L2M2: A Hierarchical Framework Integrating Large Language Model and Multi-agent Reinforcement Learning

Minghong Geng[1,*], **Shubham Pateria**[1], **Budhitama Subagdja**[1], **Lin Li**[2], **Xin Zhao**[3,*] and **Ah-Hwee Tan**[1]
[1]Singapore Management University, Singapore
[2]MIGU Co., Ltd, Beijing, China
[3]Tsinghua University, Beijing, China
mhgeng.2021@phdcs.smu.edu.sg, {shubhamp, budhitamas}@smu.edu.sg,
lilin@migu.chinamobile.com, zhaoxin1117@tsinghua.edu.cn, ahtan@smu.edu.sg

Abstract

Multi-agent reinforcement learning (MARL) has demonstrated remarkable success in collaborative tasks, yet faces significant challenges in scaling to complex scenarios requiring sustained planning and coordination across long horizons. While hierarchical approaches help decompose these tasks, they typically rely on hand-crafted subtasks and domain-specific knowledge, limiting their generalizability. We present L2M2, a novel hierarchical framework that leverages large language models (LLMs) for high-level strategic planning and MARL for low-level execution. L2M2 enables zero-shot planning that supports both end-to-end training and direct integration with pre-trained MARL models. Experiments in the VMAS environment demonstrate that L2M2's LLM-guided MARL achieves superior performance while requiring less than 20% of the training samples compared to baseline methods. In the MOSMAC environment, L2M2 demonstrates strong performance with pre-defined subgoals and maintains substantial effectiveness without subgoals - scenarios where baseline methods consistently fail. Analysis through kernel density estimation reveals L2M2's ability to automatically generate appropriate navigation plans, demonstrating its potential for addressing complex multi-agent coordination tasks.

1 Introduction

Multi-agent reinforcement learning (MARL) has emerged as a powerful framework for coordinating autonomous systems, demonstrating remarkable success in both real-world applications, such as urban traffic control [Zhang *et al.*, 2019] and multi-robot navigation [Oroojlooy and Hajinezhad, 2023], and complex simulated environments such as StarCraft II [Vinyals *et al.*, 2017; Samvelyan *et al.*, 2019]. Recent studies have advanced MARL towards increasingly challenging problems, including environments with rich stochasticity [Ellis *et al.*, 2023], human-agent collaboration [Li *et al.*, 2023b; Liu *et al.*, 2024a], and long-horizon sequential planning [Geng *et al.*, 2024a]. These complex domains intensify challenges for MARL, such as sample efficiency, temporal credit assignment, exploration in large state-action spaces, and non-stationarity [Hernandez-Leal *et al.*, 2019].

Hierarchical multi-agent architectures [Ahilan and Dayan, 2019; Geng *et al.*, 2024b] attempt to address these challenges through multi-level control structures that decompose complex tasks into manageable subtasks. These methods typically employ a centralized high-level planner agent for subtask allocation and multiple low-level executor agents for subtask accomplishment. However, existing methods rely heavily on domain knowledge for subtask definition and assignment, resulting in task-specific policies with limited transferability across tasks. This dependency on domain knowledge and the costly retraining in new scenarios presents a critical bottleneck in scaling MARL to diverse real-world applications.

Recent research has explored integrating large language models (LLMs) into multi-agent systems [Li *et al.*, 2023b], leveraging their pre-trained capabilities in contextual inference and natural language understanding. As existing approaches primarily focus on pure *multi-LLM-agent* systems, they generally face scalability challenges with large agent populations. To address these challenges, we propose L2M2 (**L**arge **L**anguage **M**odel and **M**ulti-agent Reinforcement Learning), a hierarchical framework that integrates LLMs with multiple reinforcement learning agents. Following the *LLM-as-policy* paradigm [Carta *et al.*, 2023; Szot *et al.*, 2024; Zhou *et al.*, 2024], L2M2 introduces an *LLM agent* that leverages a large language model as its decision-making policy for task decomposition and allocation, while reinforcement learning agents execute the resulting subtasks through direct environmental interactions. L2M2 enables *zero-shot planning* [Dalal *et al.*, 2023] to guide RL agents directly with pre-trained LLMs. This approach significantly reduces the computational overhead of developing policies for high-level controllers from scratch [Agarwal *et al.*, 2023; Blumenkamp *et al.*, 2024; Geng *et al.*, 2024b] while enhancing the generalizability of L2M2 across different scenarios.

To validate our approach, we conduct comprehensive experiments across progressively challenging scenarios in VMAS [Bettini *et al.*, 2022] and MOSMAC [Geng *et al.*,

*Corresponding authors.

2024a; Geng *et al.*, 2025] environments, ranging from basic particle navigation to strategic navigation through obstacle-rich terrains in StarCraft II maps. Our evaluation compares L2M2 against various baselines including classic non-hierarchical MARL algorithms and hierarchical frameworks, such as HiSOMA [Geng *et al.*, 2024b] and MOSMAC's multi-agent model with rule-based controller (RBC). The results demonstrate L2M2's significant advantages across multiple dimensions. In VMAS scenarios, L2M2 achieves comparable performance to baseline methods while using only 15-20% of training samples, demonstrating remarkable sample efficiency. On complex MOSMAC scenarios with the default pre-defined subgoals, L2M2 achieves a 98.75% success rate, outperforming both RBC (83.13%) and HiSOMA (94.38%). Most notably, while baseline methods struggle in the challenging MOSMAC scenarios without pre-defined subgoals, L2M2 maintains a 68.13% success rate through effective *zero-shot planning* [Dalal *et al.*, 2023] and *policy transfer* using pre-trained MARL models from the MOSMAC complex scenario with subgoals. Analysis using kernel density estimation reveals that L2M2's LLM agent consistently generates strategic navigation paths that avoid challenging areas, demonstrating effective high-level planning without domain expertise. These results highlight L2M2's potential as a generalizable framework for scaling up multi-agent systems towards long-horizon cooperative multi-agent navigation problems. The main contributions of this work are summarized as follows:

1. We propose L2M2, a novel hierarchical framework integrating LLM-based high-level planning with MARL-based execution, where an environment translator enables robust LLM-MARL coordination.
2. We introduce a zero-shot planning approach where pre-trained LLMs directly guide MARL agents, supporting both end-to-end training and integration with existing MARL policies while reducing the cost of repetitively training high-level controllers for different tasks.
3. Experimental results on VMAS and MOSMAC demonstrate L2M2's effectiveness, achieving superior performance with only 15-20% of baseline training samples and maintaining high success rates (98.75% with pre-defined waypoints, 68.13% without) where baseline methods fail.

