2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Language-Guided Pattern Formation for Swarm Robotics with Multi-Agent Reinforcement Learning

Hsu-Shen Liu[1], So Kuroki[2], Tadashi Kozuno[2], Wei-Fang Sun[3], and Chun-Yi Lee[1]

***Abstract*— This paper explores leveraging the vast knowledge encoded in Large Language Models (LLMs) to tackle pattern formation challenges for swarm robotics systems. A new framework, named LGPF (Language-Guided Pattern Formation), is proposed to address these challenges. The framework breaks down the pattern formation into two key components: pattern synthesis and swarm robotics control. For the former, this study utilizes the exceptional few-shot generalizability of LLMs to translate high-level natural language descriptions into the desired spatial pattern coordinates. This approach allows for overcoming previous limitations in representing and designing complex patterns. The framework further employs a centralized training with decentralized execution (CTDE) based multi-agent reinforcement learning (MARL) approach to control the swarm robots in forming the specified pattern while avoiding collisions. The decentralized policies learned with the CTDE-based MARL algorithm consider coordination between robots without direct communication under a partially observable setup. To validate the effectiveness of our framework, we perform extensive experiments in both simulation and real-world environments. These experiments validate LGPF's effectiveness in accurately and safely forming diverse user-specified patterns.**

I. INTRODUCTION

Pattern formation holds a pivotal role in the field of swarm robotics, with applications spanning from drone control [1] to autonomous vehicles in factories, as well as exploration and rescue missions [2]. A primary challenge in these applications lies in enabling robots to form arbitrary patterns. Conventionally, defining complex shapes, along with the calculation of their corresponding coordinates, present significant challenges due to the limited capacity for generalization across a spectrum of intricate geometries. This limitation also extends to the challenges of transforming high-level concepts into patterns due to the absence of an efficient and autonomous strategy. Nevertheless, the emergence of LLMs provides a promising solution. These models, pre-trained with extensive natural language knowledge, can be utilized to assist in generating diverse patterns. Such an approach consequently unveils novel opportunities in swarm robot pattern formation that have not been properly explored.

Previous exploration of pattern formation has primarily concentrated on two main components: goal generation and robot control. For the former, earlier endeavors have been dedicated to presenting shapes through various methodologies and subsequently generating goal coordinates or identifying feasible target regions for robots. Despite the variety of techniques to represent shapes, defining complex shapes remains a concern and calls for the development of more intuitive and flexible approaches to pattern representation and goal generation.

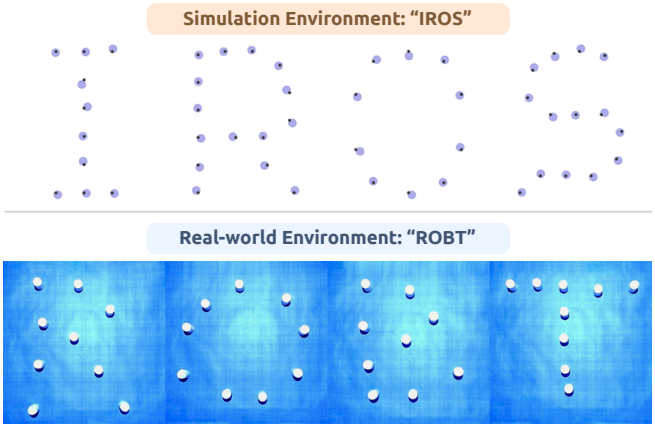

Fig. 1. Swarm robots form the words defined by natural language prompts.

On the other hand, controlling a swarm of robots also requires overcoming various challenges under different constraints. These challenges include (1) employing appropriate assignment algorithms [3] to establish the relationship between agents and their goal positions, (2) ensuring the proper formation and maintenance of patterns, and (3) avoiding collisions [4], [5]. Achieving these typically necessitates comprehensive communication among robots to acquire global information for synchronized actions. However, facilitating communication among all robots introduces complexities associated with synchronization and the real-time management of dynamic topologies [6]. Furthermore, sophisticated algorithms are essential for addressing potential collisions within a large swarm of robots. Simple collision avoidance algorithms, when applied independently to each robot, could be ineffective due to the challenges robots face in coordinating with one another. While more advanced collision avoidance techniques may incorporate holistic group dynamics, they often encounter scalability issues as the addition of more robots increases computational demands, which presents a challenge for robots with limited processing power. Addressing the concerns and developing an integrated solution for swarm robot pattern formation represents a crucial area for exploration and is the focus of this study.

In light of these shortcomings, we propose investigations into two innovative directions. Our first initiative, *knowledge-driven pattern synthesis*, takes advantage of the few-shot generalizability of LLMs. This approach enables efficient and intuitive pattern generation and eliminates the need for

[1] Elsa Lab, National Tsing Hua University, Hsinchu City, Taiwan.
[2] OMRON SINIC X Corporation, Tokyo, Japan.
[3] NVIDIA AI Technology Center (NVAITC), Santa Clara, CA, USA.

manual shape design. Our second avenue, *swarm robotics coordination*, involves implementing the MARL algorithm within a CTDE framework [7], [8], which allows for effective robot control and addresses the control challenges mentioned previously. Regarding the first strategy, prior research has demonstrated that LLMs possess a level of generalizability that is advantageous for single robotic tasks [9]–[13]. Moreover, some single robot case studies have further integrated multi-modal information [14], [15]. To the best of our knowledge, while LLMs have been used in single-robot scenarios, their potential in multi-agent or swarm robotics contexts remains largely unexplored. As a result, this motivates us to investigate the application of LLMs in swarm robotics pattern formation and aims to facilitate the shape generation process. For the latter, we propose to introduce robot control using a CTDE-based MARL approach. The main objective is to allow swarm robot agents to learn collectively under a centralized training scheme, and cooperate without direct communication in challenging settings such as Decentralized Partially Observable Markov Decision Process (Dec-POMDPs) [16], which is common when the communication channel is constrained. By adopting the above strategies, we aim to address swarm robot pattern formation challenges within an integrated framework, including pattern formulation and maintenance, as well as enhancing robot cooperation and collision avoidance, all under the context of POMDP.

