 6.

For the GATHER domain, we generate PDDL problem files by sampling the initial player position, the number of balls of each color, and the positions of each ball, controlled by the ball-density configuration across different random seeds and grid sizes. We construct action sequences in the same manner as above, with the goal-achieving plan defined as the action sequence that collects all colored balls. The ball density and the color configuration for each grid size are reported in Table 6.

We present the domain descriptions of MAZE (Table 12), FROZENLAKE (Table 13), SOKOBAN (Table 14), PACMAN (Table 15), and GATHER (Table 16) used in the input prompts of SVoT with PRM, including the specified action preconditions and effects. We shorten the length of the action sequence to two for clarity of illustration. For MAZE, we include both free-response and classification examples, covering three types of responses, intermediate state prediction, image prediction, and goal prediction, to illustrate the full SVoT process. For the other domains, we only show free-response goal prediction examples.

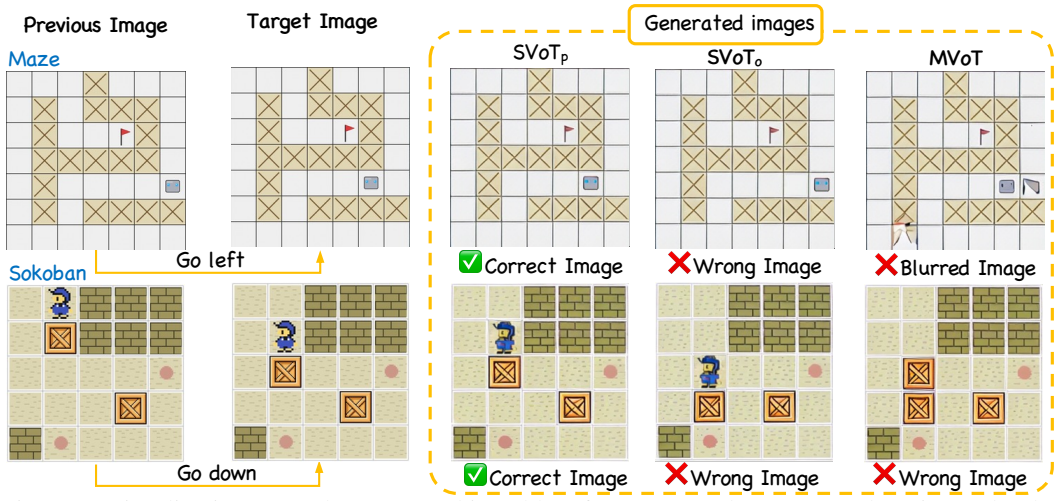


Figure 7: Visualization examples generated in Table 1 for MAZE and SOKOBAN domains under the OOD and free-response settings.

A.6 Dataset Collection

For each domain and grid size, we construct the training set from 100 valid maps with 5 valid action sequences per map, and the validation set from 30 valid maps with 2 valid action sequences per map. We further generate separate ID and OOD test sets, each from 60 valid maps with 2 valid action sequences per map. This yields 500 unique training instances, 60 unique validation instances, 120 unique ID test instances, and 120 unique OOD test instances per domain–size pair, with no duplication within or across splits. For the OOD settings, we extend the action-sequence length and introduce additional interactive objects within each grid size to evaluate SVoT's generalization capability. Table 6 reports training, validation, ID test, and OOD test sizes measured by the total number of state, visualization, and goal predictions, together with action-sequence length ranges and domain statistics for each grid size.

A.7 Candidate Answer Construction

For each instance in the introduced domain, we construct a four-choice classification set by pairing the ground-truth answer with three randomly generated incorrect candidates. We additionally impose constraints to ensure that one incorrect candidate is partially correct. For example, in PACMAN, we include a distractor that either has the correct final player position but an incorrect number of collected coins, or has an incorrect final position but a correct coin count. We also constrain candidate coordinates to lie within a distance of 2 cells from the ground-truth coordinate, and require all candidates to be valid (e.g., the predicted coin count must be between 0 and the total number of coins).

A.8 Visualization Quantitative Analysis

Table 7 provides a more fine-grained breakdown of the visual generation quality reported in Table 3 across different grid sizes. Overall, SVoT$_p$ maintains strong visual fidelity across evaluated domains and grid sizes, and in SOKOBAN at size 6, it achieves a particularly high visual reward r_v of 0.8834, where the other methods exhibit noticeable degradation. This supports the effectiveness of transition-aware supervision in mitigating visual hallucinations in multi-object interactive environments. However, for FROZENLAKE with size 4, SVoT$_p$ exhibits slightly lower visual quality compared to SVoT$_o$. Finally, visual reward scores for all methods drop markedly in the GATHER domain at grid size 6, highlighting clear direction for future work.