The rest of this paper is organized as follows. In Section 2, we discuss the relevant literature in the field. The L2M2 framework is detailed in Section 3. Section 4 presents the VMAS and MOSMAC benchmark environments. Section 5 discusses the experimental results. Section 6 concludes with a discussion of our findings and future work.

2 Related Work

2.1 LLM in Multi-agent Frameworks

Recent studies have demonstrated two primary approaches for integrating LLMs into multi-agent systems: *LLM-as-policy* and *LLM-as-enhancement*. The *LLM-as-policy* approach employs LLMs as decision-making policies for *LLM agents* , similar to neural networks in deep reinforcement learning agents. This approach leverages natural language as a communication protocol, enabling agents to incorporate perception, memory, and execution modules [Zhang *et al.*, 2023]. In contrast, the *LLM-as-enhancement* approach uses LLMs to generate supplementary signals such as rewards that influence the agents' decision-making process. For a comprehensive review of this approach, we refer interested readers to a recent survey by Cao et al. [2024]. Multi-agent architectures on the *LLM-as-policy* approach have evolved along two main paths: the pure *multi-LLM-agent* paradigm [Li *et al.*, 2023a; Zhang *et al.*, 2023; Liu *et al.*, 2024b; Mandi *et al.*, 2024; Huang *et al.*, 2025] and the *mixed multi-LLM/RL-agent* paradigm [Li *et al.*, 2023b; Liu *et al.*, 2024a]. While pure systems focus on interactions between multiple LLM agents, mixed systems explore how reinforcement learning [Li *et al.*, 2023b] or executable scripts [Yu *et al.*, 2023] can complement LLM capabilities. This work introduces a novel perspective by investigating how language models can enhance the capabilities of low-level MARL agents through a *mixed multi-LLM/RL framework*, where the LLM agent follows the *LLM-as-policy* approach.

2.2 LLM-guided Reinforcement Learning

The mixed multi-LLM/RL-agent paradigm encompasses two complementary research directions. The first focuses on enhancing LLM agents through learned execution policies for plans and subtasks, while the second aims to improve the efficiency of reinforcement learning through *LLM-guided policy learning*. While substantial research has focused on the former, emerging studies investigate how LLMs can strengthen existing MARL algorithms. In the single-RL-agent domain, Dalal et al. [2023] showed the potential of using LLMs as high-level task planners to guide low-level policy training for robotic tasks. However, the extension to multi-agent scenarios remains largely unexplored. Specifically, L2M2 addresses the unexplored challenge of using LLMs to coordinate multiple reinforcement learning agents in complex navigation tasks, thereby filling a significant gap in the current literature on LLM-MARL integration.

2.3 MARL and Hierarchical Multi-agent Frameworks

MARL has made substantial progress in both on-policy and off-policy algorithms in recent years. Representative on-policy methods such as COMA [Foerster *et al.*, 2018], MAA2C [Papoudakis *et al.*, 2021], and MAPPO [Yu *et al.*, 2022], along with off-policy methods including MADDPG [Lowe *et al.*, 2017], VDN [Sunehag *et al.*, 2018], QMIX [Rashid *et al.*, 2018], QTRAN [Son *et al.*, 2019], and MAVEN [Mahajan *et al.*, 2019], have demonstrated strong performance on cooperative tasks. Building on these foundations, recent research has focused on scaling multi-agent systems to address long-horizon tasks. This progression, while promising, has intensified several key challenges, particularly temporal credit assignment [Jiang and Agarwal, 2018] and catastrophic forgetting [Elsayed and Mahmood, 2023]. Consequently, while existing MARL algorithms perform effectively in short-horizon scenarios such as SMAC [Samvelyan *et al.*, 2019], they may face significant challenges when dealing with extended operation horizons that require sustained coordination and planning.

Hierarchical approaches have emerged as a promising direction for addressing these challenges. Methods such as

Feudal Multiagent Hierarchies (FMH) [Ahilan and Dayan, 2019] and HAMMER [Gupta *et al.*, 2023] demonstrate how task decomposition can enhance multi-agent capabilities. HiSOMA [Geng *et al.*, 2024b] recently advances this approach by implementing a three-level control hierarchy that integrates multiple groups of RL agents trained through centralized MARL algorithms. This hierarchical framework has successfully tackled challenging long-horizon navigation problems that proved difficult for non-hierarchical approaches, highlighting the potential of hierarchical control structures to improve sample efficiency in policy learning for low-level RL agents.

3 The L2M2 Framework

We present L2M2, a hierarchical multi-agent framework that integrates large language models (LLMs) for high-level strategic planning with multi-agent reinforcement learning (MARL) for low-level execution. Our framework addresses the challenges of long-horizon task decomposition and coordination without requiring domain-specific heuristics or pre-defined subtasks. The framework consists of two primary components: (1) a high-level LLM agent that generates strategic plans and subtask allocations, and (2) multiple low-level RL agents that execute primitive actions to accomplish these subtasks. The system leverages pre-trained LLMs [Llama Team, 2024] for *zero-shot* planning while enabling both end-to-end training and direct integration with existing MARL policies.

3.1 Framework Overview

L2M2 implements a two-level hierarchy where a single LLM agent coordinates multiple RL agents through *temporally extended* [Vezhnevets *et al.*, 2017] macro-actions. For the high-level component, we employ the representative Llama 3.1-8B model [Llama Team, 2024], though our framework remains compatible with other LLMs. The low-level agents utilize state-of-the-art centralized training decentralized execution (CTDE) MARL algorithms, representatively QMIX [Rashid *et al.*, 2018]. To facilitate efficient interaction between components, we implement a generalizable communication protocol comprising an environment translator ω and a subtask allocation mechanism. This modular design ensures flexibility, allowing each component to be implemented using any suitable model within its respective domain.

