An Emergent Symbolic Representation of Space as a Bridge Between Language and Reinforcement Learning in Continuous Environments

Ziqi MA
U2IS, ENSTA, IP-Paris*
ziqi_ma0605@163.com

Sao Mai NGUYEN
U2IS, ENSTA, IP-Paris*
nguyensmai@gmail.com

Philippe XU
U2IS, ENSTA, IP-Paris*
philippe.xu@ensta.fr

Abstract

Large Language Models (LLMs) exhibit their potential for interacting with reinforcement learning (RL) agents, for instance as high-level planners. However, space representation still impedes their application for embodied agents. We tackle this problem by taking advantage of a discrete world representation learned online by reinforcement learning. Our proposed algorithm, SGIM-STAR is a hierarchical RL method where the top-level agent is augmented with a partition-wise, learning-progress–driven switch between a RL-based planner and an LLM planner. The agent builds a discrete reachability-based partition of space online and uses intrinsic motivation to query the LLM only when beneficial, defaulting to RL as planner otherwise. This yields usage/cost efficiency: the RL-based planner dominates early and the LLM is leveraged as the representation matures. In Ant Maze, SGIM-STAR achieves the best and most stable success among STAR, LLM-only, and a non-partitioned adaptive variant, avoiding mid-training collapses while reducing LLM calls. The results demonstrate a practical fusion of LLMs taking advantage of emerging symbolic models of the environment for long-horizon tasks.

1 Introduction

Recent foundation models have shown that Large Language Models (LLMs), although trained mostly on abstract content such as from web scrapping, can carry internal world models [Shentu et al., 2024, Chakraborty et al., 2023, Jang et al., 2024, Driess et al., 2023, Ma et al., 2023, Zitkovich et al., 2023] that can be exploited for effective interaction in embodied AI to drive reinforcement learning and planning agents [Hu et al., 2019, Carta et al., 2025]. However, their performance is poor in the case of real-world inference because of the grounding problem [Jokinen, 2024]. Although training the LLMs on massive text data can endow LLMs with a representation of the world, the connection to sensory modalities of the real world is essential to solve a wider range of problems grounded in the physical world such as in robotics. Moreover, the shortcomings of LLMs hardly predictable, especially for open-ended worlds. Thus, the world models of LLMs need to be probed to identify their limitations and capabilities, so as to enhance them. In this work, we use the intrinsic motivation [Gottlieb et al., 2013, Schmidhuber, 1991] empirical measure of learning progress to assess their limitations and capabilities.

Another limitation of the use of LLMs for effective interaction within dynamic environments is the opposition of a symbolic representation of the world by LLMs and the continuous world of physical embodied environments. We need to bridge this gap by learning a symbolic representation of the environment that aligns with the LLM symbols. Thus, we propose an algorithm that creates online a discrete world representation by reinforcement learning exploration and combines it with

*828 Bd des Maréchaux, 91120 Palaiseau, France

symbols from an LLM. As open-learning tasks, we will address long-horizon tasks with a hierarchical reinforcement learning (HRL) algorithm enhanced by an LLM. To address long-horizon tasks by building discrete representations, HRL algorithms using reachability analysis have successfully solved problems as in Ant Maze environments [Zadem et al., 2023, 2024]. To further these works by taking advantage of the internal world representation of LLMs, the top-level agent of our algorithm can identify its limitation and actively choose between an LLM-driver planner and a Q-learning planner, using intrinsic motivation criteria. This combination can enhance the world representation of our system. As such, our algorithm combines an emerging symbolic representation using a partition of space output by a bottom-up process based on reinforcement learning, and symbols from an LLM as a planner to top-down exploration of the embodied environment.

2 Related Works

2.1 Space Representation for Hierarchical Reinforcement Learning

To address complex tasks which involve long-term planning and multi-step actions, HRL algorithms decompose a task into simpler subtasks, allowing them to be subsequently solved efficiently. HIRO [Nachum et al., 2018] introduced a two-level manager–controller for continuous control with off-policy data by exploiting temporal abstraction; HAC [Levy et al., 2019] complexified the architecture into multiple levels of reinforcement learning agents to address longer-horizon tasks. HRAC [Zhang et al., 2020] approximates reachability to propose subgoals in k-step adjacent regions, but computing reachability relationships in continuous high-dimensional spaces is costly.

While HAC and HIRO sample subgoals from the raw state space, recent HRL algorithms sought to solve the curse of dimensionality problem of subgoal space by learning a representation of the subgoal space. LESSON learns latent slow features to capture long-horizon dynamics [Li et al., 2021]. GARA [Zadem et al., 2023] and STAR [Zadem et al., 2024] solved the computational cost problem of HRAC by discretizing the subgoal space : they partition the state space to build reachability-aware regions and refine them based on learned k-step reachability, combining spatial abstraction with hierarchical control. While GARA used a two-level agent, STAR uses a 3-level agent to address higher-dimensional environments, showing good results in 5-D continuous state spaces, but these results can be unstable. To address long-horizon tasks in robotic real-world environments that are high-dimensional, we extend this line of work to enhance this space abstraction with complementary mechanisms to stabilize the performance.

2.2 Socially Guided Intrinsic Motivation

To address sparse-reward multi-task learning, Intrinsic Motivation (IM) [Gottlieb et al., 2013, Schmidhuber, 1991] drives the exploration by automatic curriculum learning before obtaining any non-zero external reward. Intrinsic motivation has used several measures for active learning [Oudeyer and Kaplan, 2007], such as novelty, competence Oudeyer et al. [2005] or progress [Baranes and Oudeyer, 2013]. However, IM still meets limitations in high-dimensional task spaces. To complement reinforcement learning, human-in-the-loop approaches Retzlaff et al. [2024] such as reinforcement learning from human feedback Knox et al. [2013] have been developed. Imitation learning [**?**] and inverse-RL methods [Finn et al., 2016, Fu et al., 2018] serve as computational frameworks for social guidance in robotics, by either transposing a policy from demonstrations or learning a reward signal.

