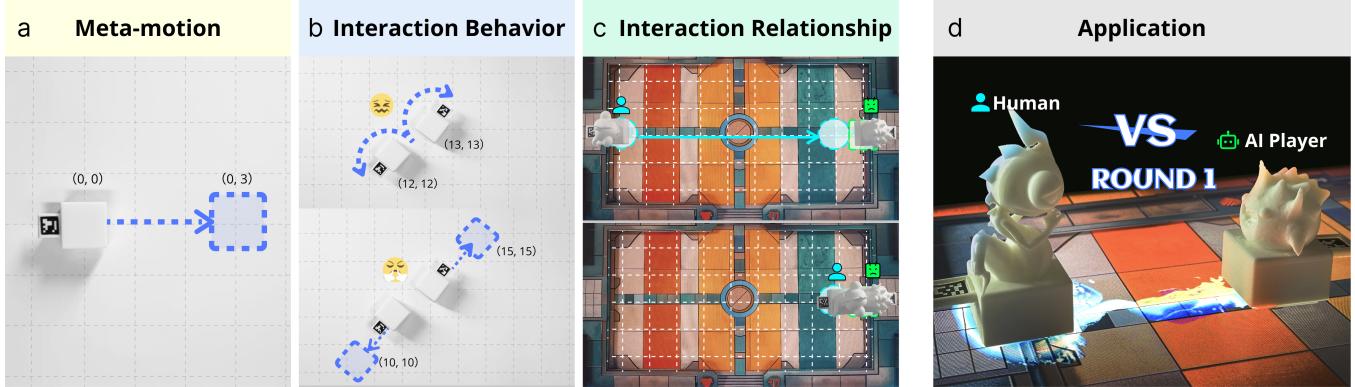


# AI-Gadget Kit: Integrating Swarm User Interfaces with LLM-driven Agents for Rich Tabletop Game Applications

Yijie Guo* Tsinghua University Beijing, China guoyijie.sh@gmail.com	Zhenhan Huang* University of Tsukuba Tsukuba, Japan zhenhan.email.jp@gmail.com	Ruhan Wang* Tsinghua University Beijing, China wangrh22@mails.tsinghua.edu.cn
Zhihao Yao Tsinghua University Beijing, China yaozh_h@outlook.com	Tianyu Yu Tsinghua University Beijing, China yty21@mails.tsinghua.edu.cn	Zhiling Xu Tsinghua University Beijing, China xzl23@mails.tsinghua.edu.cn
Xinyu Zhao Tsinghua University Beijing, China xyzhao23@mails.tsinghua.edu.cn	Xueqing Li Tsinghua University Beijing, China li-xq23@mails.tsinghua.edu.cn	Haipeng Mi Tsinghua University Beijing, China mhp@tsinghua.edu.cn
<b>a Meta-motion</b> 	<b>b Interaction Behavior</b> 	<b>c Interaction Relationship</b> 



**Figure 1: Building a tabletop game with a multi-agent system enabled by AI-Gadget Kit.** a) Extend the meta actions of SUIs to gadgets' complex motion planning, b) interaction behavior generation, c) interaction relationship management. d) Robotic gadget plays a turn-based strategy game with a human player.

## ABSTRACT

While Swarm User Interfaces (SUIs) have succeeded in enriching tangible interaction experiences, their limitations in autonomous action planning have hindered the potential for personalized and dynamic interaction generation in tabletop games. Based on the AI-Gadget Kit we developed, this paper explores how to integrate LLM-driven agents within tabletop games to enable SUIs to execute complex interaction tasks. After defining the design space of this

kit, we elucidate the method for designing agents that can extend the meta-actions of SUIs to complex motion planning. Furthermore, we introduce an add-on prompt method that simplifies the design process for four interaction behaviors and four interaction relationships in tabletop games. Lastly, we present several application scenarios that illustrate the potential of AI-Gadget Kit to construct personalized interaction in SUI tabletop games. We expect to use our work as a case study to inspire research on multi-agent-driven SUI for other scenarios with complex interaction tasks.

\*Equal contribution.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY'

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-XXXX-X/18/06...\$15.00

<https://doi.org/XXXXXXX.XXXXXXX>

## CCS CONCEPTS

- Human-centered computing → Interaction devices.

## KEYWORDS

Personalization; Tangible UIs; LLM-Based Agent; Tabletop Game; Swarm User Interface

**ACM Reference Format:**

Yijie Guo, Zhenhan Huang, Ruhan Wang, Zhihao Yao, Tianyu Yu, Zhiling Xu, Xinyu Zhao, Xueqing Li, and Haipeng Mi. 2018. AI-Gadget Kit: Integrating Swarm User Interfaces with LLM-driven Agents for Rich Tabletop Game Applications. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX) ACM, New York, NY, USA, 14 pages. <https://doi.org/XXXXXXX.XXXXXXX>*

## 1 INTRODUCTION

Swarm User Interface (SUI) as an emerging interface has garnered significant attention from researchers. Researchers have explored multiple application scenarios of SUI, such as data physicalization[28], remote collaboration[9], and educational purposes[12], based on the unique tangible interaction behaviors of SUI, such as collaborative motion, objects actuation, or self-shape changes. However, most existing SUIs utilize a pre-programmed set of action planning rules to execute different interaction tasks during use, i.e., users need to program the action each time they face a new interaction task. This approach struggles to adapt to real-world tasks that are usually more complex, dynamic, and possibly changeable beyond the pre-programmed scope.

In recent years, Large Language Models (LLM) have shown benefits for robotic motion control and planning. Based on the embedded knowledge of LLMs, they are capable of understanding and reasoning about the complex contexts in different tasks, thus dynamically generating responses, e.g., motion planning, based on the contexts in the tasks. Traditional methods for robotic motion planning included using one-decision models, which relied on the prediction of every step of the robot's movement[4, 31]. Recent research has explored using LLMs, particularly LLM-driven agents, to conduct robotic motion planning for more complex interaction tasks. For instance, DiscussNav[17] has used multiple LLM agents with different expertise to make decisions for robotic navigation in a complex interior scenario. Nevertheless, most of these works focused on single-robot systems. Research on multiple-robot systems, e.g., SUI, and how to use LLMs to assist the motion planning of these systems for complex interaction tasks still remains unexplored.

Among the various applications of SUI, the tabletop game is a typical scenario that contains versatile complex interaction tasks. Existing research has explored the application of tangible and swarm user interfaces in tabletop games to enhance interactivity and enjoyment, such as using robots to facilitate embodied AI players[18, 29], robotic game masters [5], and automated gadgets[1, 10]. However, the action planning for robots in these studies still relies on pre-programmed rules, which makes it challenging to execute the complex interaction tasks in tabletop games, such as understanding and reacting to complex game narratives, improvised decisions of players, or emotional expressions from players.

In this paper, we aimed to use the tabletop game as a case study to explore the application of LLMs on action planning of SUI in scenarios with complex interaction tasks. We proposed AI-gadget Kit, a multi-agent SUI tabletop gaming system, which is designed to facilitate dynamic and complex interaction tasks in tabletop games. We first introduced the system architecture of the AI-gadget Kit, which includes a set of swarm robots based on existing platforms to perform the gadget behaviors, and a multi-agent system responsible for executing the game and generating action plans for the swarm

robots. We then elaborated the design of the multi-agent system, comprising a series of meta-motions for individual robots, two LLM-based agents for complex action planning, and a set of add-on prompts aimed at reinforcing the understanding and reacting capabilities of the agents. At last, we demonstrate four application examples using AI-gadget Kit to showcase the effect of the multi-agent-driven SUI on executing complex interaction tasks in tabletop games. Through this work, we aim to inspire the research of multi-agent-driven SUI on other scenarios with complex interaction tasks.