We additionally report three foreground–background correctness metrics in Table 8. These metrics evaluate whether generated visualizations preserve object-level correctness in grid cells. Foreground Cell Accuracy measures whether cells containing ground-truth foreground objects are correctly generated, while Background Error Rate measures hallucinated foreground objects in cells that are

Category	Domain	Grid Size	Training Size	Validation Size	Test Size		Color	Variables	ID → OOD
					ID	OOD			
Statics	MAZE	5	6342	832	1658	2682	-	Wall Density	$0.3 \rightarrow 0.5$
		6	8164	982	1956	3164	-		
		7	8544	1166	2326	3372	-		
	FROZENLAKE	4	5120	736	1494	2362	-	Hole Density	$0.2 \rightarrow 0.4$
		5	7346	882	1758	2588	-		
		6	7754	1042	2028	3018	-		
	SOKOBAN	4	5326	762	1534	2430	-	Wall Density Block Number	$0.2 \rightarrow 0.3$ $1\text{–}2 \rightarrow 2$
		5	7146	936	1864	2704	-	Wall Density	$0.2 \rightarrow 0.3$
		6	8362	1112	2216	3136		Block Number	$1\text{–}2 \rightarrow 3$
	PACMAN	4	5342	652	1308	2396	-	Coin Density	$0.3 \rightarrow 0.5$
		5	7426	846	1698	2642	-		
		6	7728	934	1910	2974	-		
	GATHER	4	5424	696	1412	2512	{R, G}	Ball Density	$0.3 \rightarrow 0.5$
		5	7330	802	1606	2670	{R, G, B}		
		6	8182	1026	2022	2956	{R, G, B, Y}		
Action	All	4	–	–	–	–	-	Sequence Length	$3\text{–}7 \rightarrow 8\text{–}10$
		5	–	–	–	–	-	Sequence Length	$4\text{–}8 \rightarrow 9\text{–}11$
		6	–	–	–	–	-	Sequence Length	$5\text{–}10 \rightarrow 11\text{–}13$
		7	–	–	–	–	-	Sequence Length	$6\text{–}11 \rightarrow 12\text{–}14$

Table 6: Domain statistics and ID→OOD parameter shifts. Training, validation, ID test, and OOD test sizes are measured by the total number of state, visualization, and goal predictions from 500 training instances, 60 validation instances, 120 ID test instances, and 120 OOD test instances for each domain–size pair, respectively. Each action step contributes one state prediction and one visualization prediction, and each instance contributes one final goal prediction.

Method	Maze			FrozenLake			Sokoban			Pacman			Gather		
	5	6	7	4	5	6	4	5	6	4	5	6	4	5	6
MVoT	0.8111	0.8030	0.8028	0.8329	0.8475	0.8278	0.8064	0.8075	0.7748	0.8064	0.8033	0.7766	0.7646	0.7590	0.7005
SVoT_o	0.8385	0.8265	0.8080	**0.8594**	0.8393	0.8406	0.8397	0.8208	0.8109	0.8219	0.8217	0.8011	0.7713	0.7687	**0.7047**
SVoT_p	**0.8748**	**0.8610**	**0.8356**	0.8560	**0.8679**	**0.8490**	**0.8486**	**0.8413**	**0.8834**	**0.8397**	**0.8444**	**0.8506**	**0.8190**	**0.7939**	0.7020

Table 7: Full quantitative analysis of generated images in Table 1 under OOD and free-response settings. Scores are computed using the visual reward r_v defined in Eq. (7)

background in the ground-truth state. We further report Full Visualization Accuracy, which counts a visualization as correct only when all ground-truth foreground cells are correctly generated and no ground-truth background cell contains an extra foreground object. These diagnostic metrics are enabled by our PDDL-based domain construction, which provides ground-truth cell-level object labels for each image. In broader applications, such semantic labels are often unavailable; therefore, the visual reward r_v, which uses a variance-based foreground heuristic, is broadly applicable.

Table 8 shows that SVoT_p consistently achieves higher Foreground Cell Accuracy and Full Visualization Accuracy than MVoT and SVoT_o, while also reducing Background Error Rate in most settings. Together with the reward-based analysis in Table 7, these results indicate that SVoT_p improves both visual fidelity and object-level correctness in generated visualizations.