3.2 The Large Language Model Agent

The LLM agent serves as a high-level planner that performs *zero-shot* [Dalal *et al.*, 2023] strategic planning by processing global environmental states s^{LLM} and generating coordinated task allocations a^{LLM} for subordinate RL agents. We adopt the *LLM-as-policy* [Carta *et al.*, 2023; Szot *et al.*, 2024; Zhou *et al.*, 2024] approach, utilizing LLMs as policy networks π^{LLM} that receive input states and produce output actions in textual forms. The LLM agent operates at a lower frequency than RL agents, which execute primitive actions at each timestep. Figure 1 illustrates this control process.

State Representation

Let $\omega : S^{Env} \rightarrow P^{LLM}$ be the *environment translator* function that maps environmental states into *environmental prompts*, where S^{Env} and P^{LLM} are the environmental state space and the environmental prompt space respectively. At time t, the *environmental state* $s_t^{Env} \in S^{Env}$ is defined as:

$$s_t^{Env} = (c_t, b_t, g_t, d_t, d_{t+1}) \tag{1}$$

where:

- c_t is the current coordinates of n RL agents
- b_t indicates obstacles in eight directions
- g_t represents available movement subtasks
- d_t denotes the current distances to the final target
- d_{t+1} indicates predicted distances after taking each potential subtask

The ω converts s_t^{Env} into an environmental prompt:

$$p_t^{LLM} = (c_t^{LLM}, b_t^{LLM}, g_t^{LLM}, d_t^{LLM}, d_{t+1}^{LLM}, f) \tag{2}$$

where $c_t^{LLM}, b_t^{LLM}, g_t^{LLM}, d_t^{LLM}, d_{t+1}^{LLM}$ represent the natural language descriptions of their corresponding elements in s_t^{Env}, and f specifies the format of LLM's response.

Action Space and Decision Making

The LLM agent generates *temporally abstracted* movement actions as *subtasks* for RL agents, illustrated in Figure 2.

The action a_t^{LLM} at time t is defined as:

$$a_t^{LLM} = \{g_{t+1}^i \in G \mid i \in \{1, \ldots, n\}\} \tag{3}$$

where G is the discrete set of available subtasks for the next timestep $t+1$ and n is the number of RL agents. To reduce computational overhead, L2M2 generates a joint plan for all RL agents in a single prompting round.

Feedback Mechanism

We implement a *negative-feedback-only* mechanism enabling the LLM agent to perform *in-context learning*. The verification process on a^{LLM} is a two-step procedure that includes:

1. **Hard verification**: Checks output format compliance as specified by f in the prompt p^{LLM}
2. **Soft verification**: Ensures selected actions in a^{LLM} are within the available action space, reducing hallucinations

When verification fails, a *self-correction* procedure initiates another action selection round with error descriptions p^{Err} incorporated into the environment prompt p^{LLM}, allowing the LLM agent to reflect and correct its mistakes.

3.3 The Reinforcement Learning Agents

RL agents execute primitive actions that directly interact with the environment based on subtasks g assigned by the LLM agent. They maintain *information-receiving channels* in their observation space to process incoming communicative information from the high-level agent.

Observation and Action Spaces

At timestep t, RL agent i receives a partial observation:

$$o_t^i = (o_t^{e,i}, o_t^{g,i}) \tag{4}$$

where $o_t^{e,i}$ represents general environment information, and $o_t^{g,i}$ encapsulates subtask-related information from the information-receiving channel. A translation module $\omega^{RL} : G^i \rightarrow O^{g,i}$ maps subtask information regarding g_{t+1}^i into the