In summary, the contribution of this paper includes:

- (1) AI-gadget Kit, a multi-agent SUI tabletop gaming system, which consists of a series of meta-motions for individual robots, two LLM-based agents for complex action planning, and a series of add-on prompts tailored to the tabletop gaming scenario to enhance the understanding capability of the multi-agent system.
- (2) A set of application examples using AI-gadget Kit on tabletop games, which demonstrates the effect of agent-driven swarm robots as gadgets in tabletop games for complex interaction tasks.

## 2 RELATED WORK

### 2.1 Swarm User Interface

Compared to traditional Tangible User Interfaces (TUI), Swarm User Interfaces (SUI) introduce multiple moving robots that enable collaborative motion, providing a flexible and extensive physical interaction space and multi-modal interaction experiences.

Researchers in Human-Computer Interaction (HCI) have explored various applications of SUI. For instance, SwarmHaptic[13] utilized small wheeled swarm robots moving on a flat surface to construct a novel tactile interface. Rovables[2] provided a series of robots that were capable of autonomous movement on wearable clothing, proposing an interaction space for sensing and actuation on wearables. ShapeBots[28] enabled a group of self-deforming robots to individually or collectively change their configuration to display information in physical space. Additionally, Holobots[9] proposed a mixed-reality remote collaboration system augmenting holographic telepresence with synchronized mobile robots. Out of academia, SUIs have also demonstrated extensive application prospects in education and entertainment scenarios. Sony employs programmable small robots called Toio<sup>1</sup> to engage learners from elementary to adult in logical thinking and learning programming Thymo<sup>2</sup> robots provide STEM education for learners of all ages.

However, most existing SUIs utilize a pre-programmed set of action planning rules to execute different interaction tasks during use. For example, although Holobots creatively proposed six interaction types for remote collaboration, constrained by predetermined programming, they struggled to dynamically generate personalized tactile feedback based on users' flexible needs during actual usage. Thus, a system that is capable of understanding and reacting to complex interaction tasks will significantly improve the generalizability of interaction behaviors of SUIs in these works.

<sup>1</sup><https://toio.io/>

<sup>2</sup><https://www.thymio.org/>

## 2.2 Agents for Action Planning

In the domain of robotics, researchers aim to issue high-level instructions to robotic agents. These agents automatically translate the instructions into low-level actions for execution by robots, eliminating the need for humans to manually program. The Skill Transformer[8], leveraging a neural network model based on the Transformer architecture[30], predicts low-level actions for robots, enabling them to accomplish embodied tasks of moving objects to specified targets and locations in complex environments. With the advent of Large Language Models (LLMs), researchers have sought to harness LLMs' robust natural language understanding capabilities to process generalized, natural language-based embodied instructions. For example, March in Chat[21] interacts with agents, LLMs, and VLMs to navigate daily activity scenes based on vague natural language instructions. VoxPoser[7] estimates the potential benefits and losses of objects in a scene towards fulfilling an instruction using LLMs, generating a 3D value map of the scene to derive the robot's trajectory.

Researchers have also recognized that collaborative decision-making among multiple agents for a robot's actions can enable adaptation to more complex scenarios compared to decisions made by a single agent alone. DiscussNav[17] is used to address navigation problems in complex scenes, where robots take each step with the involvement of multiple LLM/VLM agents with different specialties, enhancing the robots' generalization ability in navigation.

Despite existing research has shown feasible performance in action planning and robotic control for embodied agents, their validation of task planning for embodied tasks has been primarily confined to operational tasks in daily routine scenes [7, 8] and navigation tasks[17] with less emphasis on other certain genres of task scenarios, e.g., fictional narrative settings. Moreover, its interaction planning capabilities in user interfaces driven by multiple robots also require validation. Exploring the integration of LLM-based agents with SUI will broaden the application scenarios and interaction cases for related work in robotics.

## 2.3 Robots in Tabletop Game

In recent years, researchers have explored the application of tangible and swarm user interfaces in tabletop games to enhance interactivity and enjoyment.

With the emerging trend of robots being small and versatile, more and more robots are used as embodied AI players or gadgets in tabletop games. For example, Brock et al.[1] utilized the robot Haru to simulate the behavior of remote human players. Researchers also used robots to serve as robotic gadgets in the game, such as creating chess that can move automatically to enable novel and compelling interaction experiences[10]. Sparkybot[6] allowed children to use mobile robots as different actors in storytelling games to enhance children's creativity.

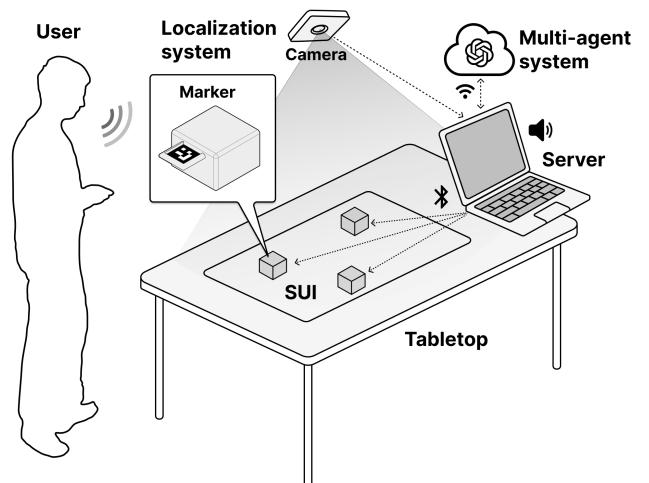
These works, that use SUI for tabletop games, provide rich user interaction spaces. However, the action planning for robots in these studies still relies on pre-programmed rules, which makes it challenging to execute the complex interaction tasks in tabletop games, such as understanding and reacting to complex game narratives, improvised decisions of players, or emotional expressions from

players. In this work, we aimed to leverage LLM-based agents to assist action planning for SUI, in order to execute complex interaction behaviors of the gadgets in tabletop games.

## 3 SYSTEM ARCHITECTURE OF AI-GAGET KIT

The system architecture of AI-Gadget Kit, shown in Figure2, consists of an SUI platform, a localization system, a server, and a LLM-based multi-agent system. The SUI platform includes multiple independent robots, which are used to actuate various gadget behaviors in tabletop games. The localization system comprises a camera and multiple on-robot markers, used to obtain the position and orientation of each robot. The server obtains users' text or verbal input commands, as well as robots' position and orientation data, and then sends them to the LLM-based multi-agent system for information processing. The server also receives the actuations generated by the multi-agent system, playing sound effects, or sending action sequences to the robots in the SUI platform. The LLM-based multi-agent system is responsible for the execution of the core game interactions. Based on the game's rules and knowledge, the multi-agent system responds to user commands, reasons the gadget behaviors in the game, and then generates the action sequences to control the SUI robots.