A.9 Detailed Ablation Study

Table 9 extends Table 4 by additionally reporting classification results for the variants of SVoT_p. While classification performance across variants broadly tracks the trends observed in the free-response setting, the classification metric is noticeably less sensitive to the removal of RL in certain domains. For example, in the MAZE domain with size 6, *w/o*-RL attains high classification accuracy (83.3%),

Method	Metric	Maze			FrozenLake			Sokoban			Pacman			Gather		
		5	6	7	4	5	6	4	5	6	4	5	6	4	5	6
MVoT	FGAcc	87.10	85.90	84.80	88.40	89.20	87.60	85.60	84.90	82.10	86.40	85.80	83.10	80.90	79.60	73.90
	BkgErr	6.60	7.10	7.70	5.90	5.40	6.20	7.80	8.20	9.50	7.30	7.70	9.00	10.70	11.30	14.10
	FullAcc	66.70	62.50	58.30	68.30	70.00	65.80	60.80	58.30	53.30	62.50	60.80	55.00	46.70	43.30	32.50
$SVoT_o$	FGAcc	90.30	88.80	86.20	91.40	89.50	89.30	89.60	87.30	86.10	88.40	87.70	85.80	82.80	81.50	74.80
	BkgErr	4.90	5.40	6.60	4.20	5.10	5.20	5.60	6.80	7.30	5.90	6.20	7.50	9.10	9.80	**13.20**
	FullAcc	75.00	70.80	64.20	78.30	73.30	72.50	70.00	65.00	62.50	69.20	67.50	62.50	54.20	50.00	36.70
$SVoT_p$	FGAcc	**94.10**	**93.20**	**90.60**	**92.00**	**92.50**	**91.60**	**91.80**	**91.10**	**93.20**	**90.40**	**90.80**	**91.10**	**86.80**	**84.30**	**76.10**
	BkgErr	**2.90**	**3.30**	**4.50**	**4.00**	**3.60**	**4.30**	**4.30**	**4.80**	**3.40**	**4.80**	**4.70**	**4.30**	**6.70**	**8.00**	13.60
	FullAcc	**84.20**	**81.70**	**75.80**	**80.00**	**81.70**	**78.30**	**77.50**	**75.80**	**83.30**	**74.20**	**75.00**	**76.70**	**63.30**	**57.50**	**38.30**

Table 8: Foreground–background correctness evaluation of generated visualizations in Table 1 under OOD and free-response settings. FGAcc denotes Foreground Cell Accuracy, BkgErr denotes Background Error Rate, and FullAcc denotes Full Visualization Accuracy. Higher values are better for FGAcc and FullAcc, while lower values are better for BkgErr. All results are reported as percentages.

Size	**Method**	**Maze**				**FrozenLake**				**Sokoban**				**Pacman**				**Gather**			
		ID		OOD		ID		OOD		ID		OOD		ID		OOD		ID		OOD	
		C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F	C	F
4	$SVoT_p$	-	-	-	-	100.0	83.3	88.3	68.3	86.7	78.3	80.0	68.3	100.0	70.0	93.3	66.7	60.0	50.0	46.7	16.7
	w/o-V	-	-	-	-	90.0	73.3	80.0	63.3	78.3	58.3	63.3	43.3	80.0	65.0	76.7	53.3	56.7	35.0	40.0	3.3
	w/o-RL	-	-	-	-	100.0	78.3	81.7	65.0	73.3	35.0	68.3	25.0	85.0	61.7	80.0	48.3	55.0	23.3	43.3	8.3
	w/o-RL-C	-	-	-	-	100.0	73.3	81.7	60.0	70.0	30.0	63.3	26.7	78.3	58.3	66.7	46.7	45.0	23.3	43.3	6.7
	MVoT	-	-	-	-	86.7	46.7	80.0	56.7	73.3	15.0	61.7	3.3	73.3	56.7	63.3	46.7	46.7	13.3	36.7	3.3
	T-CoT	-	-	-	-	75.0	33.3	66.7	33.3	65.0	5.0	53.3	0.0	71.7	53.3	50.0	36.7	41.7	6.7	36.7	0.0
5	$SVoT_p$	93.3	80.0	70.0	41.7	91.7	86.7	85.0	58.3	78.3	71.7	73.3	53.3	93.3	66.7	83.3	60.0	46.7	3.3	33.3	1.7
	w/o-V	90.0	68.3	65.0	35.0	81.7	70.0	73.3	53.3	76.7	53.3	65.0	40.0	76.7	61.7	73.3	50.0	33.3	0.0	25.0	0.0
	w/o-RL	90.0	33.3	63.3	28.3	81.7	65.0	73.3	50.0	70.0	33.3	66.7	21.7	80.0	55.0	70.0	46.7	35.0	3.3	26.7	1.7
	w/o-RL-C	86.7	36.7	60.0	23.3	78.3	50.0	60.0	36.7	63.3	23.3	55.0	16.7	73.3	53.3	66.7	41.7	30.0	1.7	21.7	0.0
	MVoT	83.3	26.7	56.7	26.7	73.3	43.3	53.3	36.7	61.7	10.0	55.0	3.3	73.3	56.7	53.3	38.3	33.3	3.3	30.0	0.0
	T-CoT	71.7	16.7	46.7	16.7	73.3	33.3	43.3	26.7	60.0	3.3	50.0	0.0	71.7	51.7	43.3	31.7	31.7	0.0	25.0	0.0
6	$SVoT_p$	90.0	81.7	66.7	36.7	93.3	83.3	76.7	55.0	71.7	68.3	65.0	48.3	83.3	60.0	75.0	56.7	45.0	6.7	30.0	1.7
	w/o-V	86.7	70.0	58.3	25.0	80.0	63.3	68.3	45.0	61.7	45.0	56.7	26.7	75.0	55.0	66.7	45.0	43.3	0.0	23.3	0.0
	w/o-RL	83.3	38.3	61.7	16.7	73.3	51.7	66.7	38.3	66.7	26.7	51.7	16.7	70.0	51.7	63.3	40.0	40.0	6.7	21.7	1.7
	w/o-RL-C	83.3	33.3	53.3	16.7	71.7	40.0	53.3	28.3	56.7	20.0	53.3	16.7	66.7	50.0	60.0	30.0	41.7	3.3	18.3	0.0
	MVoT	83.3	26.7	46.7	3.3	63.3	40.0	43.3	20.0	58.3	6.7	51.7	6.7	60.0	45.0	51.7	26.7	40.0	0.0	20.0	0.0
	T-CoT	70.0	16.7	36.7	1.7	50.0	36.7	33.3	13.3	58.3	3.3	41.7	3.3	48.3	36.7	40.0	28.3	38.3	0.0	23.3	0.0
7	$SVoT_p$	80.0	70.0	65.0	33.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	w/o-V	78.3	50.0	48.3	21.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	w/o-RL	78.3	33.3	48.3	13.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	w/o-RL-C	73.3	26.7	46.7	6.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	MVoT	73.3	6.7	40.0	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	T-CoT	60.0	3.3	30.0	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 9: Full ablation study of $SVoT_p$ evaluated on goal coverage accuracy, using the same settings as Table 1. *w/o*-V denotes $SVoT_p$ without visualizations; *w/o*-RL denotes $SVoT_p$ without GRPO (trained using only SFT for 10 epochs); and *w/o*-RL-C represents SFT with only state representations and visualizations without transition reasoning chains.