To achieve the above directions, we introduce a new LGPF framework to overcome the complexities in formulating shapes and to coordinate swarm robotics control for pattern formation seamlessly. We begin by feeding desired shapes as language prompts into pre-trained LLMs, which then produce the corresponding pattern coordinates for the swarm. These patterns, generated by the LLMs, act as target positions for our CTDE-based robotic agents, and direct them to form the specified shape within the constraints of a POMDP. Fig. 1 illustrates robots forming different alphabets in both simulated and real-world scenarios. To evaluate our framework, we assess the precision and congruence of the patterns formed by LLM-guided swarm robots with the intended shapes using the CLIP similarity score. We conduct experiments in both simulated and real-world environments to test the framework's versatility. In simulated environments, the Multi-Agent Particle Environment (MPE) [17] serves as the basis for evaluation, where we employ various metrics to assess the effectiveness of the swarm robots' learned policies. Moreover, to demonstrate our method's capability to bridge simulation and reality, we implement these policies on actual robots in real-world settings. The quantitative and qualitative outcomes affirm the robots' ability to form the designed patterns as dictated by high-level language prompts, which validate that the learned CTDE-based MARL policies are effectively transferable from simulated environments to real-world ones. Our contributions are summarized as follows:

- We introduce a framework that can leverage the few-shot generalizability of LLMs to guide pattern formation for the swarm from high-level natural language prompts.
- We adopt CTDE-based MARL to enable robots to cooperate effectively without needing direct communication.
- We conduct experiments to verify our framework's generalizability for various pattern formation tasks using language prompts in simulated and real-world scenarios.

II. RELATED WORKS

Many methodologies have been proposed to address pattern formation problems in swarm robotics for organizing robots into specified shapes. These methodologies typically comprise two main stages: goal generation and robot control.

A. Shape Definition and Goal Generation

In general, the desired shapes for pattern formation can be defined using various representation formats, such as mathematical functions [2], [18]–[20], binary maps [21]–[26], and direct coordinates [27], [28]. After defining the shape, several approaches can be utilized to establish intermediate guidance signals. These include generating intermediate goal coordinates [2], [21], [23], computing gradient fields based on the shape [18]–[20], identifying the saliency maps of images [24], and employing other strategies [25], [26]. Unfortunately, each of these shape representation methods has inherent limitations. For instance, defining shapes through mathematical functions can be non-intuitive and is often restricted to simpler shapes. Binary images sometimes require the use of graphical user interfaces, which might be impractical in deployment environments. Moreover, manually designing direct coordinates is both labor-intensive and time-consuming. Therefore, this study explores a novel approach to shape definition: natural language, which supports high-level descriptions through typing or potentially speech-to-text technology. This method offers simplicity and intuitiveness when describing complex shapes. To the best of our knowledge, this has not been properly explored in prior research.

B. Swarm Robotics Control

Given an intermediate goal representation or guidance signals, the robot controller's role is to direct the swarm robots to the target positions or areas. Traditional approaches to swarm robotics control for pattern formation have been investigated in previous works [18]–[20], [22]–[26]. However, centralized control mechanisms, which rely on communication for effective operation, face significant challenges in real-time applications, including issues with synchronization and routing in dynamic topologies [6]. On the other hand, some researchers have explored decentralized control, which offers the advantage of relying solely on local observations and can mitigate the need for real-time communication. Nevertheless, traditional decentralized methods sometimes overlook the importance of joint planning, which can lead to non-collaborative agent behavior. For instance, agents may become stuck when attempting to reach the goal via the shortest path without coordinating their actions. Furthermore, introducing new constraints in these methods frequently necessitates the addition of extra algorithms, which could further complicate the overall development process [29].

With the recent advent of MARL, a new avenue for addressing pattern formation has emerged. MARL offers several benefits to tackle the challenges in traditional methods [27], [28]. With a CTDE-based training framework, robots can learn a joint policy that considers coordination between each other during centralized training, without the need for direct communication during decentralized execution. Moreover, by carefully designing reward functions, various constraints or algorithmic requirements can be incorporated into the learning process, which enables robots to adapt to a wide range of pattern formulation scenarios [30].

III. PRELIMINARY

A. Large Language Models

LLMs aim to predict the probability $p(W)$ of a sequence of n tokens $W = \{w_1, w_2, ..., w_n\}$ derived from an input text. Each token is an element of a predefined vocabulary $\mathcal{W}$. This is typically done by factorizing the joint probability into conditional probabilities using the chain rule: $p(W) = \prod_{k=1}^{n} p_{LLM}\left(w_k | w_1, ..., w_{k-1}\right)$, where p_{LLM} represents the model's learned probability function [10]. Modern models predominantly utilize Transformer [31] for this purpose, with examples including BERT [32], GPT-3 [33], and PaLM [15].

Recent advancements in LLM development have led to significant performance improvements, with models such as ChatGPT-4 [34], Mixtral 8x7b Instruct [35], and LLaMA-2 Chat [36] at the forefront of this progress. These models demonstrate capabilities in understanding and generating natural language and exhibit remarkable zero-shot generalizability. This allows them to tackle unseen tasks without relying on prior examples. Moreover, by providing LLMs with few-shot in-context demonstrations, their performance and reliability on unseen tasks can be significantly enhanced.

B. Dec-POMDP

A *fully decentralized and partially observable multi-agent task* can be described as a Dec-POMDP [16], [37], which is defined by a tuple $\langle S, A, P, r, Z, O, K, \gamma \rangle$. The state set S consists of all possible states s of the environment. At each time step, each agent $i \in B \equiv \{1, ..., K\}$ perceives its local observation $z \in Z$ based on an observation function $O(s, i) : S \times B \rightarrow Z$. Each agent i selects an action $a_i \in A$, which results in a joint action $\mathbf{a} \in \mathbf{A} \equiv A^K$ that triggers the environment to transition to a new state s' based on the state transition function $P(s'|s, \mathbf{a}) : S \times \mathbf{A} \times S \rightarrow [0, 1]$. The reward function $r(s, \mathbf{a}) : S \times \mathbf{A} \rightarrow \mathbb{R}$ is shared by all agents.