Whereas most human-in-the-loop algorithms have considered the learning agent as passive in their interaction with humans, *active imitation learning* proposes algorithms for the learning agent to actively request information from teachers Shon et al. [2007], with for instance the sucessful algorithm DAGGER Ross et al. [2011]. For robots, *Socially Guided Intrinsic Motivation (SGIM)* couples IM with social guidance to enable the agent to actively decide on different aspects of its interaction with the teachers : while teachers are available to provide demonstrations of policies, goals [Nguyen and Oudeyer, 2012b], or task decomposition [Duminy et al., 2021], while the agent chooses what, when, how and whom to imitate based on learning progress [Nguyen and Oudeyer, 2012b]. This yields an adaptive curriculum that focuses its requests for demonstrations where competence improves fastest, enabling efficient exploration in high-dimensional spaces, by benefiting from implicit world models from teachers. Our work adopts the same mechanism of active choice between reinforcement learning or requesting expert help based on intrinsic motivation.

2.3 Large Language Models in Decision-Making

Recent breakthroughs in LLMs have significantly expanded their capabilities beyond natural language processing to complex reasoning and decision-making tasks. Studies have explored using LLMs as planners or controllers in robotic systems, highlighting their potential to exploit internal world representations within LLMS. Hu et al. [2019] generates a plan in natural language, which is then executed by a separate model.

However, integrating LLMs into reinforcement learning frameworks remains challenging due to poor space representation of a continuous environment, whereas LLMs use a discrete, symbolic representation. Jiang et al. [2019] uses language as the interface between high- and low-level policies in hierarchical RL, with a low-level policy that follows language instructions, and the top-level policy producing actions in the space of language. In Shentu et al. [2024], LCB uses a learnable latent code to act as a bridge between LLMs and low-level policies. This enables LLMs to flexibly communicate goals in the task plan without being entirely constrained by language limitations. To alleviate the lack of grounding of LLMs in space, these works add to the reinforcement learning agents a new layer to translate between the continuous space of states and the discrete space of LLM symbols. However, the reinforcement learning algorithms GARA and STAR learn directly a discrete representation, which symbols can be more readily used by a LLM. In this work, we explore how the emerging symbolic representation of STAR can be exploited by LLMs.

Our *contribution* is to study how a symbolic representation of space can be taken advantage of by LLMs for learning long-horizon tasks. Our proposed algorithm, SGIM-STAR, in the HRL framework with a high-level agent can actively choose its learning strategy between reinforcement learning or a LLM-based planner. The originality is that both strategies use a symbolic representation of the sensorimotor space, by learning an emergent symbolic representation, making it compatible with the symbolic space of language.

We first explore whether a space partition into regions learned online by HRL can have a correspondence with natural language instructions in section 3. To exploit this correspondence, we introduce in section 4 the algorithm SGIM-STAR, which chooses between a LLM-based planner or a RL-based planner based on intrinsic motivation. The experimental results are presented in section 5, where we test SGIM-STAR in Ant Maze environment, originally introduced by Duan et al. [2016] and later popularized in hierarchical reinforcement learning benchmarks by Nachum et al. [2018].

3 Spatial Abstractions used to Translate Natural Language Instructions

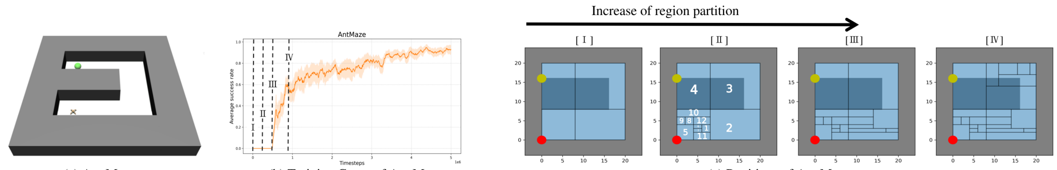
We examine how symbolic spatial abstractions can be used for grounding language. In this section, we outline the algorithm STAR, then report our first investigation whether language can be grounded in the emergent spatial abstraction.

3.1 Preliminary: Spatio-Temporal Abstraction via Reachability (STAR)

The STAR algorithm Zadem et al. [2024] is a reinforcement learning algorithm that uses a three-layered hierarchical structure:

- Navigator: the top-level agent plans the long-horizon path. It is trained by Q-learning which samples an abstract goal $G \in \mathcal{G}$ every k steps that should help to reach the task goal g^* from the current agent's state $(G_{t+k} \sim \pi_{Nav}(s_t, g^*))$.
- Manager: the mid-level agent trained by TD3 which picks subgoals in the state space every l steps $(g_{t+l} \sim \pi_{Man}(s_t, G_{t+k}))$, we notice that k is a multiple of l.
- Controller: the low-level policy trained by TD3 that samples actions to reach the subgoal every step $(a \sim \pi_{Cont}(s_t, g_{t+l}))$

STAR incrementally refines the partition of the sensorimotor space (cf. Fig. 1 (c)) by analyzing k-step reachability relations between goal regions. The Refinement module uses as inputs the past episodes $\mathcal{D}$ and a the list of abstract goals $\mathcal{E}$ visited during the last episode, and outputs a partition of the state space.



(a) Ant Maze (b) Training Curve of Ant Maze (c) Partitions of Ant Maze

Figure 1: (a) Ant Maze environment, (b) Average success rate of STAR (from Zadem [2024]), (c) Partition into regions of STAR. The regions in (c) are the internal representation emerging during the training at timestamps noted in (b). The red point is the initial position of the robot. The yellow point is the goal position. Our translator translates instructions to guide the robot (eg: "go east to the end, turn north until past the wall and go west until the end"), into a sequence of traversed regions (eg for Partition II, the output is $5 \rightarrow 11 \rightarrow 2 \rightarrow 3 \rightarrow 4$).