yet its free-response score collapses dramatically (38.3% vs. 81.7%). This gap indicates that the SFT-only variant (*w/o*-RL) tends to exploit shortcuts to predict the final outcome label without truly learning the underlying reasoning dynamics. By contrast, the strong alignment between classification and free-response performance in $SVoT_p$ suggests that its gains arise from genuine spatial reasoning rather than superficial pattern matching.

A.10 Additional Baseline Comparisons

To further evaluate SVoT against recent RL-based visual reasoning methods, we introduce two additional baselines: MVoT+GRPO and an adapted version of Visual Planning (VP*) [43]. MVoT+GRPO applies GRPO to MVoT using the SVoT rewards corresponding to its textual reasoning trajectory

Method	Maze	FrozenLake	Sokoban	Pacman	Gather
$SVoT_p$	**39.2**	**56.7**	**50.8**	**58.4**	**1.7**
MVoT+GRPO	17.5	39.2	9.2	38.3	0.0
MVoT	15.0	28.4	5.0	32.5	0.0
VP*	15.0	32.5	7.5	36.7	0.0

Table 10: Goal coverage accuracy (%) under the OOD and free-response settings, averaged over grid sizes 5 and 6. MVoT+GRPO applies GRPO to MVoT using the applicable SVoT rewards, while VP* denotes Visual Planning adapted to sequential state tracking.

Method	Maze	FrozenLake	Sokoban	Pacman	Gather
$SVoT_p$	39.2	60.5	56.6	61.1	6.7
$SVoT_p$-Oracle	**45.8**	**66.7**	**64.5**	**66.7**	**9.4**
$SVoT_p$-Shuffle	25.0	43.3	23.9	34.4	0.0
w/o-V	30.0	53.9	36.7	49.4	1.1

Table 11: Goal coverage accuracy (%) under the OOD and free-response settings, averaged across the grid sizes evaluated in Table 1. Oracle replaces generated intermediate visualizations with ground-truth visualizations, while Shuffle replaces them with visualizations from other instances. *w/o*-V denotes $SVoT_p$ without intermediate visualizations.

and visualizations. For VP*, we adapt Visual Planning to sequential state tracking, where the model predicts the next-state visualization conditioned on the current-state image and prescribed action sequence. Since the original goal-progress reward of Visual Planning is designed for path-finding, we optimize VP* using GRPO with the visual reward of SVoT to supervise state transitions.