In this project, we utilized a 1m\*1m tabletop as the interaction space. The space was divided into a 30\*30 coordinate system, with the east and north directions serving as the positive directions of x and y axes, respectively. We used Sony's Toio robots as the SUI platform. Each individual robot has dimensions of 3.2\*3.2\*2.5cm. We used a PC as the server, which communicates with Toio robots via Bluetooth and communicates with the LLM via WiFi. In our localization system, we employed an imx415 network IP camera, positioned 1m above the tabletop, and we used ArUco codes as the localization markers on the robots. The server retrieves the video stream from the camera using the RTSP protocol and uses the Python OpenCV to track the position and orientation of each robot. We developed the multi-agent system based on GPT4 LLM. The design of the multi-agent system is introduced as follows.



**Figure 2: System Architecture of AI-Gadget Kit.**

## 4 MULTI-AGENT SYSTEM

In the AI-Gadget Kit, we utilized a multi-agent system to compute the core interaction of the game. Based on the rules, and knowledge of the game, the system responds to the user's command, reasons the gadget behaviors within the game, and then generates the action sequences to control the SUI robots.

To build the multi-agent system, we first defined a set of meta-actions for each single robot, and then designed a two-agent system, including a *Coordinator* agent and a *Controller* agent, to learn and use those meta-actions for complex motion planning. We also designed a set of add-on prompts, including prompts for *Interaction behavior planning* and *Interaction relationship planning*, to enhance the agents to understand and react to complex interactions during the game.

### 4.1 Meta-action for Individual Robot

To leverage the LLM-based agents for planning and generating complex actions for our SUI, we first defined the primitive movement patterns of an individual robot, i.e., meta-actions (Figure 3a). An individual Toio robot moves by controlling the motors of the two wheels. Thus, in this work, we controlled the robot's meta-action by controlling the rotation direction (clockwise by default or anti-clockwise), speed (three levels - 10, 20, or 30<sup>3</sup>), and the duration (x seconds) of each motor.

We categorized the robot's meta-actions into two types of movements: *translation* and *rotation*, defined as follows:

- **Rotation:**
  - **Rotation A:** The robot rotates around its center, by spinning its wheels in opposite directions at the same speed for a specified seconds, achieving a precise angle of rotation.
  - **Rotation B:** The robot rotates around one side of itself, by spinning only one wheel for a specified seconds, achieving a specific angle of rotation.
- **Translation:** The robot adjusts its orientation to the desired direction through Rotation A, followed by spinning both wheels in the same direction and speed for a few seconds to achieve linear translation over a specific distance.

During each movement, the server determines the duration based on a simple calculation of expected translation or rotation displacement and official kinematic data<sup>4</sup>.

Furthermore, considering interactions between multiple robots, we designed another type of meta-action to adjust their relative orientations, including face-to-face, back-to-back, face-to-back, parallel, and counter-parallel, defined and illustrated in Figure 3a. The two robots conducting one of these meta-actions adjust their orientations by executing Rotation A for a time.

By repeatedly calling and combining these three types of meta-actions with custom parameters, the agents in the AI-Gadget kit could generate multiple actions in sequences to facilitate the complex motion planning of these robots in SUI.

### 4.2 Complex Motion Planning

To facilitate the complex motion planning for SUI to execute the gadget behaviors in tabletop games, we developed an LLM-based two-agent system, that aims to understand and react to the game contexts and then generate the action sequences for the robots (Figure 3d). Specifically, we proposed two agents in the system with expert prompts: *Coordinator* and *Controller*:

**(1) Coordinator Agent.** The operation in a tabletop game typically involves the user's commands and the processing of context information. Taking chess as an example, if the user inputs a command of "move the queen to A1", the execution of this command involves the actual movement of the queen, and processing of context information such as the dimensions of the chessboard, the queen's movement rules, and whether an opponent's piece can be captured. Hence, we designed a Coordinator agent to respond by processing the players' commands and the game context information, then reasoning the commands for each gadget within this interaction step. In the prompts for Coordinator, we asked the agent to act as administrator, coordinator, and referee in the game. We added the description of a series of duties to the prompts, such as explaining rules, coordinating actions, and updating game states. We also inputted the game's environmental settings to the prompts, including the size of the map and the coordinate system.

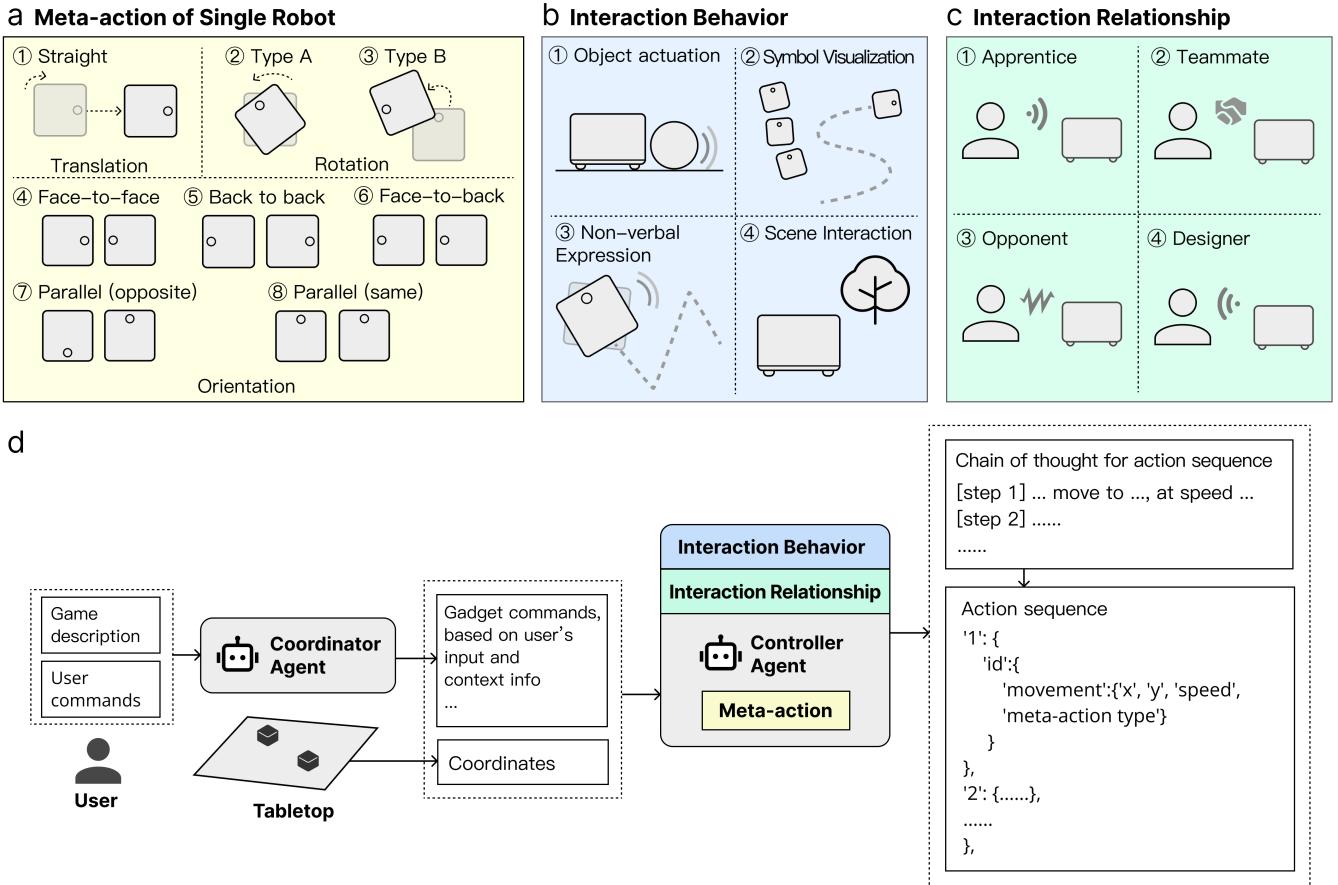
Note that, to allow the Coordinator agent to focus on game operations, we employed a "reality-agnostic" approach. Specifically, the Coordinator only facilitates the execution of the game, without dealing with the physical parameters of the SUI robots, such as their locations or next movements. We aim to use this method to enhance the Coordinator's understanding and reaction capabilities through efficient prompting of LLM. The prompts of the Coordinator are detailed in Supplementary Material.