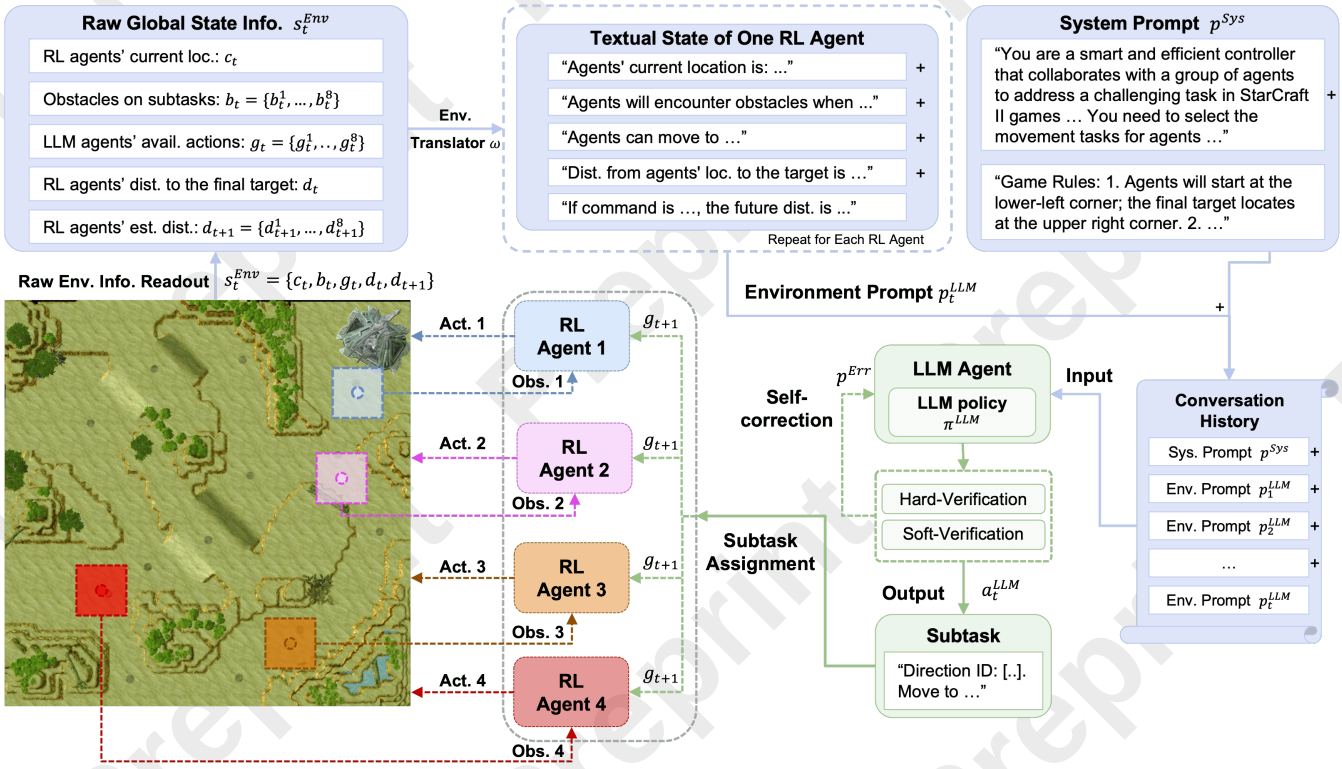


Figure 1: The L2M2 framework exemplified with one LLM agent and four RL agents on the MOSMAC complex scenario.

RL agents' observation space, ensuring effective communication of high-level strategic decisions.

The action space of RL agents remains identical to non-hierarchical MARL algorithms on the benchmark environments. For example, actions in the MOSMAC environment include *no-op*, *move* to four directions and *stop*.

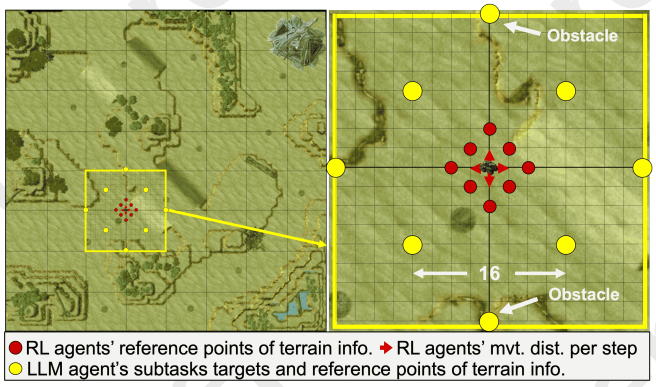


Figure 2: LLM agent's action space for controlling a single unit. Yellow circles represent selectable movement subtasks in eight directions. North and south directions are blocked by obstacles. Red circles indicate locations where RL agents sense terrain information. The red arrow illustrates the distance of a single RL unit movement.

Reward Structure

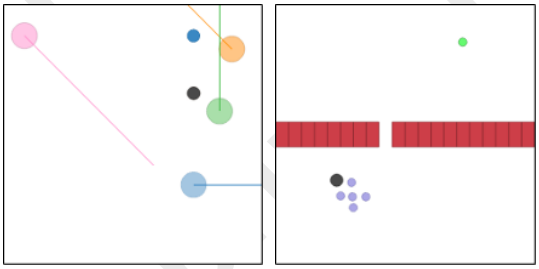
The reward function for RL agent i at timestep t combines environmental and subtask-related rewards:

$$r_t^i = r_t^{e,i} + r_t^{g,i} \tag{5}$$

where $r_t^{e,i}$ is the environmental reward and $r_t^{g,i}$ is the subtask-related reward. This composite reward structure encourages RL agents to balance immediate environmental feedback with progress towards their assigned subtasks.

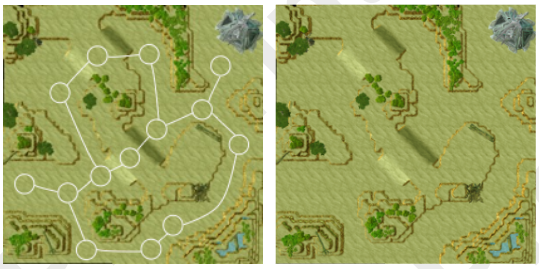
4 Benchmark Domains

To demonstrate the effectiveness of L2M2, this work compares L2M2 against baseline methods on two popular multi-agent environments, including VMAS [Bettini *et al.*, 2022], a vectorized 2D physics environment that simulates the movements of multiple particles, and StarCraft II [Vinyals *et al.*, 2017], a popular real-time strategy (RTS) game environment where agents need to learn to move and attack enemy units. For each environment, we adopt two tasks closely related to the multi-agent navigation problem. For the VMAS environment, we adopt the navigation and passage scenarios. For the StarCraft II environment, we adopt two long-horizon multi-agent navigation scenarios with complex terrain from MOSMAC [Geng *et al.*, 2025], which provides a more realistic and challenging setting. In both environments, agents share the same goal and need to reach the goal collaboratively. The following subsections detail the specific characteristics of each environment.



(a) Navigation (b) Passage

Figure 3: The VMAS navigation and passage scenarios implemented in this study. In the *navigation* scenario, four agents collaborate to reach a shared goal. The *passage* scenario features five agents pursuing a shared goal, with the map bisected by a barrier containing a single randomly-positioned passage. Hierarchical multi-agent frameworks select intermediate subgoals (grey spheres) to guide RL agents toward the goal through sequential navigation instructions.



(a) Complex w. subgoals (b) Complex w/o subgoals

Figure 4: Long-horizon multi-agent MOSMAC scenarios implemented in this study. In each scenario, four units spawn in the bottom-left region and navigate to the goal location (denoted by a building) in the upper-right region of a 128×128 map. For the scenario with subgoals, white spheres indicate pre-defined intermediary waypoints, with connecting lines representing traversable paths.

4.1 The VMAS Environment

VMAS (Vectorized Multi-Agent Simulator) [Bettini *et al.*, 2022] provides a two-dimensional physics-based platform for simulating multi-agent interactions through particle-based dynamics. Among its scenarios, the *navigation* and *passage* scenarios (Figure 3) are particularly relevant for evaluating multi-agent navigation problems. The *navigation* scenario presents an obstacle-free environment where agents must cooperatively reach a shared goal position, while the *passage* scenario extends this by introducing a barrier that divides the map into two sections, with agents starting in the lower section and targeting a randomly placed goal in the upper section.