Table 1: Mean G-BLEU scores of translation of natural language instructions over 4 runs for each partition in the Ant Maze environment.

translator	**Ant Maze**			
	P-I	P-II	P-III	P-IV
GPT o3-m	1	1	1	0.87
Claude	1	1	0.73	0.34
Deepseek	1	0.9	0.53	0.65
GROK	1	1	1	0.89

3.2 Translation of Natural Language to the Spatial Abstraction

In the first experiment, we probe the possibility of grounding instructions in natural language by means of a spatial abstraction leaned from a reinforcement learning. Concretely, we test if we can translate the plan outlined by natural language instructions into a succession of subgoals. For this translation, we exploit and evaluate several LLMs in interpreting fixed natural language instructions across varying symbolic abstractions. We keep the agent's start and goal positions fixed and apply the same instruction to all partition levels. The natural language instruction for Ant Maze is: "Move right until you completely pass the wall on your left, move up until you have crossed the upper wall, turn left and proceed until you reach the goal."

We report in Table 1 the G-BLEU score for the translation of this instruction into a sequence of regions by four commonly used reasoning LLMs: GPT o3-mini, Claude 3.7, DeepSeek-r1, and GROK. While GPT o3-mini achieves the highest scores, all translators achieve scores above 0.5 across tasks, indicating a generally successful translation of human instructions into the agent's internal symbolic representation. In the Ant Maze environment, all translator scores decrease as the number of regions increases. However, the degradation of performance varies by model: GPT o3-mini and GROK demonstrate greater robustness than DeepSeek-r1 and Claude 3.7.

Given the consistent trends observed across LLMs, we select GPT o3-mini as the representative LLM for subsequent experiments.

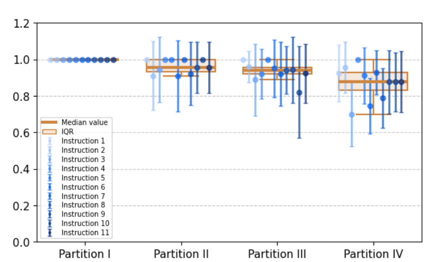


Figure 2: G-BLEU scores for translation of natural language instructions tested in Ant Maze. For each internal representation, we plot in blue the average and standard deviation of 10 queries for each instruction, and boxplot in brown, orange and yellow the average and IQR over the 11 instructions.

In order to test the robustness of the previous results with statistical tests, for our second experiments, we design 11 different natural language instructions for each environment (cf. Appendix B), and tested all 11 instructions on all four partitions. We constructed the prompt (cf. Appendix C and queried the LLM 10 times. Figure 2 illustrates the average G-BLEU scores across the symbolic partitions of Ant Maze. We observe a perfect translation performance in Partition I. The trend of translation performance observed in Ant Maze is a consistent slightly drop from Partition I to Partition IV. This trend is interpretable through the partition structures shown in Figure 1. Partition I represents a very coarse abstraction with minimal region division, making it easier for the LLM to infer plausible region sequences regardless of the instruction quality. Then, when the partition becomes more granular in Ant Maze, the LLM is more prone to mistakes.

4 SGIM-STAR : Combining LLM and RL as planners

Our proposed algorithm follows STAR's hierarchical structure and reachability-aware abstraction, but augments the high-level agent with a large language model (LLM) and introduces a partition-wise, progress-driven *active choice* between a (Q-learning) RL-based planner and an LLM planner. This design aims to keep low-level learning intact while (i) leveraging language for top-level guidance when beneficial, (ii) reducing LLM usage cost by invoking it only when progress warrants it, and (iii) improving stability on long-horizon tasks via per-partition selection rather than a single, global switch. In this section, we outline STAR, our integration of an LLM at the high level and the active imitation mechanism.

4.1 Integration of an LLM into the Top Level Agent

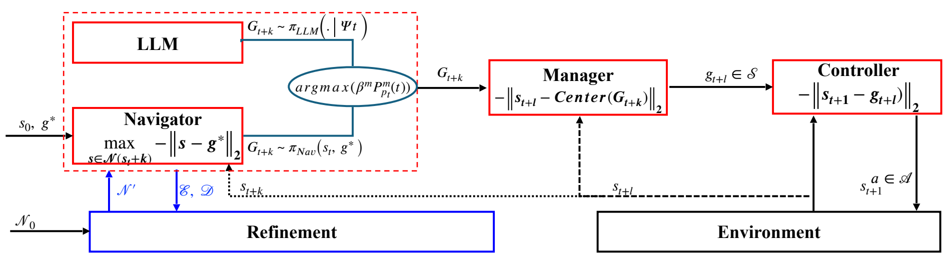


Figure 3: Algorithmic architecture of SGIM-STAR which integrates the LLM and the Navigator of STAR at the top level agent. The Navigator or the LLM selects subgoal regions $G \in \mathcal{G}$, while the middle-level manager and low-level controller are unchanged. The choice between the LLM and the Navigator.

We extend STAR by incorporating an LLM into the top-level agent. The structure is shown in Fig. 3. Instead of relying solely on the Navigator policy $\pi_{\text{STAR}}(G|s_t)$, we introduce an LLM-based planner $\pi_{\text{LLM}}(G|\Psi_t)$, which operates on a prompt Ψ_t encoding the agent's current region, available partitions, and task description. Thus, the top-level goal selection becomes:

$$G'_{t+k} \sim \pi_{\text{LLM}}(\cdot \mid \Psi_t), \quad \Psi_t = \psi(s_t, \mathcal{G}_N, g^*, \mathcal{M}_t), \tag{1}$$

where $\mathcal{G}_N$ is the set of admissible regions, g^* is the task goal, and $\mathcal{M}_t$ summarizes recent exploration. This modification allows the Navigator to integrate human-readable instructions and world knowledge expressed in natural language, thereby aligning regional exploration with external guidance or commonsense priors.

4.2 Active Imitation Learning of the Top Level Agent

To dynamically balance between the original STAR Navigator and the LLM-based planner, we use intrinsic motivation based on progress measure as a selection mechanism.

Initialization. For the first N decision steps, the planner is chosen randomly between the Navigator and LLM in order to populate both buffers with initial experience.