Agents possess individual action-observation history $\tau_i \in T \equiv (Z \times A)^*$, on which they condition a stochastic policy $\pi_i(a_i|\tau_i) : T \times A \rightarrow [0, 1]$. The joint policy π possesses a joint *action-value function*: $Q^{\pi}(s_t, \mathbf{a}_t) = \mathbb{E}s_{t+1:\infty}, \mathbf{a}_{t+1:\infty}\left[R_t | s_t, \mathbf{a}_t\right]$, where $R_t = \sum_{j=0}^{\infty} \gamma^j r_{t+j}$ is the *discounted return* with $\gamma \in [0, 1)$ as the discount factor. Although Dec-POMDP requires decentralized policy execution, current methods often adopt CTDE-based frameworks. This allows the use of more information during training, while maintaining the capability for decentralized execution. The learning algorithm can utilize all local histories of action-observation pairs $\boldsymbol{\tau} \in \mathbf{T} \equiv T^K$ and the true state s, while the policy of each agent can depend solely on its own individual action-observation history τ_i.

C. MADDPG

Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [8] stands as one of the pioneering algorithms in MARL. It extends the DDPG algorithm [38] with the CTDE framework to support multi-agent scenarios. The goal of each agent's policy $\pi_i, i \in 1, ..., K$ in MADDPG is formulated as:

$$\max_{\theta_i} \mathop{\mathbb{E}}_{\mathbf{z}, \mathbf{a} \sim \mathcal{D}} \left[Q_i(\mathbf{z}, a_1, ..., a_K)|_{a_i = \pi_i(z_i)} \right], \tag{1}$$

where θ_i is the parameter of π_i, and Q_i is the critic for agent i with parameter ϕ_i. On the other hand, the parameter ϕ_i of each critic Q_i is optimized by the following loss function:

$$L(\phi_i) = \mathop{\mathbb{E}}_{(\mathbf{z}, \mathbf{a}, \mathbf{r}, \mathbf{z}') \sim \mathcal{D}} \left[\left(Q_i(\mathbf{z}, \mathbf{a}) - y_i \right)^2 \right], \tag{2}$$

where the temporal difference (TD) target y_i is defined as:

$$r_i + \gamma Q_{\bar{i}}(\mathbf{z}', a'_1, ..., a'_K)|_{a'_j = \pi_{\bar{j}}(z'_j)}, \tag{3}$$

where $\bar{i}$, $\bar{j}$ indicate that the network uses the delayed parameters $\bar{\phi}_i$, $\bar{\theta}_j$, where $\bar{\phi}_i = \rho \phi_i + (1 - \rho)\bar{\phi}_i$ at each time step, with $\rho \ll 1$. The same update mechanism is applied to $\bar{\theta}_j$.

D. Permutation Invariant Critic (PIC)

In MADDPG, the centralized critic receives joint observations and actions from all agents. When dealing with homogeneous robots, the order of individual observations and actions should not affect the joint Q-value. Therefore, if a critic is permutation-invariant (PI), i.e., it produces the same output regardless of the input order in terms of agent indices, the efficiency of the learning process can be enhanced [39].

The Permutation Invariant Critic (PIC) [39] follows this concept and employs a Graph Convolutional Network (GCN) [40] within the MADDPG framework. A GCN consists of L graph convolutional layers $\{f_{\text{GCN}}^{(1)}, ..., f_{\text{GCN}}^{(L)}\}$. Each layer $f_{\text{GCN}}^{(l)}$ maps the input node features $\mathbf{h}^{(l-1)} \in \mathbb{R}^{K \times D^{(l-1)}}$ to the output features $\mathbf{h}^{(l)} \in \mathbb{R}^{K \times D^{(l)}}$ for the next layer based on the graph's adjacency matrix, where $D^{(l)}$ is the representation dimension for each node at layer l. A permutation-invariant function f_{PI} can be defined as follows:

$$f_{\text{PI}}(M_i \mathbf{x}) = f_{\text{PI}}(M_j \mathbf{x}), \ \ \forall M_i, M_j \in \mathcal{P}, \tag{4}$$

where $\mathbf{x}$ is any valid input, and M_i, M_j are permutation matrices from the set of all possible permutation matrices $\mathcal{P}$. With weight sharing, each GCN layer has the property:

$$f_{\text{GCN}}^{(l)}(M\mathbf{h}^{(l-1)}) = M f_{\text{GCN}}^{(l)}(\mathbf{h}^{(l-1)}), \ \ \forall M \in \mathcal{P}. \tag{5}$$

By applying a PI pooling operation f_{pool}, such as average pooling or max pooling, a PI critic is constructed as follows:

$$Q_{\text{PI}}(\mathbf{z}, \mathbf{a}) = f_v \circ f_{\text{pool}} \circ f_{\text{GCN}}^{(L)} \circ ... \circ f_{\text{GCN}}^{(1)}(\mathbf{h}^{(0)}), \tag{6}$$

where $\mathbf{h}^{(0)} \equiv [\mathbf{z}, \mathbf{a}]$, and f_v maps the input vector to a scalar. Our framework adopts the setup of PIC as the swarm robotics control algorithm due to its learning efficiency. Note that we adopt MADDPG as the MARL baseline for our experiments.