**(2) Controller Agent.** To use SUI in tabletop games, we require the robots to plan their motion according to the gadget behaviors in the game. To this end, we proposed a Controller agent in our system, which is responsible for embodying characters and generating the action sequence of each robot that represents these characters. The Controller agent is designed to gather information from two sources: the gadget commands outputted by the Coordinator agent, along with the physical location data of the robots at the given time. Next, we asked the Controller to generate the action sequences for the robots using the meta-actions, while simultaneously considering the logical flow of the game and the gameplay experience.

To ensure these action sequences are properly formatted for the robot actuation, we designed a Chain-of-Thought (CoT) prompting [32] for the Controller. The CoT prompting of the Controller unfolds as follows: (1) Output a textual description of the action sequence of the robots that will move, along with the current location information of these robots; (2) Generate an action sequence based on the aforementioned textual description in the form of a Python dictionary. This Python dictionary encompasses ID for the robots, as well as details for each subsequent meta-action, including destination locations/angles, speeds, types of movement, etc. Furthermore, we utilized in-context learning and few-shot learning approaches[16], which provide specific examples in the prompts to demonstrate the process of the aforementioned CoT prompting, to

<sup>3</sup>Speed values in Toio platform

<sup>4</sup>[https://toio.github.io/toio-spec/en/docs/hardware\\_components](https://toio.github.io/toio-spec/en/docs/hardware_components)



**Figure 3: Design space for SUI action planning in tabletop game scenarios through our kit. The kit includes a two-agent system(d) that can generate action sequences for gadgets to complete interaction tasks based on eight meta-actions (a), four types of interaction behaviors (b), and four types of interaction relationships (c).**

assist the agent in effectively learning how to generate and plan the motion sequences.

During use, users first input the game's description and rules to the system, in order to declare the game to play. The system then understands and initiates the game based on the inherent knowledge of LLMs. Then, users continuously input the game commands to the system to engage with the game. The two agents in the system analyze these commands and the ongoing contextual information of the game, reasoning the gadget behaviors in the game, and then generating the action sequences to actuate the motion of SUI robots, embodying the interactions of the gadgets with users.

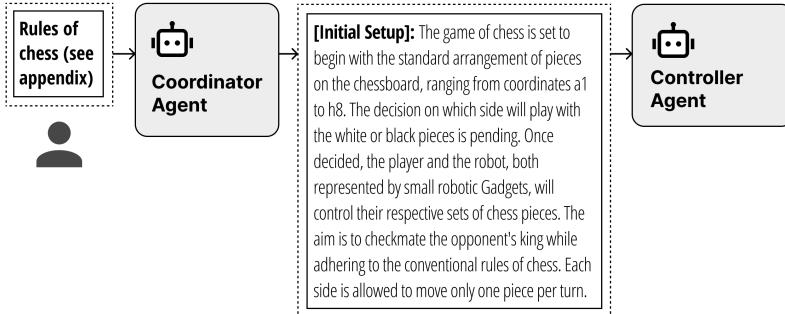
Here we presented a specific case of using the two-agent system to play a chess game. The comprehensive process of the interaction was illustrated in Figure 4, including the initiation step of the game and a typical step during one round of the game. In the initiation step, the user first inputted the game name and a piece of introduction to the system (see Supplementary Material). The Coordinator then initialized the game based on the user's input and outputted the response as shown in Figure 4a. Note that in this step, since the user did not input any specific game commands,

the Controller agent did not generate specific action sequences, either. Subsequently, as the game started, the user continuously gave commands to the system. For instance, in the first round, the user inputted a command: "Move the pawn from d2 to d4". Next, the Coordinator responded to the command and informed the Controller of the gadget behavior of the pawn actuated by the user, as shown in the middle block in 4b. The Controlled then received the gadget behavior of the pawn, along with the actual location data (shown in 4c) of the robot that represented the pawn gadget, proceeding to generate the corresponding action sequence for the pawn robot through the CoT process. The output action sequence of the pawn robot is shown in the right block in 4b. The server in the system then decoded the action sequence from the Controller, sent the motion commands to the pawn gadget, and then actuated the movement of the pawn gadget, shown in 4d.

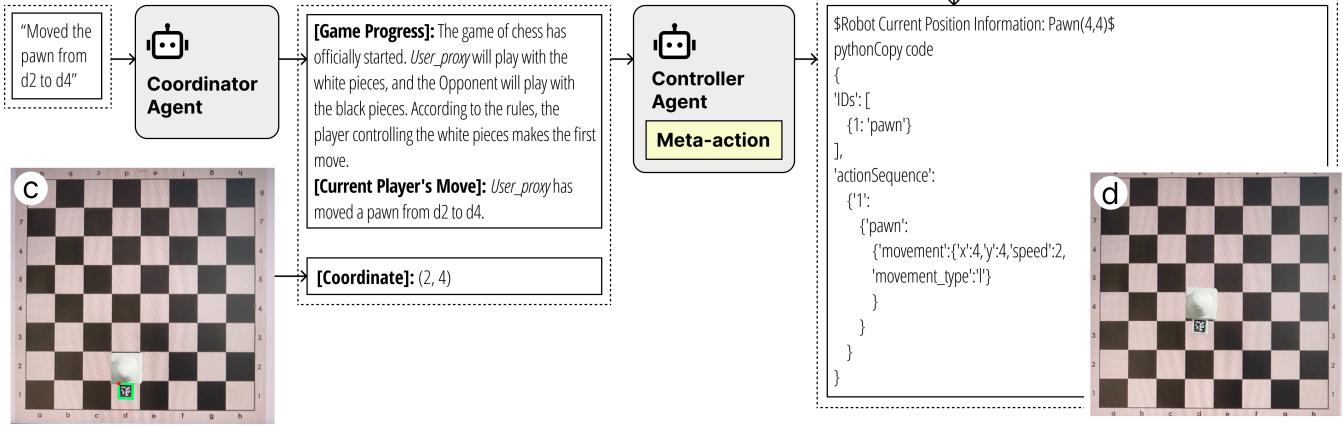
### 4.3 Interactive Behavior Planning

Generally, in the context of utilizing SUI for tabletop games, depending on the specific game scenario, SUI often needs to control the movements of robots and translate various game operations

### a At the beginning of the game



### b At the (first) round of the game



**Figure 4: An example using our kit: (a) Users can declare the game they are playing through an introduction and rules entry. (b) Then, they input game commands based on the scenario to interact with the gadget. The gadget generates corresponding action sequences based on a two-agent system within the kit.**

into tangible interactive behaviors[19]. However, in practice, we have found that due to the complexity and specificity of various interactive behaviors, the agents in our system, particularly the Controller, may not be able to rely solely on the generic prompts to realize all of these tangible interactive behaviors. To address this challenge, we have enhanced the Controller’s capability by incorporating several sets of additional prompts (add-on prompts) following a one/few-shot learning scheme. This approach aids the Controller in comprehending certain specific robotic operations (abilities) applied to different contexts. We anticipate that these add-on prompts will optimize the system’s ability to generate action sequences for Gadgets across various game scenarios.