As shown in Table 10, MVoT+GRPO improves goal coverage accuracy over MVoT across most domains, suggesting the benefits of RL optimization. Nevertheless, $SVoT_p$ substantially outperforms MVoT+GRPO across all five domains, highlighting the importance of explicit intermediate state representations and transition reasoning chains in improving state tracking accuracy. Similarly, $SVoT_p$ achieves higher accuracy than VP* across all domains, further supporting the effectiveness of jointly generating and supervising structured states, transition reasoning chains, and intermediate visualizations.

A.11 Inference-Time Visualization Interventions

To further investigate whether intermediate visualizations contribute to subsequent state predictions, we conduct inference-time interventions on $SVoT_p$ without additional training. Specifically, we replace generated intermediate visualizations with either ground-truth visualizations (Oracle) or visualizations from other instances (Shuffle), while keeping the generated textual states unchanged. We additionally compare with *w/o*-V, which removes intermediate visualizations. All experiments are evaluated under the OOD and free-response settings, with results averaged across the grid sizes evaluated in Table 1.

As shown in Table 11, replacing generated intermediate visualizations with ground-truth visualizations consistently improves goal coverage accuracy across all five domains. In contrast, shuffled visualizations substantially reduce accuracy and generally underperform *w/o*-V, indicating that incorrect intermediate visualizations can adversely affect subsequent state predictions. These results suggest that $SVoT_p$ utilizes visual cues from intermediate visualizations to support sequential state tracking, rather than treating them as redundant outputs.

A.12 Additional Visualizations

Figure 7 illustrates additional visualization examples for the MAZE and SOKOBAN domains, omitted from Figure 5. In both domains, $SVoT_p$ produces consistently high-fidelity visualizations that faithfully follow the transition reasoning chain: in MAZE, the agent correctly moves one step to the left, and in SOKOBAN, the player accurately moves one step down and updates the block position

SVoT+PRM input prompts and responses in MAZE

MAZE (Free Response)

Task: Maze
Determine the positions of the player following the action sequence. Please include the reasoning process, starting with `<think>` and ending with `</think>`, to derive the positions of the player and the new image after executing the actions.
The definitions of the actions are given below.
Go up / left / down / right moves the agent by one grid cell in the corresponding absolute direction, subject to the following preconditions and effects.
Preconditions: the adjacent cell in the chosen direction is outside the grid boundary.
Effects: the agent remains at its current position (out-of-bounds movement is invalid).
Preconditions: the adjacent cell in the chosen direction is a wall.
Effects: the agent remains at its current position (movement blocked by an obstacle).
Preconditions: the adjacent cell in the chosen direction is inside the grid boundary and is not a wall (i.e., a clear cell).
Effects: the agent moves one cell in the chosen direction.
Full Action Sequence: Go right, Go right.
Coordinates are (row, column), starting at (0,0) in the top-left corner.
Initial maze: <image>
Initial player position: (2, 0).

Image prediction
Input: . . . Go right, player position: (2,1).
Response: <*think*> . . . <*/think*> <*image*>
State prediction
Input: . . . Go right, player position: (2,1). <image>
Response: <*think*> . . . <*/think*> *Go right*, *player position*: *(2,2)*.
Goal prediction
Input: . . . Go right, player position: (2,1). Go right, player position: (2,2). <image>
Response: *Action sequence finished*. *Player position*: *(2,2)*.

MAZE (Classification)

Task: Maze Navigation Simulation
...(same as free response)
Initial maze: <image>
Initial player position: (2, 0).
Return A, B, C or D.
A. Player position: (2,0).
B. Player position: (2,2).
C. Player position: (1,2).
D. Player position: (2,1).

Image prediction
Response: . . . (same as free response)
State prediction
Response: . . . (same as free response)
Goal prediction
Response: *Action sequence finished*. ***The answer is***: ***B***.

Table 12: Examples SVoT with PRM input prompts and model responses in the MAZE domain under free-response and classification settings (we omit duplicated prompts and responses for conciseness). *Italicized* text denotes the expected response.

SVoT+PRM input prompts and responses in FROZENLAKE

FROZENLAKE (Free Response)

Task: FrozenLake
Determine the position of player and the state of player (ice, hole, or goal) following the action sequence. Please include the reasoning process, starting with `<think>` and ending with `</think>`, to derive the position of the player, the state of player, and the new image after executing the actions.
The definitions of the actions are given below.
Go up / left / down / right moves the agent by one grid cell in the corresponding absolute direction, subject to the following preconditions and effects.
Preconditions: the adjacent cell in the chosen direction is ice.
Effects: the agent moves one cell in the chosen direction. The player's state remains ice.
Preconditions: the adjacent cell in the chosen direction is a hole.
Effects: the agent moves one cell into the hole and falls in. The player's state becomes hole.
Preconditions: the adjacent cell in the chosen direction is the goal.
Effects: the agent moves into the goal cell and successfully reaches the goal. The player's state becomes goal.
Full Action Sequence: Go down, Go down.
Coordinates are (row, column), starting at (0,0) in the top-left corner.
Initial maze: <image>
Initial player position: (0, 0), state: ice.