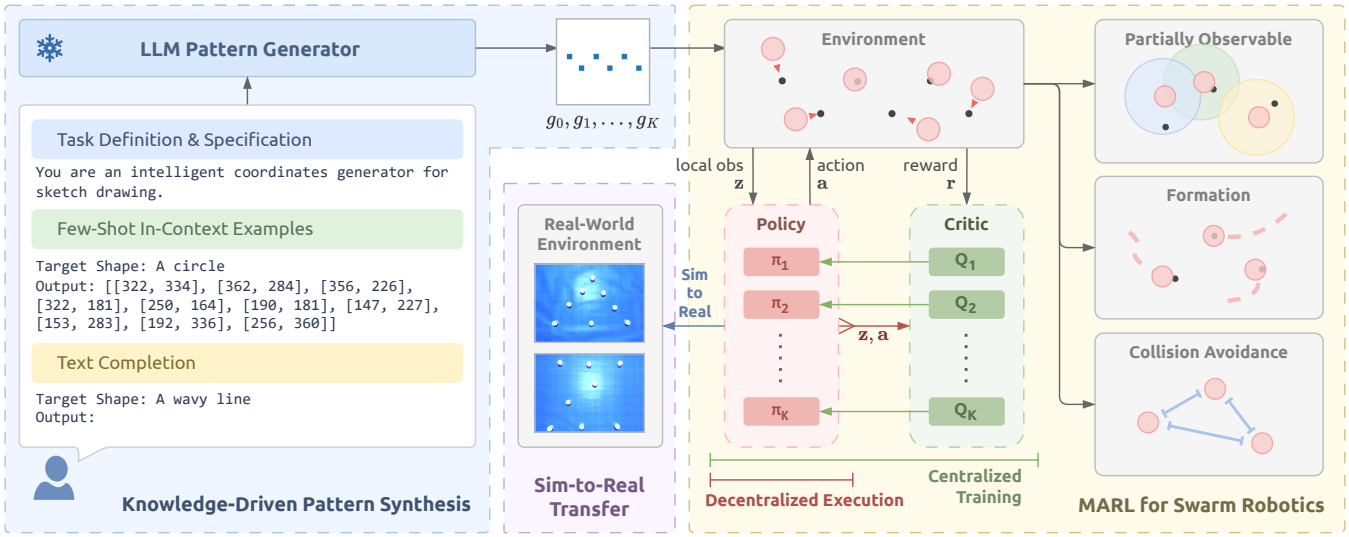

Fig. 2. An overview of our proposed framework, which contains knowledge-driven pattern synthesis and MARL for swarm robotics control.

IV. METHODOLOGY

A. Problem Formulation

This section presents the problem formulation of the proposed methodology. The targeted *Language-Guided Pattern Formation* problem consists of two main phases: *knowledge-driven pattern synthesis*, which generates pattern coordinates of a desired given pattern through high-level language prompts, and *swarm robotics coordination* for controlling swarm robots to form the specified shape. The knowledge-driven pattern synthesis procedure takes as input a text that describes the target pattern in natural language. This input text is tokenized into a sequence of n tokens $W_{shape} \equiv \{w_1, w_2, ..., w_n\}$. Subsequently, the procedure outputs an intermediate goal representation $G \equiv \{g_1, g_2, ..., g_K\}, g_i \in \mathbb{R}^2$, which is arranged in the shape designated by W_{shape}.

Once G is determined, the *swarm robotics coordination* phase commences. This phase considers K homogeneous swarm robots $i \in B \equiv \{1, 2, ..., K\}$, each associated with a uniform radius R and navigating on a 2D Euclidean coordinate platform to form the desired pattern G. The robotic agents operate in a decentralized manner. The position of agent i is defined as $p(i) = (x_i, y_i)$. In addition, this task enforces a collision avoidance constraint to prevent robots from colliding with each other during the formation process [27]. The objective of the problem is formulated as:

$$\min \sum_{i=1}^{K} \|g_i - N(g_i, B)\|, \tag{7}$$

$$\text{subject to } \|p(i) - p(j)\| > 2R, \text{ for } i \neq j, \tag{8}$$

where $\|\cdot\|$ denotes the Euclidean distance defined on $\mathbb{R}^2$, and $N(g_i, B) = p(\arg\min_j \|g_i - p(j)\|)$ is the position of the agent $j \in B$ that is nearest to the point g_i. Eq. (8) describes the constraint essential for collision avoidance [23], [27].

B. Overview of the LGPF Framework

Fig. 2 presents an overview of our framework developed for tackling the *Language-Guided Pattern Formation* (LGPF) problem through three stages: knowledge-driven pattern synthesis, MARL for robotics control, and sim-to-real transfer.

In the first stage, a pre-trained LLM is provided with a task definition prompt, few-shot in-context examples, and the natural language text input representing the desired shape. The LLM then generates intermediate goals in the form of 2D coordinates, which serve as inputs for the subsequent stage. In the second stage, a decentralized MARL policy is adopted to ensure that the robots not only achieve the desired pattern formation but also adhere to collision avoidance constraints. Finally, to deploy our approach in real-world scenarios, we introduce a sim-to-real transfer technique. The objective of adopting sim-to-real transfer is to minimize the domain gap between simulated and real-world environments.

C. Knowledge-Driven Pattern Synthesis

The objective of this phase is to use an LLM to produce the intermediate goal coordinates of the desired shape described by a natural language prompt. To guide LLM in generating the specified shape, this work presents a tailored prompt template that takes advantage of LLM's few-shot generalizability [33], [36]. The framework first receives a user-defined prompt W_{shape}, which describes the desired shape in natural language. This prompt W_{shape} is then embedded into our specially crafted prompt template. Subsequently, the LLM generates the desired pattern G. The prompt template is designed and structured according to the following procedure.

Task Definition and Specification. The template begins by clearly defining the task that the LLM is set to accomplish.

```
You are an intelligent coordinates generator for sketch drawing. I will provide you a target shape. Your task is to use some boxes to form the outlines of the target object, and then generate the coordinates for those boxes.
```

Next, the details and content generation rules are specified.

```
The images are of size 512x512. The boxes are of size 20x20. ... Each coordinate should be in the format of [center x coordinate, center y coordinate].
```

Preventing Unwanted Behaviors. Alongside the task, the prompt template further incorporates guidance on common pitfalls to ensure that the LLM generates appropriate content.

```
Do not introduce extra lines or curves to the target shape.
... Please only reply the coordinates as the following
format without any explanations.
```

Few-shot Prompting. To ensure clear comprehension of the task, the prompt also includes several in-context examples that leverages the few-shot capabilities of LLMs [33], [41].

```
Refer to the example below for guidance.
Target Shape: A circle
Output: [[269, 351], [330, 319], [360, 254], [333, 195],
[259, 166], [194, 177], [154, 228], [157, 287], [198, 338]]
...
```