To determine which interactive behaviors required the design of additional prompts, we referenced past work in SUI [6, 9, 15, 19, 20, 28, 34] and recruited five designers with a background in Human-Computer Interaction (HCI) and experience in tabletop games for an informal interview and brainstorming session. During this process, we identified four common types of interaction behavior related to Gadget operations in tabletop games, including Objective Actuation, Symbol Visualization, Non-verbal Expression, and Scene Interaction. After that, we designed additional prompts

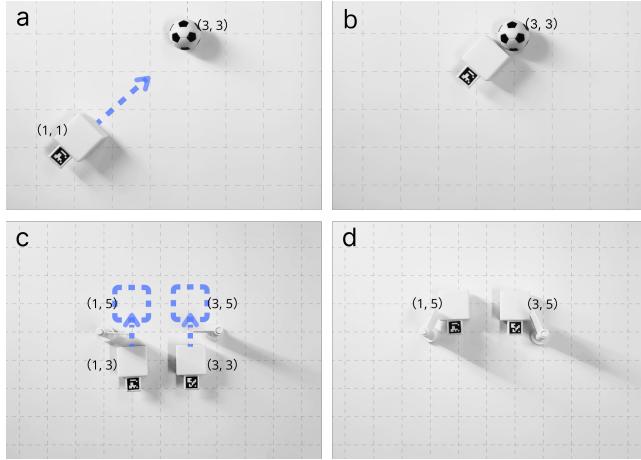
for the Controller based on each type of interaction behavior, as detailed below.

**4.3.1 Object Actuation.** In tabletop games based on SUI, the ability of the system to automatically manipulate objects on the table using robotic gadgets [9] enables the automation of game prop operations or simulates interactions between players and AI opponents or remote players. To this end, we added a set of additional prompts for the controller to assist in generating action sequences for our Gadgets to "actuate objects". In these prompts, we specify that the Controller should focus on the speed and trajectory of robot movement to achieve object movement along a prescribed path and simulate the properties of objects, such as weight. We also included a specific example in the prompts, which involves the task of pushing a heavy box to a designated location and the expected action sequence generated by the Controller.

After incorporating the add-on prompt for object actuation into the Controller’s original prompt, we test the after-modified effectiveness by using the Controller to actuate the Gadgets in scenarios involving pushing heavy objects. We tested the effectiveness of using the controller to actuate the Gadgets in pushing light objects. For instance, it was stipulated that the Gadgets start at a position

(1, 1) and need to kick a light plastic soccer ball located at (3, 3). The outcome of the Gadget's movement and the action sequences generated by the controller are depicted in Figure 5a-b.

Also, in another example, two gadgets are initially located at (1, 1) and (3, 1), and we require the gadgets to push the two very heavy doors to (1, 4) (3, 4) from (1, 3) and (3, 3). The demonstration of the movements of the Gadgets and the action sequence generated by the Controller is shown in Figure 5c-d.

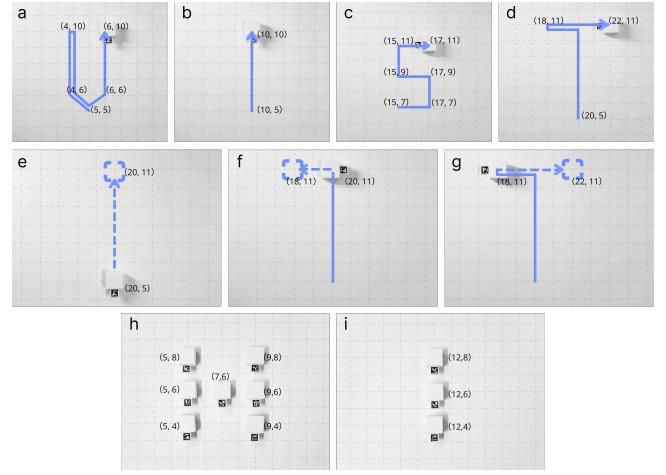


**Figure 5:** The gadget starts at position (1,1) and needs to kick a light plastic soccer ball located at (3,3). a) Gadget: Moves northeast at speed 3 to approach the soccer ball's location b) Kicks the soccer ball, simulating the action by moving east to (3,3). c-d) Two gadgets open a door together by moving towards it.

**4.3.2 Symbol Visualization.** In SUI-based tabletop games, the system's ability to use robotic gadgets to visualize certain graphic symbols can enhance the presentation of textual or graphical information in the game [74]. For this purpose, we added a set of add-on prompts for the Controller to assist in generating action sequences for swarm robots (the Gadgets) for Symbol Visualization. Specifically, we designed two methods for visualizing graphic symbols: "trajectory tracing" and "robotic formation," along with corresponding prompts for each method. To make this visualization function more consistent, we also added certain rules, such as "not inverting the symbols vertically" and "using uppercase letters for English alphabets". We also provided an example for each visualization method in the add-on prompt, including a description of a natural language task for visualizing the letters "HCI," along with the expected action sequence the Controller should generate. Detailed prompts are provided in the Supplementary Material.

After incorporating the add-on prompt for symbol visualization into the Controller, we tested its effectiveness by using the Controller to drive certain Gadgets in visualizing "UIST." For the "tracing" method, the Controller successfully determined to utilize four Gadgets and generated appropriate trajectories of movement for each Gadget. The trajectory patterns and the action sequences that formed these trajectories are illustrated in Figure 6a-g.

For the "formation" method, the Controller determined to utilize multiple robots to separately form the shapes of the letters "H" and "I." The effectiveness of the robotic forming these letters and the action sequences used are illustrated in Figure 6h-i.



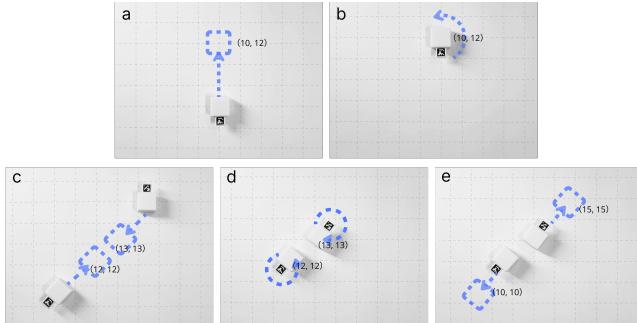
**Figure 6:** The gadgets were asked to present the word 'UIST' or 'HI'. a-d) Agents generated trajectories of four gadgets, which were arranged to form 'U', 'I', 'S'. e-f) A detailed demonstration of how one of the gadgets presents the letter "T" through its movement trajectory. h-i) Agents make each letter composed of multiple gadgets, using 7 and 3 gadgets to form the letters 'H' and 'I' respectively.

**4.3.3 Non-Verbal Expression.** In SUI-based tabletop games, the ability of the system to use robotic gadgets for non-verbal expressions, such as displaying characters' emotions, can significantly enhance the narrative expressiveness of tabletop games [20]. To this end, we incorporate a set of add-on prompts for the Controller to assist in generating action sequences for swarm robots for "non-verbal expression." Specifically, we designed two methods of non-verbal expression: "mood expression" and "social expression," along with corresponding prompts for each method. We also provided an example for each method in the prompts, along with the expected action sequences the Controller should generate. The examples include asking the Gadget to express sadness and a greeting between two Gadgets. Related prompts are detailed in the Supplementary Material.