Progress signal. At each timestep t, let $m \in \{\text{STAR}, \text{LLM}\}$ denote the planner used, and let $p_t = \phi(s_t)$ be the current region of the state space. We define the incremental reward difference: $\Delta_t = r_t - r_{t-1}$ which reflects the immediate progress attributable to the planner's decision at t. This value Δ_t is stored as $\Delta_t^{(m)}$ in the buffer of the corresponding planner m for the active region p_t.

Discounted progress accumulation. For each region p_t and planner m, we compute a discounted cumulative progress over a sliding window of length n:

$$P_{p_t}^{(m)}(t) = \sum_{j=0}^{n} \alpha^j \, \Delta_{t-j}^{(m)} \tag{2}$$

where $\alpha \in (0, 1)$ is a progress discount factor that emphasizes recent progress while retaining memory of past improvements.

Planner selection rule. At each decision step, the algorithm selects the planner according to a progress-maximization criterion:

$$m(t) = \underset{m \in \{\text{STAR}, \text{LLM}\}}{\arg\max} \left\{ \beta^{(m)} P_{p_t}^{(m)}(t) \right\}, \tag{3}$$

where $\beta^{(\text{LLM})} \geq 0$ is a scaling factor that controls the relative influence of LLM-derived progress ($\beta^{(\text{STAR})} = 1$).

Algorithm 1 SGIM-STAR

Require: Discount factor $\alpha \in (0, 1)$, window size $n \in \mathbb{N}$, warm-start steps $N \in \mathbb{N}$,
    Weights $\beta^{(\text{LLM})} \geq 0,\ \beta^{(\text{STAR})} = 1$
    Planners $\mathcal{M} = \{\text{STAR}, \text{LLM}\}$; partition map $\mathcal{P}_0 = \phi(s_0)$
Initialize
    $\forall$ region $p \in \mathcal{P}_0, \forall$ planner $m \in \mathcal{M}$, initialise buffer $\mathcal{B}_p^{(m)}$ at capacity $n{+}1$ to store Δ values
    $\forall$ region $p \in \mathcal{P}_0, \forall$ planner $m \in \mathcal{M}$, discounted progress scores $P_p^{(m)} \leftarrow 0$
    $t \leftarrow 0$, observe s_0 and reward r_0
while episode not terminated **do**
    $p_t \leftarrow \phi(s_t)$ {Identify current region}
    if $t < N$ **then**
        Choose $m(t) \sim \text{Uniform}(\mathcal{M})$ {Warm-start randomization}
    else
        Compute weighted scores in region p_t:
        $P_{p_t}^{(m)}(t) = \sum_{j=0}^{n} \alpha^j \, \Delta_{t-j}^{(m)}$
        $m(t) \leftarrow \underset{m \in \{\text{STAR}, \text{LLM}\}}{\arg\max} \left\{ \beta^{(m)} P_{p_t}^{(m)}(t) \right\}$
    end if
    Use planner $m(t)$ to select top-level region G_t and act for one decision step
    Observe next state s_{t+1} and reward r_{t+1}
    $\Delta_{t+1}^{(m(t))} \leftarrow r_{t+1} - r_t$ {Incremental reward difference}
    Append $\Delta_{t+1}^{(m(t))}$ to buffer $\mathcal{B}_{p_t}^{(m(t))}$ (drop oldest if $|\mathcal{B}_{p_t}^{(m(t))}| > n{+}1$)
    $t \leftarrow t + 1$, $r_t \leftarrow r_{t+1}$, $s_t \leftarrow s_{t+1}$
end while=0

We notice that if a region p has already been well explored, then both planners yield low incremental progress ($\Delta_t^{(m)} \approx 0$), resulting in small accumulated scores $P_p^{(m)}(t)$. Conversely, when the agent enters a novel region, progress signals tend to be larger, biasing selection toward the planner that has demonstrated stronger improvement in unexplored regions. This mechanism naturally encourages exploitation of novel areas while reducing reliance on planners that fail to generate additional progress in familiar regions.

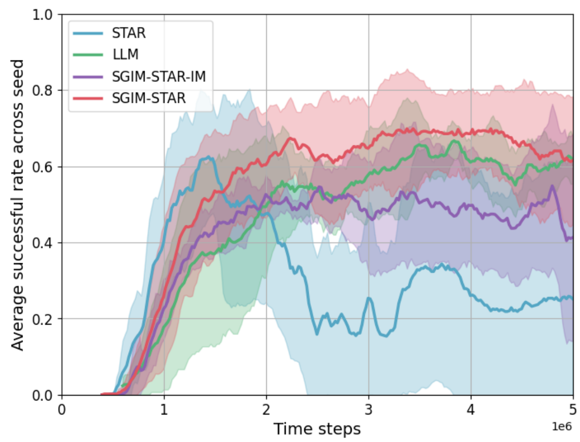


Figure 4: Average successful rate of four methods

5 Performance of SGIM-STAR

5.1 Experiment Setup

We evaluate our proposed algorithm, SGIM-STAR in the Ant Maze environment. Our primary evaluation task is Ant Maze, in which the Ant robot must navigate a $\supset$-shaped maze and reach an exit located at the top left corner. This task is inherently hierarchical: success requires both fine-grained locomotion control (low-level) and long-horizon navigation through the maze (top-level). Moreover, Ant Maze is a suitable benchmark for evaluating LLM integration, as solving the task requires reasoning over abstract spatial regions and selecting long-horizon subgoals rather than relying solely on local control.