Input: ...Go down, player position: (1,0), state: ice. Go down, player position: (2,0), state: ice. <image>
Response: *Action sequence finished. player position: (2,0), state: ice.*

Table 13: Example SVoT with PRM input prompts and model responses in the FROZENLAKE domain for free-response goal prediction. *Italicized* text denotes the expected response.

accordingly. By contrast, SVoT_o generates incorrect visualizations due to erroneous movements in all examples. As in examples in Figure 5, MVoT tends to generate blurrier and less accurate visualizations.

A.13 Hyperparameters Analysis in Visual Reward

The visual reward r_v in SVoT involves several hyperparameters, including the threshold δ for foreground–background identification, the threshold τ for cell matching, the foreground weighting used to emphasize semantically salient cells, and the weights λ_h, λ_e, and λ_c assigned to s_{hash}, s_{edge}, and s_{color}, respectively, in the computation of the composite similarity score $\mathcal{S}$. Tables 17– 20 report the results of hyperparameter study for r_v in SVoT_p on the validation sets of the five introduced domains with grid sizes 5 and 6 under the free-response setting.

Table 17 indicates that $\delta = 0.002$ yields the most balanced performance across domains. While $\delta = 0.001$ remains competitive, increasing the threshold to $\delta = 0.003$ leads to a decline, suggesting that an overly restrictive foreground criterion may exclude semantically salient regions from the foreground mask, causing them to receive lower weights in reward computation. Table 18 indicates that stricter matching thresholds are beneficial, with both $\tau = 0.8$ and $\tau = 0.9$ outperforming $\tau = 0.7$, while $\tau = 0.8$ provides the most consistent overall performance. The results in Table 19 underscores the importance of foreground emphasis, as the settings that assign higher weights to foreground cells outperform the uniform baseline. The strongest results are obtained with a $10\times$ ratio, whereas increasing the ratio further to $20\times$ leads to mild degradation on several domains. Table 20 shows that the three similarity components are most effective when used jointly. Among the single-component variants, the perceptual-hash term ($\lambda_h = 1$) yields the strongest standalone performance, indicating that coarse structural correspondence provides the dominant signal for reliable cell matching, while the edge-only ($\lambda_e = 1$) and color-only ($\lambda_c = 1$) settings contribute

SVoT+PRM input prompts and responses in SOKOBAN

SOKOBAN (Free Response)

Task: Sokoban
Determine the positions of the player and block following the action sequence. Please include the reasoning process, starting with `<think>` and ending with `</think>`, to derive the positions of the player and block, and the new image after executing the actions.
The definitions of the actions are given below.
Go up / left / down / right moves the agent by one grid cell in the corresponding absolute direction, subject to the following preconditions and effects.
Preconditions: The adjacent cell in the chosen direction is a block, and the cell immediately beyond that block in the same direction is empty and within the grid.
Effects: The agent moves one cell in the chosen direction, and the block is pushed one cell in the same direction.
Preconditions: The adjacent cell in the chosen direction is empty and within the grid.
Effects: The agent moves one cell in the chosen direction, and the block positions remain unchanged.
Preconditions: The adjacent cell in the chosen direction is a wall, outside the grid, or a block whose next cell is not empty.
Effects: The agent and all blocks remain unchanged.
Full Action Sequence: Go up, Go left.
Coordinates are (row, column), starting at (0,0) in the top-left corner.
Initial maze: <image>
Initial player position: (2, 3), block position(s): (2,1), (1,2).

Input: ...Go up, player position: (1,3), block position(s): (2,1), (1,2). Go left, player position: (1,2), block position(s): (2,1), (1,1). <image>
Response: *Action sequence finished. Player position*: *(1,2)*, *block position(s)*: *(2,1)*, *(1,1)*.

Table 14: Example SVoT with PRM input prompts and model responses in the SOKOBAN domain for free-response goal prediction. *Italicized* text denotes the expected response.

complementary information. These findings are consistent with the superior performance of the combined setting $(\lambda_h, \lambda_e, \lambda_c) = (0.4, 0.3, 0.3)$.

We also conduct an ablation study on r_v in SVoT_p to evaluate the respective contributions of r_{str} and r_{qua}. The results are reported in Table 21. The term r_{str} provides the stronger standalone supervisory signal, whereas r_{qua} is less effective when used in isolation. Their combination nevertheless yields the best overall performance, indicating that both cell-level consistency, as provided by r_{str}, and perceptual quality, as captured by r_{qua}, are important for achieving strong performance.