After incorporating the add-on prompt for non-verbal expression into the Controller, we first tested the effectiveness of using the controller to drive a single Gadget to express excitement. The demonstration of this expression and the corresponding action sequences are illustrated in the Figure 7a-b.

Next, we tested the effectiveness of using the controller to drive two Gadgets to express a disputing social behavior. The demonstration of this expression and the corresponding action sequences are illustrated in the Figure 7c-e.

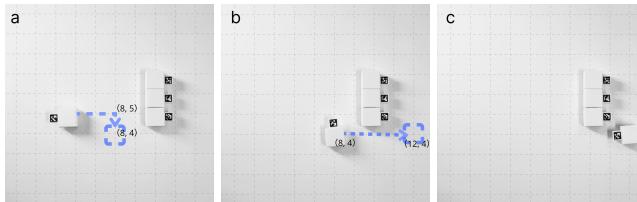
**4.3.4 Scene Interaction.** In SUI-based tabletop games, as it is expected that a robotic gadget should be able to perform actions such as storytelling within a scene, the gadget's behavior may need to



**Figure 7: a-b)** A single gadget expresses excitement, the gadgets move upward a little and rotate for a circle. **c-e)** Two gadgets express an argument, they move toward each other, each rotates for a circle during the argument, and each retreats when the argument is over.

interact with the settings of map environment [6]. To this end, we incorporate a set of add-on prompts for the Controller to assist in generating action sequences for swarm robots for "scene interaction." Specifically, we proposed that the Controller possesses a "global perspective" in the Gadget's action planning, taking into account additional map information, scene props, and even the positions or statuses of other Gadgets among other factors. We also provided an example for each method in the prompts, along with the expected action sequences the Controller should generate. The add-on prompt includes an example demonstrating a Gadget navigating around an obstacle of a certain length to reach the other side of the obstacle. Related prompts are specified in the Supplementary Material.

After incorporating the add-on prompt for scene interaction into the Controller, we tested it with an example of navigating a Gadget around an obstacle formed by three other Gadgets. The demonstration of this expression and the corresponding action sequences are illustrated in Figure 8.



**Figure 8:** The gadget goes around a wall consisting of three other gadgets, approaching the obstacle before moving to the south side of the obstacle and moving east to get over the obstacle.

#### 4.4 Interaction Relationship Planning

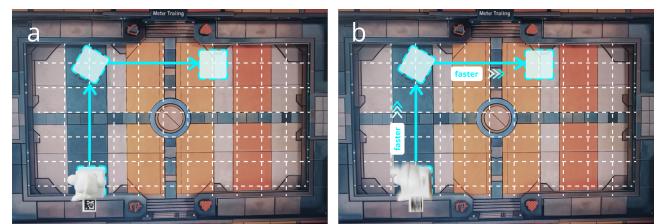
Within the multiple interaction rounds in a game, it is crucial for agents to reflect upon the relational structure underpinning their engagements with players. This encompasses scenarios where agents may either confront or collaborate with players, or independently

modulate their responses in alignment with the unfolding narrative, thereby contributing to the thematic ambiance. Drawing from this premise, researchers' investigations have distilled the potential stances an agent might assume into four distinct categories: Apprentice, Competitor, Teammate, or Designer [35]. To enable agents to grasp the relationship between human-computer interaction relationships and the generation of interactive behaviors within the game, we have augmented the Controller with the additional prompt (add-on prompt) to understand the varying interaction relationships.

**4.4.1 Apprentice.** Within the context of most tabletop gaming environments, Gadgets in our system typically embody the player's avatar or operate as player-controlled entities<sup>5</sup>. We believe that the ability of the system to act as an "apprentice," autonomously and precisely modifying the actions of robotic gadgets in response to users' suggestions, would significantly enhance personalized interactive behaviors. To achieve this, we established a set of additional prompts for the Controller, enabling it to refer to and adjust its action planning as much as possible according to the user's guidance. The specific prompts are detailed in the Supplementary Material.

After incorporating the add-on prompt for "apprentice" into the Controller, we tested it with an example of requesting to "speed up Gadgets". The following is an example of an action sequence generated by the Controller controlling a Gadget to move from (5,5) to (10,10) on the grid map. The translation (movement) speed is set to 2.

Following this, if we then request the Gadget to move faster, the Controller updates and generates the following sequence of actions as Figure 9



**Figure 9:** The gadget was asked to move from (5, 5) to (10, 10), and after being asked to speed up, the gadget moved faster.

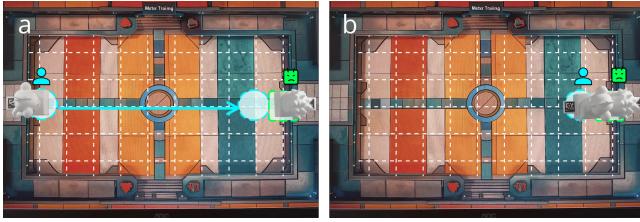
**4.4.2 Opponent.** Many tabletop games feature the roles of opponents not controlled by players, engaging in competitive activities with the players. We believe it can enrich the gaming experience of tabletop games if our system comprehends the concept of "opponent", establishes non-player-controlled opponent roles, and uses the Gadgets to materialize their interactions. To facilitate this, we have established a set of additional prompts for the Controller, helping it generate and enact the roles of single or multiple opponents for competition or matches in games.

We specify in the prompt the planning about the principles and duties that the Controller should adhere to when generating "Opponent" roles [22]. For example, we set the goal of the opponent "... to

<sup>5</sup><http://scruffygroggnard.com/>

challenge the opponent characters through strategy and decision-making while keeping the game fair and enjoyable." Furthermore, we have defined a mechanism for the Controller how to analyze player behavior and dynamically formulate challenging interaction strategies. For example, we specify in the prompt that the Controller "... formulate challenging strategies based on the current state of the game and the behaviors of opponent characters." The prompts are specified in the Supplementary Material.

After incorporating the add-on prompt for "opponent" into the Controller, we tested it with an example of using Gadgets for combat behavior. In this example, one gadget embodies the role of a Monster1 commanded by the user (with action sequences generated by the Controller), while another gadget was controlled by the system, acting as an opponent Monster2. After the Monster1 is commanded to use Thunderbolt to attack, the Controller generates the action sequence of their battle as Figure 10.

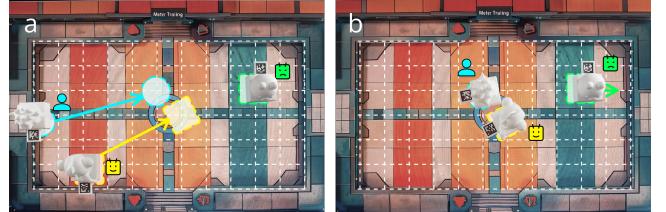


**Figure 10: Monster1 is asked to attack Monster2, and Monster2 responds being attacked.**

**4.4.3 Teammate.** Similar to opponents, in tabletop gaming, there exist numerous non-player controlled characters that support the player by collaborating to accomplish tasks or combat adversaries. Enriching the gaming experience becomes more feasible if our system can comprehend the "teammate (ally)" relationship, autonomously establish these non-player controlled teammate roles, and materialize interactions using the Gadgets. To achieve this, we have also developed a set of add-on prompts for the Controller, fostering it in generating and embodying one or multiple teammate roles. We specify in the prompt the planning about the principles and duties that the Controller should adhere to when generating "Teammate" roles [22]. For example, we set a goal for the teammate "... is to support teammate characters through effective collaboration and strategy, ensuring the success of the entire team and the enjoyment of the game..." Furthermore, we have established guidelines in the prompts for how Controllers can support teammate roles through effective collaboration and strategic planning, such as "providing necessary support to help teammate characters overcome challenges while ensuring not to overshadow them, maintaining the game's balance and interest.", to ensure the success of the entire team and enhance the enjoyment of the game. The prompts are specified in the Supplementary Material.