Text Completion. Finally, a structure for user-defined shape description is provided to the LLM for deriving coordinates.

```
Target Shape: [User-defined shape description]
Output:
```

By using this template, LLMs can generate the specified shapes from high-level descriptions. The resulting coordinates are subsequently passed into the MARL framework as the joint goal for the robots. In the above template, the number of points determined by LLM depends on the complexity of the target shape. To further control the number of points, the desired number can be incorporated into the few-shot examples and text completion section. For example:

```
Target Shape: A circle
Number of boxes: 9
Output: [[269, 351], [330, 319], [360, 254], [333, 195],
[259, 166], [194, 177], [154, 228], [157, 287], [198, 338]]
```

D. MARL for Swarm Robotics Control

Our framework employs CTDE-based MARL to guide swarm robots in forming desired shapes. To ensure flexibility in robot control, fixed goals are not assigned to individual agents. Instead, agents learn a joint policy that considers coordination with other robots during the formation process. The configurations of our MARL algorithm are as follows:

Observation and Action Spaces. The environment follows Dec-POMDP, in which agents perform actions conditioned only on their local observations. The local observation z_i for each agent i contains the velocity (v_i^x, v_i^y) and position (x_i, y_i) of the agent itself, along with the relative positions of the k nearest agents and k nearest goals. The actions an agent can take include moving up, right, down, left, or remaining idle. Each action can take a continuous value within $[0, 1]$.

Reward Function Design. The reward function is designed to achieve two objectives: pattern formation and collision avoidance. The reward for pattern formation, denoted as $r_p = -\sum_{i=0}^{K} \|g_i - N(g_i, B)\|$, is defined as the negative distance between each goal $g_i \in G$ and its nearest agent. This formulation guides agents to cover all the goals and form the specified pattern. To prevent collisions, agents are penalized when the distance between any two agents is less than their diameter. The penalty term r_c is defined as follows:

$$r_c = -\frac{1}{2}\sum_{i=1}^{K}\sum_{j\neq i} \mathbb{1}[\|p(i) - p(j)\| \leq 2R], \quad (9)$$

where $\mathbb{1}[\cdot]$ is the indicator function. Finally, the complete reward function is defined as $r = r_p + \alpha r_c$, where α is used for balancing the importance between the two objectives.

E. Sim-to-Real Transfer

The proposed framework is deployed to a custom real-world robot environment to validate its adaptability. In this environment, we utilize the mobile robots proposed in [42] as the individual units that compose the entire swarm. The robots operate based on position control and navigate towards specified xy-coordinates provided by the host computer.

When transferring the policy trained in a simulation environment to the real world, several sim-to-real gaps need to be addressed. First, the learned policy is designed to provide velocity offsets, which differs from the position-controlled nature of the robots. Secondly, these robots are based on differential-drive control, which is non-holonomic, while the agents in the simulation environment are holonomic. These sim-to-real gaps are mitigated through several alignment strategies. First, the velocity offsets from the policy are converted to positional offsets by being multiplied with a constant factor. The agents then move based on these positional offsets. In addition, to address the behavior discrepancy between holonomic and non-holonomic robots, a constraint is introduced to prevent the robots from moving continuously to reach the exact target position. When a robot receives a new target position, it halts its movement after moving for 0.1 seconds. This measure ensures that the robots do not move excessively to directly reach the target.

V. EXPERIMENTAL RESULTS

A. Experimental Setup

Task Setup. Two types of task setups are employed to demonstrate the capability of our framework. The first one is the formation of fundamental geometric patterns, which is defined and discussed in Section IV. This type of tasks provides an evaluation of swarm robots' collaboration policies and their abilities to precisely form designated patterns. The second setup is *shape chaining*, which represents a more dynamic and complex challenge that assesses the adaptability and coordination of the swarm. In this type of task, robots are required to sequentially form shapes in a specific order. For instance, they may start by forming a circle, then transition into a square, and finally arrange themselves into a triangle.

Simulation Environments. In this section, we describe the simulation environment for evaluating the proposed methodology. Please note that the real-world evaluation platform is described in Section IV-E. The simulated experiments are performed on the Multi-agent Particle Environment (MPE) [8], [17]. MPE provides a number of environments, each of which is designed for a particular multi-agent scenario. In this work, the *simple spread* environment, also known as *cooperative navigation*, is selected to emulate the

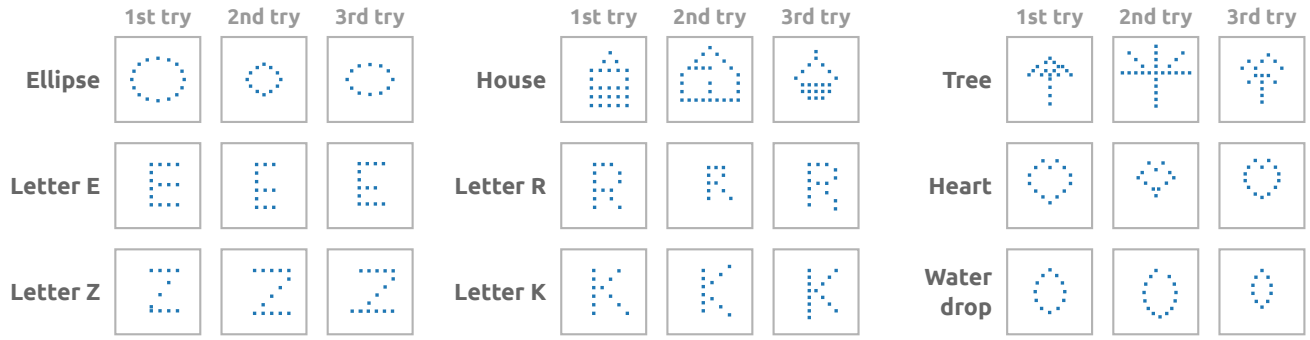

Fig. 3. Visualizations of the patterns generated by the LLM, which demonstrate its ability to form diverse patterns based on high-level instructions.