A.14 Limitations and Future Work

SVoT introduces additional inference cost compared with outcome-only or text-only baselines, since it autoregressively generates structured intermediate states and visualizations guided by transition reasoning chains along the trajectory. This overhead is a deliberate trade-off for making sequential state tracking explicit, verifiable, and reward-optimizable. In future work, the efficiency of SVoT could be further improved through selective visualization, adaptive reasoning length, and intermediate-state caching, while preserving the benefits of explicit transition modeling.

SVoT+PRM input prompts and responses in PACMAN

PACMAN (Free Response)

Task: Pacman
Determine the position of the player and total number of coins collected by the player along the path by following the given action sequence. Please include the reasoning process, starting with `<think>` and ending with `</think>`, to derive the position of the player, the total number of coins collected, and the new image after executing the actions.
The definitions of the actions are given below.
Go up / left / down / right moves the agent by one grid cell in the corresponding absolute direction, subject to the following preconditions and effects.
Preconditions: The adjacent cell in the chosen direction contains a coin.
Effects: The agent moves one cell in the chosen direction, and the number of collected coins increases by one.
Preconditions: The adjacent cell in the chosen direction is clear (i.e., does not contain a coin).
Effects: The agent moves one cell in the chosen direction, and the number of collected coins remains unchanged.
Full Action Sequence: Go up, Go up.
Coordinates are (row, column), starting at (0,0) in the top-left corner.
Initial maze: <image>
Initial player position: (3, 3), collected coin(s): 0.

Input: ...Go up, player position: (2,3), collected coin(s): 1. Go up, player position: (1,3), collected coin(s): 2. <image>
Response: *Action sequence finished. Player position: (1,3), collected coin(s): 2.*

Table 15: Example SVoT with PRM input prompts and model responses in the PACMAN domain for free-response goal prediction. *Italicized* text denotes the expected response.

SVoT+PRM input prompts and responses in GATHER

GATHER (Free Response)

Task: Gather
Determine the position of player and the number of balls of each color collected along the path by following the given action sequence. Please include the reasoning process, starting with `<think>` and ending with `</think>`, to derive the position of the player, the count of collected balls for each color, and the new image after executing the actions.
The definitions of the actions are given below.
Go up / left / down / right with a specified step count moves the agent by that number of grid cells in the corresponding absolute direction, subject to the following preconditions and effects.
Preconditions: The next specified number of cells in the chosen direction contain balls.
Effects: The agent moves the specified number of cells in the chosen direction, collects all balls along the path (including the destination cell), and increments the collected count for each ball color accordingly.
Preconditions: The next specified number of cells in the chosen direction contain no balls (i.e., all cells are clear).
Effects: The agent moves the specified number of cells in the chosen direction, and the collected ball counts remain unchanged.
Full Action Sequence: Go right 2, Go down 2.
Coordinates are (row, column), starting at (0,0) in the top-left corner.
Initial maze: <image>
Initial player position: (0, 0), collected ball(s): red: 0, green: 0.

Input: ... Go right 2, player position: (0,2), collected ball(s): red: 1, green: 0. Go down 2, player position: (2,2), collected ball(s): red: 1, green: 2. <image>
Response: *Action sequence finished. Player position: (2,2), collected ball(s): red: 1, green: 2.*

Table 16: Example SVoT with PRM input prompts and model responses in the GATHER domain for free-response goal prediction. *Italicized* text denotes the expected response.

Size	Setting	Maze	FrozenLake	Sokoban	Pacman	Gather
5	$\delta = 0.001$	83.3	**91.7**	**81.7**	**73.3**	10.0
	$\delta = 0.002$	**85.0**	**91.7**	**81.7**	**73.3**	**11.7**
	$\delta = 0.003$	73.3	83.3	75.0	66.7	3.3
6	$\delta = 0.001$	83.3	88.3	75.0	68.3	**11.7**
	$\delta = 0.002$	**86.7**	**90.0**	**78.3**	**70.0**	10.0
	$\delta = 0.003$	76.7	73.3	68.3	66.7	0.0

Table 17: Parameter study on the foreground–background threshold δ for r_v across five domains under the free-response setting for grid sizes 5 and 6. Best results are shown in **bold**, and the adopted parameter setting for r_v is highlighted in gray.