After incorporating the add-on prompt for "teammate" into the Controller, we tested it with an example of using Gadgets for collaborative combat behavior as Figure 11. In this scenario, one Gadget embodies the role of a Monster1 commanded by the user (with action sequences generated by the Controller), while two other Gadgets, generated and governed by the Controller, take on the

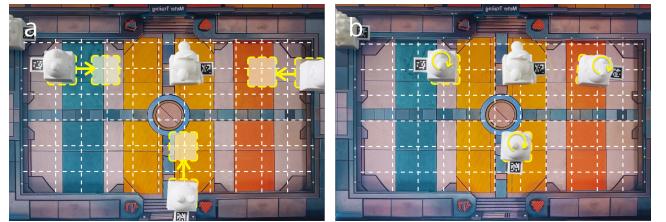
roles of a teammate Monster2 and an opponent Monster3, respectively.



**Figure 11: Gadgets that can generate collaborative behavior act as a teammate of the user and attack the enemy together.**

**4.4.4 Designer.** In tabletop games, more roles that serve to enhance the narrative, such as NPCs (Non-Player Characters) or props, may also be present. For instance, upon the players and their teammates defeating adversaries, villager NPCs can be configured to celebrate players' victory. If the Controller can act as a "designer," generating these roles based on the game's narrative and materializing their actions through gadgets, then it would elevate the dramatic effect and overall experience of the game. To facilitate this, we have developed a set of add-on prompts for the Controller, enabling it to "... spontaneously generate new characters, NPCs, items, or plots, as well as the corresponding robotic action sequences, to advance the game storyline." Related prompts are detailed in the Supplementary Material.

After incorporating the add-on prompt for "designer" into the Controller, we tested it with an example focusing on a pivotal narrative moment following a battle victory. Specifically, when the user-controlled Monster1 and the teammate Monster2 defeated Monster3, the Controller generated a storyline in which villager NPCs appeared to celebrate the victory. To bring this celebratory behavior to life, the Controller then invoked three new Gadgets designed for this purpose as Figure 12.



**Figure 12: Three new NPCs generated by the agent celebrate the user's victory together, they surround the player, each taking a step forward and expressing excitement by spinning.**

## 5 EXAMPLE APPLICATION

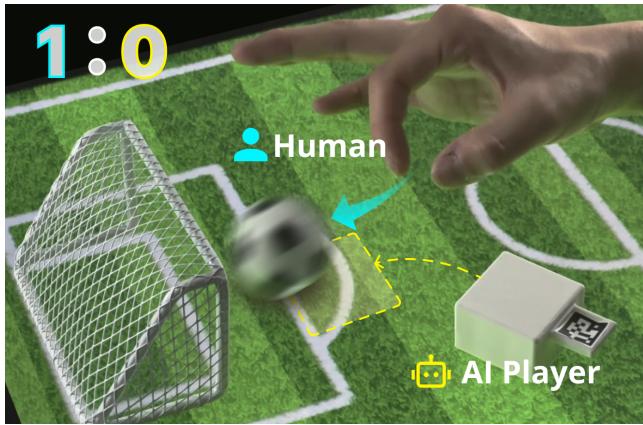
We designed several different tabletop games using the AI-Gadget Kit to demonstrate its capabilities with a two-agent system and the effectiveness of the eight add-on prompts (comprising four Interaction Behaviors and four Interaction Relationships). For each game played, we followed the procedure outlined in Figure 4, beginning

with giving the game's introduction and rules to the AI-Gadget Kit (details provided in the Supplementary Material). This initial step involved declaring the game being played. Subsequently, based on the game scenario, we provide game commands to interact with the Gadgets, engaging in an interactive gameplay experience.

### 5.1 Soccer-Ball-Shooting Game

Using the AI-Gadget Kit, we implemented a soccer-ball-shooting game, as illustrated in the Figure 13. The setup includes a stationary goal on the tabletop, with players and a Gadget taking turns to shoot the ball. Players propel the ball using two fingers, while the Gadget executes its shots by striking the ball. We began by providing the game rules into the system (details in the Supplementary Material). Once the Coordinator comprehended the rules, it assumed the roles of both host and referee within the game. It then tasked the Controller with deploying an opponent role to control the action sequences of the Gadget. The Kit shall leverage the capability through the "object actuation" add-on prompt of the Controller to simulate and execute collision planning based on the position of the goal and the ball. When it is the Gadget turn to take a shot, Controller generates the opponent's direction and speed by considering factors such as the distance of the ball from the goal, the ball's orientation relative to the goal, and the speed required to propel the ball.

Figure 13 demonstrates the performance of the system in this gaming scenario, highlighting the Gadget acting as the opponent. After the player takes a shot, the Gadget autonomously aims for the kickoff spot, moves towards the ball, and pushes it towards the goal.



**Figure 13: Human player and AI player play soccer games together.**

Additionally, the Controller monitors past scores to adjust strategies to enhance gameplay and competitiveness. For instance, in a single game session, if the player fails to score consecutively over multiple attempts, the opponent's accuracy will be dynamically lowered in subsequent rounds. This adjustment is designed to enrich the users' gaming experience by maintaining a competitive balance and ensuring that the game remains engaging and challenging without becoming discouragingly difficult.

### 5.2 Turn-Based Strategy Game

Within the Turn-Based Strategy (TBS) game scenarios, the system is designed to support battles between players whose commands are embodied through Gadget controlling (Player vs. Player, or PVP) as well as battles where the Gadget itself acts as the opponent (Player vs. Environment, or PVE). During gameplay, robotic gadgets assume roles as combatants, generating robotic action commands based on the attack targets and skills released as designated by the players. Following the game rules outlined, the Coordinator updates and transmits game information during interactions, such as the status and capabilities of the gadgets. The Controller then synthesizes this information to produce corresponding sequences of actions. In combat scenarios, our Kit primarily relies on an apprentice, opponent, and non-verbal expression add-on prompts. It adheres to the players' interaction commands (such as attack/defense, skill deployment, etc.) to generate corresponding interactive behaviors. As illustrated in the figure 14, when a player commands their character to "attack the enemy with the power of thunderbolt," the kit generates an attack action directed at the opposing Gadget. Considering the characteristics of "thunderbolt," the kit is designed with actions for the attacking gadget that simulate the buildup of electrical charge followed by a swift charge forward, and for the opponent gadget, it simulates the motion of being electrocuted with on-the-spot swaying movements. Moreover, through the add-on prompts of teammates and designers within our kit, TBS games can accommodate a larger number of gadgets for combat or interaction. This can complement the combat theater in existing Dungeons and Dragons (D&D) type games.