4.2 The MOSMAC Environment

The MOSMAC (Multi-Objective SMAC) benchmark [Geng *et al.*, 2024a; Geng *et al.*, 2025] (Figure 4) extends the StarCraft II Multi-Agent Challenge (SMAC) [Samvelyan *et al.*, 2019] towards long-horizon and multi-objective problems. Our study focuses on MOSMAC scenarios featuring long-horizon multi-agent navigation: *4t_vs_0t_large_flat* and *4t_vs_0t_large_complex* (referred to as 'flat' and 'complex' hereafter), where four siege tank units navigate on a 128×128 map. Complex terrain scenarios incorporate challenging features such as high/low-ground elevation changes, ramps, cliffs, and dead-end corners that test agents' navigation capabilities in difficult areas. In addition to the original pre-defined scenario-specific subtasks, we also evaluate scenarios without these subgoals to create a more challenging environment that better reflects real-world applications.

5 Experiments and Results

We evaluate L2M2 against state-of-the-art baselines across progressively challenging scenarios in VMAS and MOSMAC environments, ranging from basic particle navigation to strategic navigation through obstacle-rich environments. The benchmark scenarios can be categorized based on whether agents should follow pre-defined subtasks, which influence the action space available to high-level controllers.

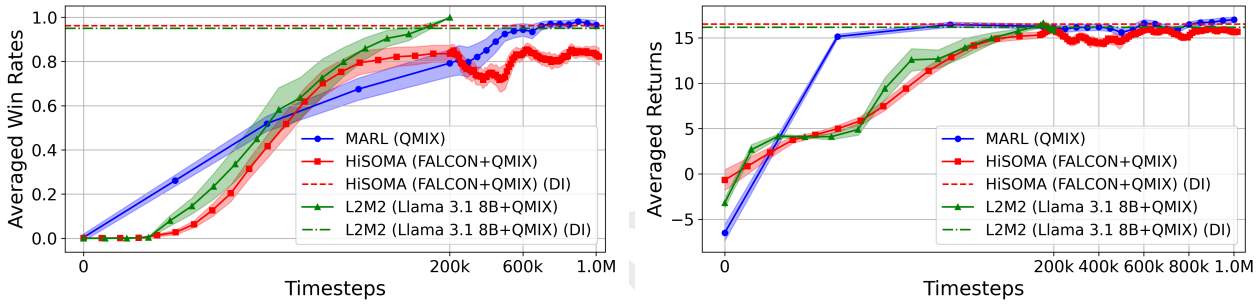
5.1 Baselines

For all baseline methods and L2M2, we employ QMIX [Rashid *et al.*, 2018] as the base algorithm. QMIX is a state-of-the-art off-policy MARL algorithm that combines individual agent Q-values into a global Q-value through a mixing network. QMIX inspired several advanced algorithms including QTRAN [Son *et al.*, 2019] and MAVEN [Mahajan *et al.*, 2019], and has demonstrated superior performance on MOSMAC scenarios [Geng *et al.*, 2024a]. While on-policy algorithms like MAA2C [Papoudakis *et al.*, 2021] and MAPPO [Yu *et al.*, 2022] could also be used for end-to-end training, their lower sample efficiency requires parallel training, which significantly increases computational costs for parallel LLM inference. Therefore, for on-policy algorithms, we focus on direct integration (DI) results where pre-trained MARL policies are combined with high-level controllers without additional training. All evaluations on L2M2 use Llama 3.1-8B [Llama Team, 2024] as the policy for its high-level controller. We set the temperature to 0 and limited the output to 150 tokens to ensure consistency.

We adopt HiSOMA [Geng *et al.*, 2024b] as our primary hierarchical baseline. HiSOMA implements a multi-level control hierarchy where *Fusion Architecture for Learning, Cognition, and Navigation* (FALCON) [Tan, 2007] functions as a central higher-level controller that coordinates multiple groups of MARL agents. FALCON is based on the three-channel Fusion Adaptive Resonance Theory model, a type of self-organizing neural network (SONN) that learns state-action-reward associations for optimal control. Although HiSOMA was originally evaluated on MOSMAC with multiple groups of agents, we adapted it to our study by configuring a single FALCON controller to guide one group of RL agents through various navigation tasks, as a fair baseline for L2M2.

Scenarios without Pre-defined Subgoals

For scenarios without pre-defined subgoals, we evaluate L2M2 against both hierarchical and non-hierarchical methods through two approaches: end-to-end training of all methods, and direct integration of high-level controllers with MARL policies trained in a non-hierarchical manner. To ensure a fair comparison, all high-level controllers are configured with



(a) Averaged Win Rates

(b) Averaged Returns

Figure 5: Results of the evaluated methods including MARL (QMIX), HiSOMA (FALCON+QMIX), and the proposed L2M2 (Llama 3.1 8B) on the VMAS navigation scenario. The direct integration results for HiSOMA and L2M2 are represented as horizontal dotted lines.

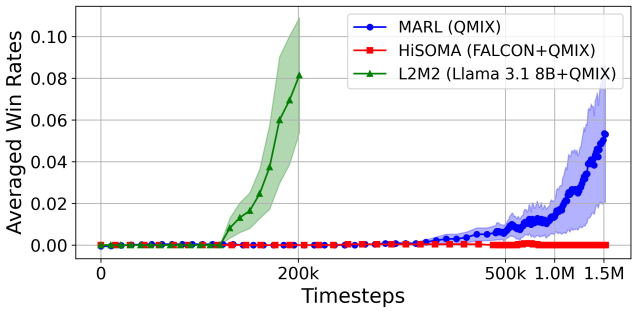


Figure 6: The averaged win rates for L2M2, non-hierarchical MARL (QMIX) and HiSOMA on the VMAS passage scenario.

enhanced environmental interaction capabilities, with observation and action ranges set to eight times those of low-level RL agents (Figure 2). Following Geng et al. [2024b], we train HiSOMA's FALCON controller across four simulated scenarios that mirror the tasks for high-level controllers in VMAS and MOSMAC scenarios before integrating with MARL.