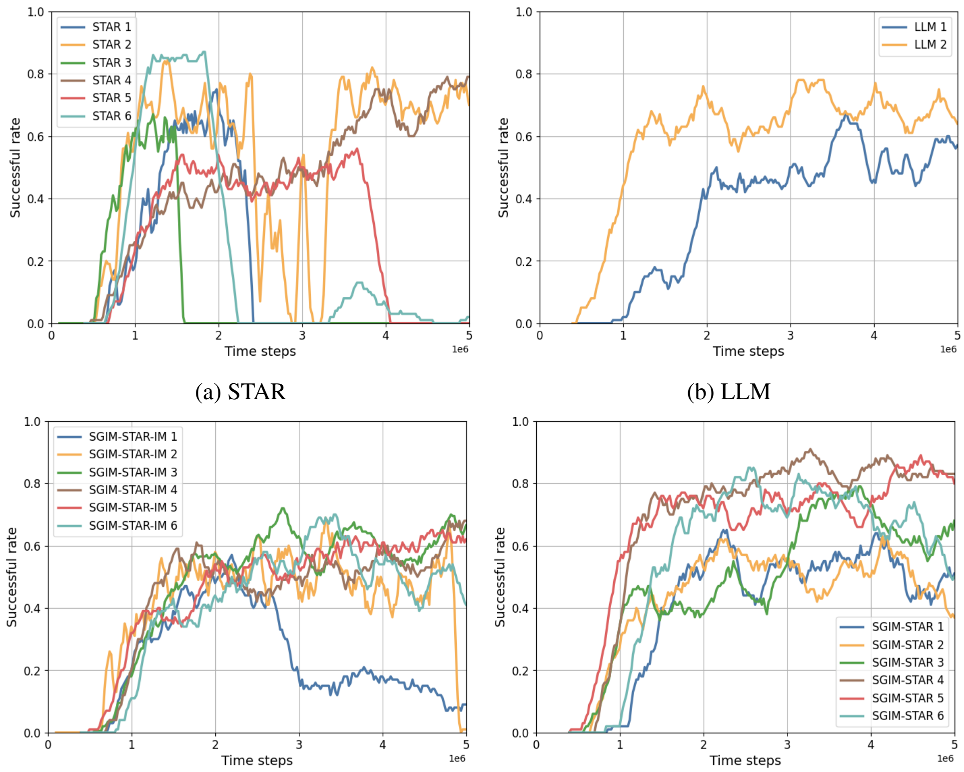
The prompt design used for the LLM-based planner in Ant Maze is provided in the Appendix. We compare our SGIM-STAR with the following methods:

- **STAR**: the original STAR framework where the high-level agent is the Navigator policy trained via Q-learning.
- **LLM Planner**: STAR algorithm where the Navigator is replaced by an LLM using a handcrafted prompt. The Manager and Controller remain unchanged.
- **SGIM-STAR-IM** (SGIM-STAR with Interactive learning at the Meta level) : to study the importance of the partition, we considered an ablation where the top-level agent adaptively switches between STAR and LLM, but without considering the environment partitions defined by the STAR abstraction, as with the algorithm SGIM-IM Nguyen and Oudeyer [2012a]: instead of computing $\mathcal{P}_{pt}^{(m)}$ for each region, we consider it for the whole state space.

All agents are trained to navigate the maze to a goal location, and we track their success rates over 5 million environment step on 6 random seeds for SGIM-STAR, SGIM-STAR-IM and STAR, and 2 seeds for LLM Planner, all experiments are trained on one NVIDIA GEFORCE RTX 4090 GPU.

5.2 Results and Analyses

Fig.4 shows the average success rate across random seeds for each approach. We observe that the partition-based method not only attains the greatest success rate of 0.7 by the end of training but also exhibits the smallest variance across seeds, indicating consistent learning outcomes. In contrast, the other methods reach lower success levels and have wider fluctuations. Notably, the LLM-only agent plateaus around a moderate success rate at around 0.6, while the pure STAR agent's average performance degrades significantly by the end of training due to collapses in some runs (Fig.5a). These results demonstrate that incorporating partitioned task structure and adaptively integrating LLM guidance produces superior resilience in this long-horizon task.

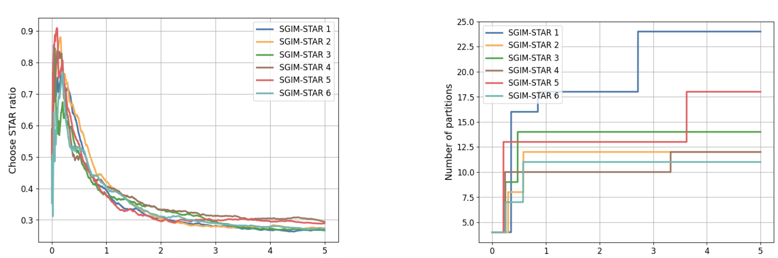


(a) STAR (b) LLM

(c) SGIM-STAR-IM (d) SGIM-STAR

Figure 5: Successful rate of each run

To further analyze stability, we examine individual training curves of each method. The pure RL baseline STAR, as shown in Fig.5a, collapses frequently mid-training across seeds. This instability indicates a lack of resilience: the convergence of STAR is not guaranteed when learning such a complex, long-horizon task without additional guidance. Figure 5b shows that the LLM-only agent can reach moderate success rates plateau without any dropping of performance, demonstrating the potential of a pretrained planner to guide exploration. However, the use of LLM-planner is too costly and the learning process of LLM-only is three times slower than the others, which limits the further use of LLM in the top-level agent. Fig.5d shows that SGIM-STAR demonstrates remarkably consistent improvement across training, with almost no catastrophic drops in performance. In contrast, the SGIM-STAR-IM variant also suffers abrupt performance collapses after initial learning spurts from Fig,5c. This suggests that state-space partitioning plays a critical role in stabilizing long-horizon learning.



(a) The ratio of chosen STAR (b) Environment Partitions

Figure 6: Adaptive choice of Planner by Partition

A key insight is that SGIM-STAR selectively shifts to LLM guidance once it becomes beneficial. Early on, STAR's fast learning of basic navigation yields quicker gains, so the agent heavily favors the STAR Navigator policy. As training progresses, the agent discovers a more refined symbolic structure of the maze via partitions, it increasingly relies on the LLM for top-level planning. Fig.6a quantifies this behavior: at the start of training, the fraction of top-level choices directed by STAR is extremely high, as the task structure emerges, this fraction steadily declines, and by the end of training the agent chooses STAR for only one third of decisions on average, indicating that it has shifted to predominantly following the LLM's guidance in later stages. In other words, SGIM-STAR intelligently "trusts" STAR in the beginning when the LLM's abstract guidance might not yet be enough, and then gradually transitions to the LLM as the partitions learned by STAR provide a reliable framework for planning. This adaptive scheduling of who controls the top-level actions is crucial to achieving both high efficiency and stability.