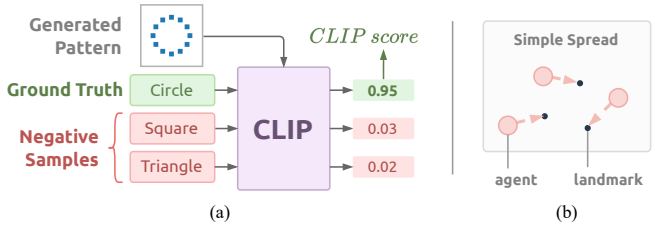

Fig. 4. The *CLIP score* metric and the *simple spread* environment.

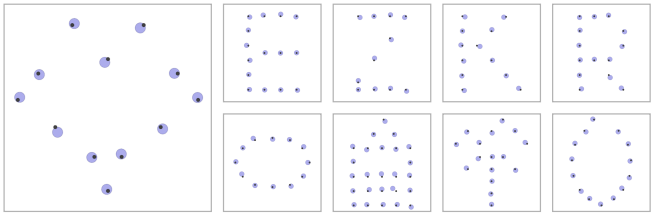

Fig. 5. Qualitative results in MPE simulation environment.

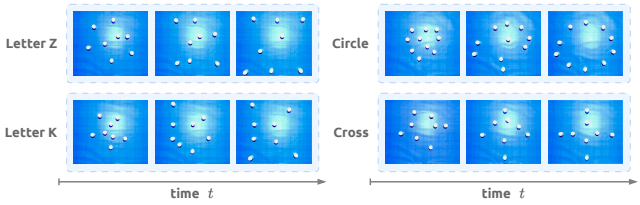

Fig. 6. Qualitative results in real-world environment with mobile-robots.

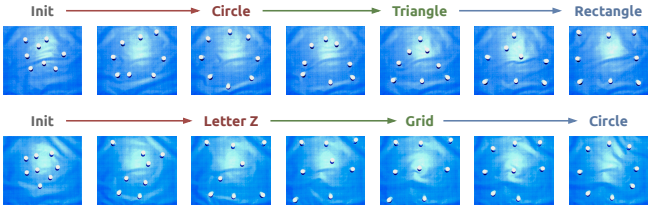

Fig. 7. Visualizations of shape chaining performed by real robots.

pattern formation problem. The *simple spread* environment contains K homogeneous agents moving on a 2D plane without communication, as well as K landmarks representing the target positions of the agents. Fig. 4 (b) presents a depiction of the *simple spread* environment. The observation, action, and reward settings are discussed in Section IV-D. Across all experiments, the number of neighboring agents and landmarks that each agent can observe, denoted as k, is set to five. For efficiently training larger numbers of robots, this work adopts the vectorized version of MPE from [39].

Baselines. The effectiveness of our trained policies based on PIC is evaluated against both MARL and traditional algorithms. MADDPG is selected as the baseline for MARL, while Optimal Reciprocal Collision Avoidance (ORCA) [4] serves as the baseline for the traditional algorithm for navigation and collision avoidance. For both this work and the MADDPG baseline, each agent's policy is parameterized by two fully-connected layers, each with 128 hidden units, and uses the ReLU activation function. The critic in MADDPG has the same deep neural network architecture as the policy network. Our approach adopts the PIC configuration for the critic, which implements a two-layer GCN with 128 hidden units per layer and the ReLU function. Finally, a max-pooling layer is incorporated at the end of the GCN for the PI property [39]. Please note that the environmental settings for the ORCA algorithm, such as agent radius and world size, are matched to those used in the *simple spread* environment.

Performance Metrics. Appropriate evaluation metrics are selected for both knowledge-based pattern synthesis and swarm robotics control for pattern formation. For the former, the *CLIP similarity score* [43] is employed to validate the alignment between the shape described by the prompt and the generated pattern coordinates. Fig. 4 (a) illustrates an overview of the implementation of the CLIP metric. In this example, the generated pattern is visualized as an image by placing squares at each coordinate point in the pattern. This image is then provided to the CLIP model, along with the input prompt and five negative prompts, to calculate the text-image pair similarity score of the ground truth. The inclusion of negative prompts is essential for the CLIP model, as its output is a softmax distribution over the provided text-image pairs. Without the presence of negative samples, the model would lack the necessary context to generate a meaningful similarity score, and can render the comparison ineffective.

For the latter, three metrics are introduced to evaluate the performance of swarm robotics control for pattern formation: *completion rate*, *collision count*, and *inter-agent distance*. *Completion rate* is defined as the ratio of goals (i.e., landmarks) covered by the robotic agents [28]. *Collision count* is computed by summing up the number of collisions at each time t throughout the entire episode [8]. A collision is identified when the distance between any two agents is less than an agent's diameter. Finally, the *inter-agent distance* is determined by first calculating the 25th percentile of the distance from each agent to other agents across the entire

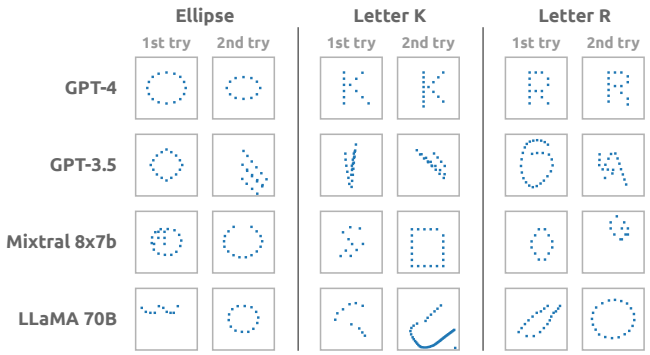

Fig. 8. Qualitative results of the patterns generated by different LLMs.