Scenarios with Subgoals

Scenarios with subgoals present a distinct evaluation setting, as they require agents to complete pre-defined subtasks in sequence. The MOSMAC benchmark provides a baseline method known as RBC-MARL, which employs a randomized path selector to guide units through strategic points toward their final goal (Figure 4b). This built-in method is essentially a hierarchical end-to-end training method where a path selector coordinates MARL agents through sequential subtask allocations. Given that pre-defined subtask selection is inherently a hierarchical control problem, we focus our evaluation on comparing L2M2 against other hierarchical approaches - specifically HiSOMA and the RBC-MARL baseline. We further demonstrate L2M2's effectiveness by showing it can directly utilize MARL policies trained with RBC without fine-tuning. This highlights its zero-shot adaptation capabilities.

5.2 Evaluation Protocols

We conduct comprehensive evaluations comparing L2M2 against established baselines while maintaining consistent hyperparameter settings across MARL algorithms. For non-hierarchical baselines and HiSOMA's end-to-end training, we allocate 1 million steps for VMAS navigation, 1.5 million steps for VMAS passage, and 20 million steps for MOSMAC complex terrain scenarios. Given L2M2's superior sample efficiency, we restrict its MARL training to fewer steps: 200k for VMAS navigation, 400k for VMAS passage, and 100k timesteps for MOSMAC scenarios, which reduces computational costs associated with LLM inference while maintaining performance. For end-to-end training evaluation, we assess all methods across 32 episodes at fixed intervals throughout the training process. For hierarchical methods using pre-trained MARL policies, we conduct five independent evaluations, each utilizing a MARL model trained with a different seed. Performance assessment relies on two standard MARL metrics: averaged win rates [Samvelyan *et al.*, 2019] and averaged episode returns [Papoudakis *et al.*, 2021], providing a comprehensive view of both task completion and reward optimization.

5.3 Results on the VMAS Scenarios

Figure 5 presents the results of L2M2 and the baseline methods on the VMAS navigation scenario. The non-hierarchical QMIX baseline achieves a 95.63% win rate within 1M steps. When integrated as the low-level controllers in hierarchical frameworks, it enables both HiSOMA and L2M2 to maintain comparable performance (96.25% and 95.00% win rates, respectively), demonstrating effective planning capabilities of both FALCON and Llama 3.1-8B controllers.

Notable differences emerge in end-to-end training. FALCON-guided training converges to an 84.38% win rate, suggesting a potential difficulty of policy learning under the commands of HiSOMA. In contrast, L2M2-guided training achieves a 94.79% win rate, suggesting more consistent and learnable guidance from the LLM controller. Most significantly, L2M2 demonstrates superior sample efficiency, converging to QMIX-level performance within 200k steps - only one-fifth of the samples required by non-hierarchical MARL.

These benefits become more pronounced in the more challenging VMAS passage scenario, where agents must learn sequential navigation through a passage to reach their goals (Figure 3b). As shown in Figure 6, L2M2 on average achieves an 8.75% win rate within 200k steps, while non-hierarchical

Metrics	Rule-based Control	HiSOMA (DI)	L2M2 (DI)
	RBC w. QMIX	FALCON w. QMIX	Llama 3.1 8B w. QMIX
Avg. Win Rates (%) (mean and std)	83.13 ± 10.96	94.38 ± 1.40	98.75 ± 2.80
Avg. Returns (mean and std)	9.45 ± 0.85	10.82 ± 0.13	9.92 ± 0.21

Table 1: The results of the L2M2 and two baseline methods on the MOSMAC complex scenario with subgoals. The rule-based controller (RBC) is the default target selector of MOSMAC, which randomly selects one of three paths for each episode.

Metrics	End-to-End	Policy Transfer	
	MARL	HiSOMA (DI)	L2M2 (DI)
Avg. Win Rates (%) (mean and std)	0.00 ± 0.00	14.68 ± 14.14	68.13 ± 25.14
Avg. Returns (mean and std)	0.73 ± 0.09	7.65 ± 2.49	9.39 ± 2.06

Table 2: Performance comparison on the MOSMAC complex scenario without subgoals. Both HiSOMA and L2M2 utilize a *policy transfer* approach, leveraging pre-trained MARL models from the MOSMAC complex scenario with subgoals and integrating them with their respective high-level controllers.

MARL requires 1.5M steps to reach a lower 6.25% win rate. While HiSOMA encounters significant challenges in *sim-to-real* policy transfer, L2M2 achieves robust performance through its *zero-shot planning* [Dalal *et al.*, 2023] capabilities, demonstrating its ability to effectively coordinate and guide multi-agent learning on complex tasks.

5.4 Results on the MOSMAC Scenarios

We first evaluate L2M2 on the default complex scenario with pre-defined subgoals, where a rule-based controller (RBC) selects from three pre-defined paths connecting start and goal locations. As presented in Table 1, end-to-end training with RBC-controlled QMIX agents achieves an average win rate of 83.13%. When integrating the trained QMIX policies with high-level controllers, both HiSOMA and L2M2 demonstrate significant improvements, achieving win rates of 94.38% and 98.75% respectively. These improvements can be attributed to enhanced planning capabilities that help agents avoid local optima and challenging terrain features that often trap RBC-controlled agents. Notably, L2M2 exhibits enhanced planning capabilities compared to HiSOMA's pre-trained controller, demonstrating its efficient zero-shot planning ability.

We further evaluate L2M2 on a more challenging variant without pre-defined subgoals. In this scenario, non-hierarchical MARL algorithms struggle to learn effective navigation behaviour, resulting in poor performance of low-level controllers that preclude direct integration with high-level controllers. To address this limitation, we adopt a *policy transfer* approach, leveraging MARL models pre-trained in scenarios with subgoals. While both scenarios share identical terrains, they present distinct hidden environment transition functions and state distributions, creating a challenging *zero-shot* transfer setting. As presented in Table 2, L2M2 achieves a win rate of 68.13%, substantially outperforming HiSOMA, suggesting the efficiency of combining generalizable subtask-based MARL policies with LLMs in zero-shot settings.