Another contributing factor to the robustness of the SGIM-STAR is the growth of its state abstraction over time. During training, the SGIM-STAR incrementally partitions the state space into more regions as it encounters new situations. Fig.6b tracks the number of partitions in each run over the course of training. In effect, the partitioning mechanism provides a form of symbolic memory that the LLM can leverage which grounds the LLM's planning in the agent's learned experience.

6 Discussion

After showing a possible parallel between natural language instructions and spatial abstraction, which hints a grounding of LLMs in a space representation, we introduced SGIM-STAR which selectively combines a RL-based planner with an LLM-based planner using a partition-wise, progress-driven rule. Our analysis yields four key characteristics of our method:

(1) Mutual stabilization and lighter planning. LLM guidance stabilizes STAR by providing supplementary top-level proposals when the RL-based planner becomes brittle, while STAR makes the overall system lighter than an LLM-only planner by supplying competent, inexpensive planning during large portions of training. Overall, the RL-based planner constitutes a bottom-up agent learning from its trial and error with the environment, whereas the LLM planner is a top-down agent sharing its internal world representation to this specific task. Their combination mutualizes both a bottom-up and a top-down process. The instrinsically motivated, progress-based selection between the two planners improves learning progress and stabilizes performance.

(2) Cost-aware usage of the LLM. SGIM-STAR uses the LLM only when necessary: calls to the LLM are conditional on partition-wise progress and thus avoided when the STAR navigator suffices. Compared to an LLM-only navigator, this conditional usage reduces planner cost and latency while still reaping the benefits of LLM exploration.

(3) Start planning with RL, then switch to LLM. The agent relies more on a RL planner in the early phase—when the internal representation is coarse—and gradually shifts toward LLM guidance as the internal representation becomes richer and more meaningful for language reasoning.

(4) Formulation that enables language grounding. Crucially, our learning formulation builds a discretized, partitioned representation—from bottom-up RL experiences. This evolving symbolic structure leverages LLMs to help the learning process of the agent, by offering a grounded correspondence of regions to LLM symbols.

7 Conclusion

We present *SGIM-STAR*, a simple, partition-wise-progress-based algorithm that switches between a STAR high-level planner and an LLM planner. In Ant Maze, it achieves the best and most stable results, avoiding STAR's mid-training drops and the high cost and low efficiency of an LLM-only planner. The LLM stabilizes STAR when learning becomes unstable, while STAR keeps the system light by handling most decisions—so we use the LLM only when needed. As the state-space partition grows during training, the agent uses STAR more frequently at the early stages of training while soliciting the LLM more in the later ones, since the richer representation gives the LLM more meaningful inputs. This indicates that a richer internal representation can offer a better spatial grounding of LLM planners. This combination makes training steadier and more efficient than using STAR or an LLM alone, owing to SGIM's active choice. SGIM-STAR thus autonomously devises learning curriculum and strategy, starting with a RL planner then switching to LLM planner.

Acknowledgment

This research work is supported by the Hi! PARIS Center.

References

A. Baranes and P.-Y. Oudeyer. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73, 2013. doi: 10.1016/j.robot.2012.05.008.

T. Carta, C. Romac, L. Gaven, P.-Y. Oudeyer, O. Sigaud, and S. Lamprier. Herakles: Hierarchical skill compilation for open-ended llm agents, 2025. URL `https://arxiv.org/abs/2508.14751`.

S. Chakraborty, K. Weerakoon, P. Poddar, M. Elnoor, P. Narayanan, C. Busart, P. Tokekar, A. S. Bedi, and D. Manocha. Re-move: An adaptive policy design for robotic navigation tasks in dynamic environments via language-based feedback. 03 2023. URL `https://arxiv.org/abs/2303.07622`.

D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. Palm-e: An embodied multimodal language model, 2023. URL `https://arxiv.org/abs/2303.03378`.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

N. Duminy, S. M. Nguyen, J. Zhu, D. Duhaut, and J. Kerdreux. Intrinsically motivated open-ended multi-task learning using transfer learning to discover task hierarchy. *Applied Sciences*, 11(3), 2021.

C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 49–58, New York, New York, USA, 20–22 Jun 2016. PMLR.

J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, 2018. URL `https://proceedings.mlr.press/v80/fu18a.html`.

J. Gottlieb, P.-Y. Oudeyer, M. Lopes, and A. Baranes. Information-seeking, curiosity, and attention: computational and neural mechanisms. *Trends in Cognitive Sciences*, 17(11):585–593, 10 2013.

H. Hu, D. Yarats, Q. Gong, Y. Tian, and M. Lewis. Hierarchical decision making by generating and following natural language instructions. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

D.-S. Jang, D.-H. Cho, W.-C. Lee, S.-K. Ryu, B. Jeong, M. Hong, M. Jung, M. Kim, M. Lee, S. Lee, and H.-L. Choi. Unlocking robotic autonomy: A survey on the applications of foundation models. *International Journal of Control, Automation and Systems*, 22(8):2341–2384, Aug 2024.

Y. Jiang, S. S. Gu, K. P. Murphy, and C. Finn. Language as an abstraction for hierarchical deep reinforcement learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

K. Jokinen. The need for grounding in LLM-based dialogue systems. In T. Dong, E. Hinrichs, Z. Han, K. Liu, Y. Song, Y. Cao, C. F. Hempelmann, and R. Sifa, editors, *Proceedings of the Workshop: Bridging Neurons and Symbols for Natural Language Processing and Knowledge Graphs Reasoning (NeusymBridge) @ LREC-COLING-2024*, pages 45–52, Torino, Italia, May 2024. ELRA and ICCL.

W. B. Knox, P. Stone, and C. Breazeal. *Training a Robot via Human Feedback: A Case Study*, pages 460–470. Springer International Publishing, 2013.

A. Levy, G. Konidaris, R. Platt, and K. Saenko. Learning multi-level hierarchies with hindsight. In *International Conference on Learning Representations*, 2019.