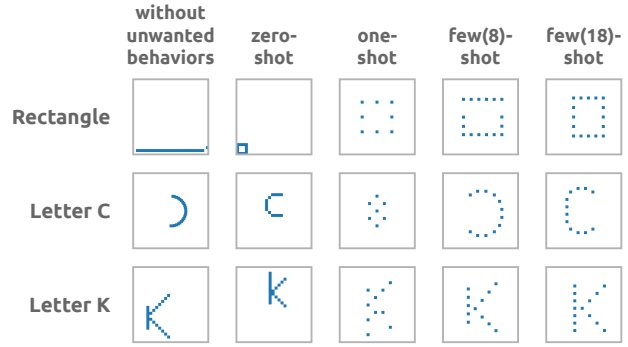

Fig. 9. An illustration of the effect of each element in Prompt Template.

episode and then finding the median value of these lower quartiles. For simplicity and clarity in comparison, the final values are expressed as multiples of a robotic agent's radius.

B. Experimental Results for Goal Generation

To visualize the synthesized patterns, images are created by placing 20x20 squares at each predicted coordinate. Fig. 3 displays the qualitative results of pattern generation using ChatGPT-4. The figure demonstrates that the LLM possesses a significant ability to generate a wide range of patterns according to the input instructions while maintaining diversity across different generations. The image illustrates the extent to which the LLM can comprehend fundamental geometric shapes such as ellipses and drops, as well as understand the concept of various characters from the English alphabet to interpret instructions for generating patterns. Moreover, when tasked with more intricate objects and forms, the LLM exhibits its comprehensive knowledge spanning a vast array of topics and subjects. For instance, it grasps the essential features of a house: a sloped roof and a rectangular body. Beyond these fundamental elements, additional details can be observed, such as an outlined area representing a window in the first house pattern and a simple line suggesting a door in the second. Furthermore, the LLM captures the essence of a tree across all three generated patterns by representing its crown of branches and foliage atop an upright trunk.

C. Experimental Results for Pattern Formation

Table I compares the results of pattern formation of our framework against the baselines. The MADDPG baseline shows a low completion rate, which indicates its inability to form the pattern correctly. In contrast, ORCA achieves

TABLE I

QUANTITATIVE COMPARISON OF PATTERN FORMATION RESULTS.

		Completion Rate	# of Collisions	Inter-Agent Distance
K = 10	ORCA	**1.000**	48.25	$4.30R$
K = 10	MADDPG	0.034	4.60	$5.82R$
K = 10	Ours	**0.947**	**0.90**	$\mathbf{7.15}R$
K = 20	ORCA	**1.000**	166.42	$2.99R$
K = 20	MADDPG	0.048	4.55	$4.29R$
K = 20	Ours	**0.906**	**0.95**	$\mathbf{5.49}R$
K = 30	ORCA	**1.000**	350.18	$2.52R$
K = 30	MADDPG	0.056	**3.18**	$3.72R$
K = 30	Ours	0.844	**5.58**	$\mathbf{4.86}R$

TABLE II

QUANTITATIVE RESULTS FOR LLM-BASED PATTERN GENERATION.

CLIP score (mean±std)	Ellipse	Alphabet E	Alphabet K	Alphabet R	Alphabet Z
ChatGPT-4	**0.922±0.068**	**0.972±0.034**	**0.992±0.007**	**0.902±0.150**	**0.931±0.110**
ChatGPT-3.5	0.513±0.383	0.253±0.094	0.306±0.161	0.257±0.122	0.327±0.190
Mixtral 8x7b	0.724±0.354	0.358±0.092	0.184±0.089	0.162±0.050	0.233±0.118
LLaMA-2 70B	0.484±0.294	0.427±0.193	0.270±0.112	0.235±0.157	0.264±0.144

a high completion rate. Nevertheless, it encounters issues with robot collisions and fails to maintain sufficient distance between them. Our framework not only attains a completion rate exceeding 90% but also demonstrates very few collisions. Moreover, our framework maintains sufficient distance between agents to prevent collisions, which highlights its effectiveness and safety. Fig. 5 illustrates the qualitative results for the pattern formation task in the MPE *simple spread* environment. Across the majority of scenarios, the agents successfully cover their designated goals. Although a small number of agents fail to reach the target positions, they are still able to reach close proximity to the targets.

Fig. 6 illustrates the experimental result of deploying our framework onto a real-world setup. This figure depicts the pattern formation process, where the robots navigate towards their goal positions while actively maintaining appropriate spacing between each other to avert any potential collisions. Furthermore, the robots tackle a more complex challenge referred to as *shape chaining*, as discussed in Section V-A. In this task, the robots must seamlessly transition into a new shape once the current configuration has been successfully achieved. Fig. 7 offers a visual depiction of this *shape chaining* process within the real-world environment. It demonstrates the robots' proficiency in transitioning between various shape formations, and validates that the learned policy enables the swarm robots to deftly handle complex, long-horizon tasks requiring sequences of coordinated actions.

D. Ablation Study for Different LLMs & Prompt Engineering

Different LLMs. This analysis is conducted using different LLMs, including ChatGPT-4, ChatGPT-3.5, Mixtral 8x7b Instruct, and LLaMA-2 70B Chat, all with the same prompt template. The qualitative results are displayed in Fig. 8, while the evaluation results based on the CLIP metric are presented in Table II. These results demonstrate that the breadth and depth of knowledge embedded within LLMs are critical factors that influence the quality of the synthesized patterns. With the state-of-the-art GPT-4 model, the generated patterns are readily recognizable and comprehensible. In contrast,

other LLMs may produce suboptimal or confusing results.

Prompt Engineering. Fig. 9 validates the effectiveness of each element in the prompt template. It is evident that removing any element from the template leads to a degradation in the quality of the generated pattern. For example, without the unwanted behavior section, LLMs can produce faulty patterns that do not adhere to the designated specifications.

VI. CONCLUSIONS

In this work, we investigated leveraging the knowledge encapsulated within an LLM for pattern formation task. Our LGPF framework demonstrated that the few-shot generalizability of LLMs enabled them to synthesize a diverse array of patterns from natural languages. Moreover, by incorporating CTDE-based MARL, LGPF can learn decentralized policies that allow swarm robots to navigate towards designated patterns without collisions. This opens up a new avenue for complex multi-robot coordination tasks and future exploration of LLM integration into swarm robotics research.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support from the National Science and Technology Council (NSTC) in Taiwan under grant numbers MOST 111-2223-E-002-011-MY3, NSTC 113-2221-E-002-212-MY3, and NSTC 113-2640-E-002-003, as well as the support from OMRON SINIC X Corporation in Japan. The authors would also like to express their appreciation for the donation of the GPUs from NVIDIA Corporation and NVIDIA AI Technology Center (NVAITC) used in this work. Furthermore, the authors extend their gratitude to the National Center for High-Performance Computing (NCHC) for providing the necessary computational and storage resources.