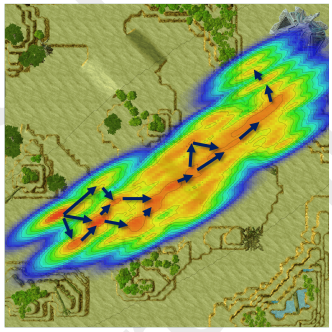


Figure 7: Target locations and example actions (denoted by arrows) selected by the LLM agent in the MOSMAC Complex scenario without pre-defined subgoals. Regions with warmer colours denote higher frequencies that were selected by the LLM agent.

5.5 Analysis on LLM Agent's Actions in L2M2

To further evaluate L2M2's capabilities, we analyzed the spatial distribution of the LLM agent's action selections using kernel-density estimation (KDE) with Gaussian kernels on the MOSMAC complex scenario without pre-defined subgoals, as illustrated in Figure 7. Notably, the LLM agent consistently generates navigation commands that avoid challenging terrain features such as cliffs, ramps, and elevation changes, which would otherwise trap the RL agents. Instead, it guides agents through the terrain's central region - a path that experienced human players would recognize as optimal for this navigation task. This spatial analysis demonstrates the LLM agent's ability to generate contextually appropriate subtask signals for the RL agents without requiring domain-specific training.

6 Conclusion and Future Work

This paper introduces L2M2, a novel hierarchical multi-agent framework that integrates LLMs with MARL to address complex, long-horizon multi-agent tasks. Our approach demonstrates superior performance and sample efficiency, requiring only 15-20% of the training samples needed by baseline methods while achieving comparable results. The effectiveness of L2M2 stems from its zero-shot planning mechanism, which enables an LLM agent to generate contextually appropriate subtasks for MARL agents without repetitive retraining.

While demonstrating strong performance, L2M2 faces several limitations, including computational overhead from LLM and constraints on the LLM agent's discrete action space. Looking forward, several promising research directions emerge from this work. These include exploring LLMs' environmental perception for automatic target identification, incorporating feedback mechanisms for LLM fine-tuning, and scaling to larger multi-agent systems through more sophisticated coordination protocols. These advancements could significantly expand the capabilities of integrated LLM-MARL systems while maintaining the sample efficiency and generalization benefits demonstrated in this work.

Acknowledgments

This research was supported in part by Tsinghua University - Migu Xinkong Culture Technology (Xiamen) Co., Ltd. Joint Research Center for Intelligent Light Field and Interactive Technology, Beijing, under its Joint research and development project (R24119F0) - Phase 1, the National Research Foundation, Singapore, under its AI Singapore Programme (AISG Award No: AISG2-RP-2020-019), and the Lee Kong Chian Professorship awarded to Ah-Hwee Tan by Singapore Management University.

Contribution Statement

Minghong Geng: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Visualization, Writing - original draft, Writing - review & editing; **Shubham Pateria**: Software, Writing - review & editing; **Budhitama Subagdja**: Software, Writing - review & editing; **Lin Li**: Resources; **Xin Zhao**: Writing - review & editing, Funding Acquisition, Supervision; **Ah-Hwee Tan**: Conceptualization, Methodology, Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition.

References

[Agarwal *et al.*, 2023] Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Legged Locomotion in Challenging Terrains using Egocentric Vision. In *CoRL 2023*, volume 205 of *Proc. Mach. Learn. Res.*, pages 403–415. PMLR, 2023.

[Ahilan and Dayan, 2019] Sanjeevan Ahilan and Peter Dayan. Feudal Multi-Agent Hierarchies for Cooperative Reinforcement Learning. *arXiv:1901.08492v1 [cs.MA]*, January 2019.

[Bettini *et al.*, 2022] Matteo Bettini, Ryan Kortvelesy, Jan Blumenkamp, and Amanda Prorok. VMAS: A Vectorized Multi-agent Simulator for Collective Robot Learning. In *DARS 2022*, pages 42–56. Springer, November 2022.

[Blumenkamp *et al.*, 2024] Jan Blumenkamp, Ajay Shankar, Matteo Bettini, Joshua Bird, and Amanda Prorok. The Cambridge RoboMaster: An Agile Multi-Robot Research Platform. In *DARS 2024*. Springer, 2024.

[Cao *et al.*, 2024] Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on Large Language Model-Enhanced Reinforcement Learning: Concept, Taxonomy, and Methods. *IEEE Trans. Neural Netw. Learn. Syst.*, pages 1–21, November 2024.

[Carta *et al.*, 2023] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *ICML 2023*, volume 202, pages 3676–3713. JMLR.org, July 2023.

[Dalal *et al.*, 2023] Murtaza Dalal, Tarun Chiruvolu, Devendra Singh Chaplot, and Ruslan Salakhutdinov. Plan-Seq-Learn: Language Model Guided RL for Solving Long Horizon Robotics Tasks. In *ICLR 2024*, October 2023.

[Ellis *et al.*, 2023] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. SMACv2: An improved benchmark for cooperative multi-agent reinforcement learning. In *Adv. Neural Inf. Process. Syst.*, volume 36, pages 37567–37593. Curran Associates, Inc., 2023.

[Elsayed and Mahmood, 2023] Mohamed Elsayed and A. Rupam Mahmood. Addressing Loss of Plasticity and Catastrophic Forgetting in Continual Learning. In *ICLR 2023*, October 2023.

[Foerster *et al.*, 2018] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI 2018*, pages 2974–2982. AAAI Press, February 2018.

[Geng *et al.*, 2024a] Minghong Geng, Shubham Pateria, Budhitama Subagdja, and Ah-Hwee Tan. Benchmarking MARL on Long Horizon Sequential Multi-Objective Tasks. In *AAMAS 2024*, pages 2279–2281. IFAAMAS, May 2024.