Size	Setting	Maze	FrozenLake	Sokoban	Pacman	Gather
5	$\tau = 0.7$	73.3	80.0	70.0	60.0	1.7
	$\tau = 0.8$	**85.0**	**91.7**	**81.7**	**73.3**	**11.7**
	$\tau = 0.9$	83.3	**91.7**	80.0	**73.3**	**11.7**
6	$\tau = 0.7$	71.7	75.0	75.0	61.7	0.0
	$\tau = 0.8$	**86.7**	**90.0**	**78.3**	**70.0**	**10.0**
	$\tau = 0.9$	**86.7**	**90.0**	75.0	**70.0**	6.7

Table 18: Parameter study on the cell-matching threshold τ for r_v across five domains under the free-response setting for grid sizes 5 and 6. Best results are shown in **bold**, and the adopted parameter setting for $r_v$is highlighted in gray.

Size	Setting	Maze	FrozenLake	Sokoban	Pacman	Gather
5	1x	73.3	85.0	71.7	70.0	0.0
	10x	**85.0**	**91.7**	**81.7**	**73.3**	**11.7**
	20x	80.0	85.0	73.3	**73.3**	8.3
6	1x	78.3	81.7	68.3	65.0	0.0
	10x	**86.7**	**90.0**	**78.3**	70.0	10.0
	20x	81.7	83.3	71.7	**73.3**	**11.7**

Table 19: Parameter study on the foreground weighting ratio for r_v across five domains under the free-response setting for grid sizes 5 and 6. Best results are shown in **bold**, and the adopted parameter setting for r_v is highlighted in gray.

Size	Setting	Maze	FrozenLake	Sokoban	Pacman	Gather
5	$\lambda_h = 1$, $\lambda_e = 0$, and $\lambda_c = 0$	83.3	90.0	78.3	68.3	**11.7**
	$\lambda_h = 0$, $\lambda_e = 1$, and $\lambda_c = 0$	80.0	83.3	71.7	68.3	5.0
	$\lambda_h = 0$, $\lambda_e = 0$, and $\lambda_c = 1$	73.3	75.0	66.7	61.7	0.0
	$\lambda_h = 0.4$, $\lambda_e = 0.3$, and $\lambda_c = 0.3$	**85.0**	**91.7**	**81.7**	**73.3**	**11.7**
6	$\lambda_h = 1$, $\lambda_e = 0$, and $\lambda_c = 0$	81.7	86.7	75.0	**70.0**	8.3
	$\lambda_h = 0$, $\lambda_e = 1$, and $\lambda_c = 0$	75.0	83.3	71.7	63.3	5.0
	$\lambda_h = 0$, $\lambda_e = 0$, and $\lambda_c = 1$	73.3	71.7	61.7	58.3	0.0
	$\lambda_h = 0.4$, $\lambda_e = 0.3$, and $\lambda_c = 0.3$	**86.7**	**90.0**	**78.3**	**70.0**	**10.0**

Table 20: Parameter study on the component weights $(\lambda_h, \lambda_e, \lambda_c)$ in the composite similarity score $\mathcal{S}$ for r_v across five domains under the free-response setting for grid sizes 5 and 6. Best results are shown in **bold**, and the adopted parameter setting for r_v is highlighted in gray.

Size	Setting	Maze	FrozenLake	Sokoban	Pacman	Gather
5	$r_v = r_{\text{str}}$	83.3	90.0	71.7	66.7	6.7
	$r_v = r_{\text{qua}}$	71.7	83.3	65.0	61.7	0.0
	$r_v = r_{\text{str}} \cdot r_{\text{qua}}$	**85.0**	**91.7**	**81.7**	**73.3**	**11.7**
6	$r_v = r_{\text{str}}$	81.7	**90.0**	71.7	63.3	**11.7**
	$r_v = r_{\text{qua}}$	73.3	81.7	61.7	58.3	3.3
	$r_v = r_{\text{str}} \cdot r_{\text{qua}}$	**86.7**	**90.0**	**78.3**	**70.0**	10.0

Table 21: Ablation study of the visual reward r_v in SVoT_p on the validation sets across five domains under the free-response setting for grid sizes 5 and 6. Best results are shown in **bold**, and the adopted reward composition for r_v is highlighted in gray.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We clearly show the paper's contributions and scope in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We show the limitations in Appendix A.14.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This work does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification:All evaluated methods are clearly explained. Experimental settings are thoroughly described in the Section 5, with detailed domain generation, dataset collection and prompts illustrations in Appendix A.5 and A.6 to ensure reproducibility.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: Our code is included in the supplementary material.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Experimental settings are thoroughly described in the Section 5, with detailed domain generation and dataset collection in Appendix A.5 and A.6 to ensure reproducibility.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Figure 6 reports results with standard deviation.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We clearly describe the computational resources in Section 5.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: This work follows the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer:[N/A]

    Justification: This work does not pose any foreseeable societal impact.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This work poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We clearly cite the original papers that introduced the datasets used in this work.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [N/A]

    Justification: This work does not release new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research involving human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research involving human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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