S. Li, L. Zheng, J. Wang, and C. Zhang. Learning subgoal representations with slow dynamics. In *International Conference on Learning Representations (ICLR)*, 2021. URL `https://openreview.net/forum?id=wxRwhSdORKG`.

Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar. Eureka: Human-level reward design via coding large language models. In *2nd Workshop on Language and Robot Learning: Language as Grounding*, 2023.

O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

S. M. Nguyen and P.-Y. Oudeyer. Interactive learning gives the tempo to an intrinsically motivated robot learner. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 645–652, 2012a.

S. M. Nguyen and P.-Y. Oudeyer. Active choice of teachers, learning strategies and goals for a socially guided intrinsic motivation learner. *Paladyn, Journal of Behavioral Robotics*, 3(3):136–146, 2012b. doi: 10.2478/s13230-013-0110-z.

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1(6), 2007.

P.-Y. Oudeyer, F. Kaplan, V. Hafner, and A. Whyte. The playground experiment: Task-independent development of a curious robot. In *Spring Symposium on Developmental Robotics*, pages 42–47. AAAI, 2005.

C. O. Retzlaff, S. Das, C. Wayllace, P. Mousavi, M. Afshari, T. Yang, A. Saranti, A. Angerschmid, M. E. Taylor, and A. Holzinger. Human-in-the-loop reinforcement learning: A survey and position on requirements, challenges, and opportunities. *Journal of Artificial Intelligence Research*, 79:359–415, Jan. 2024.

S. Ross, G. J. Gordon, and J. A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635, 2011. URL `https://proceedings.mlr.press/v15/ross11a.html`.

J. Schmidhuber. Curious model-building control systems. In *Proc. Int. Joint Conf. Neural Netw.*, volume 2, pages 1458–1463, 1991.

Y. Shentu, P. Wu, A. Rajeswaran, and P. Abbeel. From llms to actions: Latent codes as bridges in hierarchical robot contro. In *IROS*, 2024.

A. Shon, D. Verma, and R. P. Rao. Active imitation learning. In *American Association for Artificial Intelligence*, volume 22, page 756. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

M. Zadem. *Automatic Symbolic Goal Abstraction via Reachability Analysis in Hierarchical Reinforcement Learning*. PhD thesis, IP Paris, 2024.

M. Zadem, S. Mover, and S. M. Nguyen. Goal space abstraction in hierarchical reinforcement learning via set-based reachability analysis. In *2023 IEEE International Conference on Development and Learning (ICDL)*, pages 423–428, Nov 2023.

M. Zadem, S. Mover, and S. M. Nguyen. Reconciling spatial and temporal abstractions for goal representation. In *The Twelfth International Conference on Learning Representations*, 2024.

T. Zhang, S. Guo, T. Tan, X. Hu, and F. Chen. Generating adjacency-constrained subgoals in hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/f5f3b8d720f34ebebceb7765e447268b-Abstract.html`. Spotlight.

B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, brian ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *7th Annual Conference on Robot Learning*, 2023.

Appendix

A Prompt

```
Data:
-State: Region 5
-Goal: Region 4

-Adjacency list:
    Region 1: [6, 7, 11, 12, 14, 18]
    Region 2: [12, 13, 14, 15]
    Region 3: [4, 19, 21]
    Region 4: [3]
    Region 5: [6, 7, 8, 9, 11]
    Region 6: [1, 5, 7, 11]
    Region 7: [1, 5, 6, 12]
    Region 8: [5, 9, 10, 12]
    Region 9: [5, 8, 10]
    Region 10: [8, 9, 12, 13]
    Region 11: [1, 5, 6, 17]
    Region 12: [1, 2, 7, 8, 10]
    Region 13: [2, 10, 15]
    Region 14: [1, 2, 15, 18]
    Region 15: [2, 13, 14, 16, 18, 20]
    Region 16: [15, 17, 18]
    Region 17: [11, 16, 18]
    Region 18: [1, 14, 15, 16, 17]
    Region 19: [3, 20, 21, 22, 23]
    Region 20: [15, 19]
    Region 21: [3, 19, 22]
    Region 22: [19, 21, 23]
    Region 23: [19, 22]

-The top-down view of the maze is shown
 below, 'W' represents walls, 'A'
 represents the ant's current position,
 'G' represents the goal. The number
 represents the region number:

4  4  4  4  4  3  21 21 22 22 22 22 23
4  4  4  4  4  3  19 19 19 19 19 19 19
4  G  4  4  4  3  19 19 19 19 19 19 19
W  W  W  W  W  W  W  W  W  19 19 19 19
W  W  W  W  W  W  W  W  W  19 19 19 19
W  W  W  W  W  W  W  W  W  19 19 19 19
W  W  W  W  W  W  W  W  W  20 20 20 20
10 10 10 10 10 13 13 15 15 15 15 15 15
9  9  8  12 12 2  2  15 15 15 15 15 15
5  5  5  7  1  14 14 15 15 15 15 15 15
5  A  5  6  1  18 18 18 18 18 16 16 16
5  5  5  11 11 17 17 17 17 17 17 17 17


-Thinking Process:
   1.Identify the agent's current region
     and the goal region.
   2.Identify where is the wall.
   3.Examine the adjacency list and the
     maze to see which regions connect to
     the current region.
   4.From these connected regions, choose
     the one that moves closest to the goal
     without hitting walls.
```