REFERENCES

[1] C. C. Cossette *et al.*, "Optimal multi-robot formations for relative pose estimation using range measurements," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 2431–2437, 2022.

[2] J. Alonso-Mora *et al.*, "Multi-robot system for artistic pattern formation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 4512–4517, 2011.

[3] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[4] J. Van Den Berg *et al.*, "Reciprocal n-body collision avoidance," in *Robotics Research: The 14th Int. Symp. ISRR*, pp. 3–19, 2011.

[5] J. Alonso-Mora *et al.*, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," in *Distributed autonomous robotic systems: The 10th Int. symposium*, pp. 203–216, Springer, 2013.

[6] J. Gielis *et al.*, "A critical review of communications in multi-robot systems," *Current Robotics Reports*, vol. 3, no. 4, pp. 213–225, 2022.

[7] F. A. Oliehoek *et al.*, "Optimal and approximate Q-value functions for decentralized pomdps," *Journal of Artificial Intelligence Research*, vol. 32, pp. 289–353, 2008.

[8] R. Lowe *et al.*, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[9] M. Ahn *et al.*, "Do As I Can, Not As I Say: Grounding language in robotic affordances," *arXiv:2204.01691*, 2022.

[10] W. Huang *et al.*, "Grounded Decoding: Guiding text generation with grounded models for robot control," *arXiv:2303.00855*, 2023.

[11] W. Huang *et al.*, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *Int. Conf. on Machine Learning (ICML)*, pp. 9118–9147, 2022.

[12] I. Singh *et al.*, "ProgPrompt: Generating situated robot task plans using large language models," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 11523–11530, 2023.

[13] S. Vemprala *et al.*, "ChatGPT for robotics: Design principles and model abilities," *Microsoft Auton. Syst. Robot. Res*, vol. 2, p. 20, 2023.

[14] A. Brohan *et al.*, "RT-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv:2307.15818*, 2023.

[15] D. Driess *et al.*, "PaLM-E: An embodied multimodal language model," *arXiv:2303.03378*, 2023.

[16] F. A. Oliehoek *et al.*, *A concise introduction to decentralized POMDPs*, vol. 1. Springer, 2016.

[17] I. Mordatch *et al.*, "Emergence of grounded compositional language in multi-agent populations," *arXiv:1703.04908*, 2017.

[18] A. H. Mong-ying and V. Kumar, "Pattern generation with multiple robots," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 2442–2447, 2006.

[19] C. C. Cheah *et al.*, "Region-based shape control for a swarm of robots," *Automatica*, vol. 45, no. 10, pp. 2406–2411, 2009.

[20] M. A. Hsieh *et al.*, "Decentralized controllers for shape generation with robotic swarms," *Robotica*, vol. 26, no. 5, pp. 691–701, 2008.

[21] H. Wang and M. Rubenstein, "Generating goal configurations for scalable shape formation in robotic swarms," in *Distributed Autonomous Robotic Systems: 15th Int. Symposium*, pp. 1–15, Springer, 2022.

[22] M. Rubenstein and W.-M. Shen, "Scalable self-assembly and self-repair in a collective of robots," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 1484–1489, 2009.

[23] H. Wang and M. Rubenstein, "Shape formation in homogeneous swarms using local task swapping," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 597–612, 2020.

[24] M. Alhafnawi *et al.*, "Self-organised saliency detection and representation in robot swarms," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1487–1494, 2021.

[25] G. Sun *et al.*, "Mean-shift exploration in shape assembly of robot swarms," *Nature Communications*, vol. 14, no. 1, p. 3476, 2023.

[26] M. Rubenstein *et al.*, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014.

[27] J. Wang *et al.*, "Pattern-RL: Multi-robot cooperative pattern formation via deep reinforcement learning," in *IEEE Int. Conf. On Machine Learning And Applications (ICMLA)*, pp. 210–215, 2019.

[28] E. A. O. Diallo and T. Sugawara, "Multi-agent pattern formation: a distributed model-free deep reinforcement learning approach," in *Int. Joint Conf. on Neural Networks (IJCNN)*, pp. 1–8, 2020.

[29] P. Rezeck and L. Chaimowicz, "Chemistry-inspired pattern formation with robotic swarms," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9137–9144, 2022.

[30] P. Sadhukhan and R. R. Selmic, "Multi-agent formation control with obstacle avoidance using proximal policy optimization," in *IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, pp. 2694–2699, 2021.

[31] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[32] J. Devlin *et al.*, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, 2018.

[33] T. Brown *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[34] J. Achiam *et al.*, "GPT-4 technical report," *arXiv:2303.08774*, 2023.

[35] A. Q. Jiang *et al.*, "Mixtral of experts," *arXiv:2401.04088*, 2024.

[36] H. Touvron *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv:2307.09288*, 2023.

[37] T. Rashid *et al.*, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[38] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv:1509.02971*, 2015.

[39] I.-J. Liu *et al.*, "PIC: Permutation invariant critic for multi-agent deep reinforcement learning," in *Conf. on Robot Learning (CoRL)*, 2020.

[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv:1609.02907*, 2016.

[41] J. Wei *et al.*, "Finetuned language models are zero-shot learners," *arXiv:2109.01652*, 2021.

[42] S. Ichihashi *et al.*, "Swarm body: Embodied swarm robots," in *Proc. of the 2024 CHI Conf. on Human Factors in Computing Systems*, CHI '24, Association for Computing Machinery, 2024.

[43] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *Int. Conf. on Machine Learning (ICML)*, pp. 8748–8763, 2021.