[Geng *et al.*, 2024b] Minghong Geng, Shubham Pateria, Budhitama Subagdja, and Ah-Hwee Tan. HiSOMA: A hierarchical multi-agent model integrating self-organizing neural networks with multi-agent deep reinforcement learning. *Expert Syst. Appl.*, 252:124117, October 2024.

[Geng *et al.*, 2025] Minghong Geng, Shubham Pateria, Budhitama Subagdja, and Ah-Hwee Tan. MOSMAC: A Multi-agent Reinforcement Learning Benchmark on Sequential Multi-objective Tasks. In *AAMAS 2025*, pages 867–876. IFAAMAS, May 2025.

[Gupta *et al.*, 2023] Nikunj Gupta, G. Srinivasaraghavan, Swarup Mohalik, Nishant Kumar, and Matthew E. Taylor. HAMMER: Multi-level coordination of reinforcement learning agents via learned messaging. *Neural Comput. Appl.*, October 2023.

[Hernandez-Leal *et al.*, 2019] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A Survey of Learning in Multiagent Environments: Dealing with Non-Stationarity. *arXiv:1707.09183v2 [cs.MA]*, March 2019.

[Huang *et al.*, 2025] Jen-tse Huang, Eric John Li, Man Ho Lam, Tian Liang, Wenxuan Wang, Youliang Yuan, Wenxiang Jiao, Xing Wang, Zhaopeng Tu, and Michael Lyu. Competing Large Language Models in Multi-Agent Gaming Environments. In *ICLR 2025*, April 2025.

[Jiang and Agarwal, 2018] Nan Jiang and Alekh Agarwal. Open Problem: The Dependence of Sample Complexity Lower Bounds on Planning Horizon. In *COLT 2018*, volume 75 of *Proc. Mach. Learn. Res.*, pages 3395–3398. PMLR, July 2018.

[Li *et al.*, 2023a] Huao Li, Yu Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. Theory of Mind for Multi-Agent Collaboration via Large Language Models. In *EMNLP 2023*, pages 180–192. ACL, December 2023.

[Li *et al.*, 2023b] Wenhao Li, Dan Qiao, Baoxiang Wang, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Semantically

Aligned Task Decomposition in Multi-Agent Reinforcement Learning. *arXiv:2305.10865v2 [cs.LG]*, May 2023.

[Liu *et al.*, 2024a] Jijia Liu, Chao Yu, Jiaxuan Gao, Yuqing Xie, Qingmin Liao, Yi Wu, and Yu Wang. LLM-Powered Hierarchical Language Agent for Real-time Human-AI Coordination. In *AAMAS 2024*, pages 1219–1228. IFAAMAS, May 2024.

[Liu *et al.*, 2024b] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic LLM-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024.

[Llama Team, 2024] Llama Team. The Llama 3 Herd of Models. *arXiv:2407.21783v3 [cs.AI]*, 2024.

[Lowe *et al.*, 2017] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Adv. Neural Inf. Process. Syst.*, volume 30, pages 6379–6390. Curran Associates Inc., December 2017.

[Mahajan *et al.*, 2019] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. MAVEN: Multi-Agent Variational Exploration. In *Advances in Neural Information Processing Systems*, volume 32, Red Hook, NY, USA, 2019. Curran Associates, Inc.

[Mandi *et al.*, 2024] Zhao Mandi, Shreeya Jain, and Shuran Song. RoCo: Dialectic Multi-Robot Collaboration with Large Language Models. In *ICRA 2024*, pages 286–299, May 2024.

[Oroojlooy and Hajinezhad, 2023] Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Appl. Intell.*, 53(11):13677–13722, June 2023.

[Papoudakis *et al.*, 2021] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proc. Neural Inf. Process. Syst. Track on Datasets and Benchmarks*, volume 1. Curran Associates Inc., November 2021.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *ICML 2018*, volume 80 of *Proc. Mach. Learn. Res.*, pages 4295–4304. PMLR, July 2018.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft Multi-Agent Challenge. In *AAMAS 2019*, pages 2186–2188. IFAAMAS, May 2019.

[Son *et al.*, 2019] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Hostallero, and Yung Yi. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *ICML 2019*, volume 97 of *Proc. Mach. Learn. Res.*, pages 5887–5896. PMLR, 2019.

[Sunehag *et al.*, 2018] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *AAMAS 2018*, pages 2085–2087. IFAAMAS, July 2018.

[Szot *et al.*, 2024] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R. Devon Hjelm, and Alexander T. Toshev. Large Language Models as Generalizable Policies for Embodied Tasks. In *ICLR 2024*, May 2024.

[Tan, 2007] Ah-Hwee Tan. Direct code access in self-organizing neural networks for reinforcement learning. In *IJCAI 2007*, pages 1071–1076. Morgan Kaufmann Publishers Inc., January 2007.

[Vezhnevets *et al.*, 2017] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal Networks for Hierarchical Reinforcement Learning. In *ICML 2017*, pages 3540–3549. PMLR, July 2017.

[Vinyals *et al.*, 2017] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. StarCraft II: A New Challenge for Reinforcement Learning. *arXiv:1708.04782v1 [cs.LG]*, August 2017.

[Yu *et al.*, 2022] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and YI WU. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In *Adv. Neural Inf. Process. Syst.*, volume 35, pages 24611–24624. Curran Associates, Inc., 2022.

[Yu *et al.*, 2023] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. Co-NavGPT: Multi-Robot Cooperative Visual Semantic Navigation using Large Language Models. *arXiv:2310.07937v2 [cs.RO]*, December 2023.

[Zhang *et al.*, 2019] Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. CityFlow: A Multi-Agent Reinforcement Learning Environment for Large Scale City Traffic Scenario. In *WWW 2019*, pages 3620–3624. ACM, May 2019.

[Zhang *et al.*, 2023] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building Cooperative Embodied Agents Modularly with Large Language Models. In *ICLR 2024*, October 2023.

[Zhou *et al.*, 2024] Zihao Zhou, Bin Hu, Chenyang Zhao, Pu Zhang, and Bin Liu. Large Language Model as a Policy Teacher for Training Reinforcement Learning Agents. In *IJCAI 2024*, pages 5671–5679. IJCAI Organization, August 2024.