B 11 Instructions given to the partitions in the Ant Maze environemnt

C Evaluating the Translation of Natural Language into Symbolic Abstractions

Let $S = \{s_1, s_2, \ldots, s_N\}$ denote the set of internal symbolic representations identified during the developmental learning process of STAR. We design a collection of instructions $I = \{I_1, I_2, \ldots, I_N\}$ from J humans, where each element is a set of instructions $I_i = \{I_{i,1}, I_{i,2}, \ldots, I_{i,J}\}$. Each instruction $I_{i,j}$ corresponds to a unique natural language command describing the goal or behavior related to s_i by the human j. We define a prompt construction function $f_{\text{prompt}}(s_i, I_{i,j})$ that takes a symbolic representation s_i and an associated instruction $I_{i,j}$ to generate a textual prompt $p_{i,j}$ for the LLM:

$$p_{i,j} = f_{\text{prompt}}(s_i, I_{i,j}) \tag{4}$$

Due to the stochastic nature of LLMs, a given prompt $p_{i,j}$ may result in different outputs across multiple queries. We introduce a random state r_k (*e.g.*, random seed) and define the LLM-generated output at query time k as:

$$o_{i,j,k}^{LLM} = \text{LLM}(p_{i,j}, r_k) \tag{5}$$

To establish a reference for evaluation, domain experts provide human-annotated ground truth outputs $G_{i,j}$ for each symbolic-instruction pair $(s_i, I_{i,j})$, so for each $G_i \in G$, $G_i = \{G_{i1}, G_{i2}, ..., G_{iJ}\}$. It is

Table 2: Natural language instructions for the Ant Maze environemnt

Instructions	Ant Maze
1	Move east until you are past the wall, then go north beyond the upper barrier, turn west, and continue until you reach the goal.
2	Head right until there's no obstruction in your way, then move up until the path is clear, turn left, and proceed to your destination.
3	Travel right to get around the first wall, ascend straight up to clear the second, then shift left and move toward the goal.
4	Move horizontally to the right until you pass the boundary, then go straight up until no walls remain, turn left, and continue forward.
5	Proceed east to navigate around the wall, then ascend north until you are clear, turn west, and move straight to your target.
6	Walk right until you exit the confined space, then go up beyond the vertical wall, turn left, and follow the open path to the goal.
7	Move sideways to the right until the wall is behind you, then climb upwards until there's no barrier, turn left, and walk toward the goal.
8	Head eastward until you escape the enclosed area, ascend northward past the last obstruction, then turn west and reach your goal.
9	Travel right along the open path until no wall blocks your way, go straight up past the top structure, then turn left and proceed to your destination.
10	Move toward the right until you have an open vertical passage, then go up until the way is clear, turn left, and walk directly to your goal.
11	Navigate eastward beyond the boundary, then ascend straight up to clear the structure, turn left, and reach the goal without further obstacles.

Algorithm 2 Translation by LLM of Human Instructions into Emergent Symbolic Representations

Require: $S = \{s_1, s_2, \ldots, s_N\}$: symbolic representations from STAR,
$I = \{I_1, I_2, \ldots, I_N\}$: set of human instructions sets,
$G = \{G_1, G_2, \ldots, G_N\}$: set of ground truth output sets
Ensure: $M = \{M_{1,1}, M_{1,2}, \ldots, M_{N,J}\}$: evaluation scores
for $i = 1$ to N **do**
  for $j = 1$ to J **do**
    $p_{i,j} \leftarrow f_{\text{prompt}}(s_i, I_{i,j})$
    for $k = 1$ to K **do**
      $o^{\text{LLM}}_{i,j,k} \leftarrow \text{LLM}(p_{i,j}, r_k)$
      $m_{i,j,k} \leftarrow \max_{q=1\ldots Q} M\left(o^{\text{LLM}}_{i,j,k}, G_{i,j,q}\right)$
    end for
    $M_{i,j} \leftarrow \frac{1}{K}\sum_{k=1}^{K} m_{i,j,k}$
  end for
end for=0

possible that multiple reference outputs are compatible with the pair $(s_i, I_{i,j})$, in this case we consider a set of references $G_{ij} = \{G_{i,j,1}, \ldots, G_{i,j,Q}\}$ for each $G_{i,j} \in G_i$. The set of ground truth output sets is the collection of all these references: $G = \{G_1, G_2, \ldots G_N\}$. We then define an evaluation metric $M\left(o^{\text{LLM}}, G\right)$ that measures the similarity between the LLM-generated output and the corresponding human-provided reference. Since there may exist several possible ground truth responses associated with one internal symbolic set and one instruction, we compute the performance for each pair by taking the maximum similarity across all human-provided references $G_{i,j,q}$. The similarity is high if at least one of the references provides a high similarity. The average score over K runs is defined as:

$$M_{i,j} = \frac{1}{K} \sum_{k=1}^{K} \max_{q=1\ldots Q} M\left(o_{i,j,k}^{\mathrm{LLM}}, G_{i,j,q}\right) \tag{6}$$

This formulation allows us to robustly evaluate the performance of LLMs to translate symbolic representations into interpretable language aligned with human expectations. The overall algorithmic procedure is described in Algorithm 2.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

1. **Claims**

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

   Answer: [Yes]

   Justification: The contribution of this paper is that, we propose an algorithm named SGIM-STAR that creates online a discrete world representation by HRL and it switches its top-level planner between a learned Q-learning Navigator and LLM planner.

   Guidelines:

   - The answer NA 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 NA 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: [No]

   Justification: We don't discuss the limitation of the method.

   Guidelines:

   - The answer NA 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: [NA]

   Justification: The method is more about application, we don't have any theoretical results

   Guidelines:

   - The answer NA 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: The paper discloses all the information needed to reproduce

   Guidelines:

   - The answer NA 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: The code is available at `https://github.com/ZiqiLoveSunshine/SGIM-STAR`

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The paper provide experiment settings.

Guidelines:

- The answer NA 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: We report the average values of learning curve with its standard deviation.

   Guidelines:

   - The answer NA 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: The training is executed on one NVIDIA GEFORCE RTX 4090 GPU.

   Guidelines:

   - The answer NA 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: We obey the rules of Ethics

Guidelines:

- The answer NA 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: [No]

Justification: We don't think our work has negative societal impact

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We don't present any risks.

Guidelines:

- The answer NA 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 cite the original paper that produced the code package, the licence that we use is CC-BY 4.0

    Guidelines:

    - The answer NA 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: [NA]

    THis work does not introduce new assets.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: our paper doesn't involve any human subject.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: our research doesn't involve any human subject.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The LLM is used in the navigator part, where SGIM-STAR choose to use LLM navigator or learned Q-learning navigator based on the progress that they made.

    Guidelines:

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.