**Figure 14: Robotic gadget plays turn-based strategy games with human players.**

### 5.3 Yes Or No?

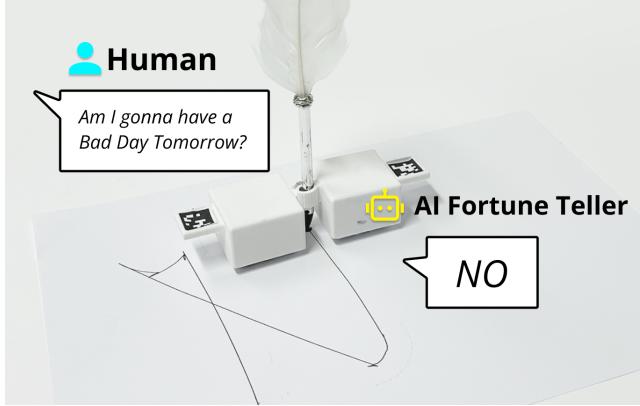
In many games, there are moments where a system provides Yes or No responses, such as ability checks and answering questions within the narrative in Dungeons and Dragons (D&D) games. Our kit enhances the presentation of Yes and No answers in these instances, making them more engaging.

For instance, we have designed a quiz game<sup>15</sup> (its rules are detailed in the Supplementary Material) where the Coordinator

accepts questions raised by users and provides corresponding answers.

The Controller, equipped with designer capabilities, utilizes add-on prompts for symbol visualization to control a Gadget. This Gadget writes out Y (Yes) or N (No) on paper, responding affirmatively or negatively to the users' queries.

To mitigate the impact of friction between the pen tip and paper on the robotic gadget's movement, we employed two gadgets to hold a pen jointly for drawing. This design is intended to offer users a more suspenseful and immersive gaming experience.



**Figure 15:** Robotic gadgets answer 'Yes' or 'No' to user questions by driving a pen to write on paper.

#### 5.4 Improvisational Theater

Improvisational theater is a form of group drama without a fixed script, where most of the content is spontaneously created by the performers during the performance[11]. Integrating AI-Gadget Kit, we have designed an application for users to perform improvisational theater in collaboration with the Gadgets. In this setup, Gadgets act as performers, with one of the Gadgets representing the user. We require the Gadgets to adhere to the core principle of improvisational theater, "Yes, and," which means freely generating subsequent performance content, including voice and action sequences, based on the previous performer's contribution and randomly selecting the next performer from among the users and other Gadgets. The Coordinator is responsible for assigning roles to all parties, recording and transmitting the content of the performance, and coordinating the turns of the performance.

After integrating non-verbal expression add-on prompts, the Controller is capable of generating dialogue that fits the characters' identities and the plot development, while also endowing robots with non-verbal "mood" expressions to convey the characters' emotional states. Moreover, by assigning specific scene information to different areas of the venue, the scene interaction capability add-on prompt enables the Controller to consider information such as the performance location when generating action sequences. In one of our improvisational theater tests set in the world of "Hamlet" (detailed game rules can be found in the Supplementary Material), our Kit initially assigns identities to a cluster of Gadgets(Figure 16),

several following the original characters of the drama and others being original characters based on user commands.

Once the performance starts, the Controller coordinates with the user, taking into account the pre-set information of the performance site, to continue the act. For example, when the robot playing Hamlet utters "To be or not to be, that is the question," in complete contrast to the original story, the user, who is consoled by a robot as Hamlet's friend (an original character), expresses his comfort and informs Hamlet that Ophelia is waiting on the terrace. Based on the lines entered by the user for the performance, the Controller Agent directs the user's robot to express emotion through a non-verbal sequence of actions. Hamlet then moved to the area representing the terrace based on the venue information and engaged in a dialog and performance with Ophelia.

It is not difficult to see that AI-Gadget has the ability to respond to high degree-of-freedom user input in performance scenarios. We also tested completely original story contexts in which AI-Gadget similarly demonstrated the ability to receive feedback on improvised content. As the art form of improvisational theater is gaining traction in the areas of imagination and creativity stimulation, mental health, and social-emotional education [25][26][3], combined with a more specialized Emergent Story Generation strategy, AI-Gadget has the potential to provide users with a richer interactive and emotional experience.



**Figure 16:** In Hamlet-themed improvisational theater, the gadget playing Hamlet flexibly reacts to the user's words outside of the script.

## 6 DISCUSSION

### 6.1 Limitation and Future Works

In this study, we employed the Sony Toio robots as our Robotic Gadget. However, due to the Toio robots' movement being reliant on motor differential speed control, the performance of the motors themselves can impact the robots' action performance. This imposes limitations on the AI-Gadget Kit's functionality. For instance, in practice, the Toio robots exhibited random deviations when executing users' movement commands, a situation exacerbated by a shift in the center of gravity due to additional attachments to the casing.

To mitigate the challenges faced by the Gadget due to the aforementioned factors, we are considering several optimizations to the system architecture. Firstly, a more precise movement correction algorithm on the robot side could support the robots' movement performance. With the anticipated action sequences of the Gadget, coupled with real-time positioning information based on ArUCo from cameras, the robots are capable of precise closed-loop control during action, enabling real-time adjustments. Furthermore, inspired by works such as Swarm Haptics[13] and Hermits[19], mechanical modifications to the robots (like increasing tire friction) could address the limitations in the Gadget's linear driving capability and performance due to the mass and speed of swarm robots in scenarios such as object propulsion.

Beyond optimizing the robots' movement capabilities, we also recognize that the current provision of meta-actions by the robot side is insufficient to meet the demand for a broader range of complex movements. For the Gadget, diverse movement modes (such as curved motion) are expected to improve the robots' interactive experience and expressiveness. Therefore, the encapsulation and provision of more movement modes will assist the Robotic Gadget in achieving interactions with complex and rich semantics in tabletop games.

On the other hand, in this study, when confronted with complex board game scenarios, such as "Improvisational Theater," the LLM (i.e., GPT-4) which the agents rely on, may encounter performance bottlenecks due to the utilization of multiple add-ons prompts and numerous game rounds, leading to complications in processing lengthy contexts. This can result in inaccuracies and errors or the so-called "hallucination" phenomenon of LLM, where the model generates content with inaccuracies related to the game's rules and scenario descriptions. In such cases, the model's output may require additional testing (e.g., The Needle In a Haystack Test) to verify its accuracy and reliability.

To address this challenge and enhance the agents' performance in complex scenarios, several strategies can be considered for the future. Firstly, introducing a more powerful LLM represents a direct solution. By enhancing the model's capability to understand and retrieve long contexts, we can directly improve agents' performance in complex interaction scenarios. However, this approach depends on external technological advancements.

Furthermore, drawing inspiration from AutoGen[33], we can introduce multiple specialized agents to handle specific contextual challenges. These specialized agents could be designed to perform distinct functions, such as generating game rules and action sequences and analyzing the consequences of players' moves. By isolating specific contexts, these specialized agents facilitate interaction among the main agents. They can work together to provide more accurate, efficient, and coherent content generation. For example, by distributing specific parts of the context and isolating the others, a certain agent can specialize in tracking the logical relationship between game states and player actions, while other agents may focus on generating descriptions and reactions that align with the specific game environment or scenario and action sequences of the Gadgets.

Through such a two-agent collaboration system, we can not only enhance the agents' performance in complex game scenarios, improve content generation speed, and reduce token consumption

to lower inference costs but also allow each agent to provide deep and precise processing within their areas of expertise. This enables the entire SUI system to better understand and reflect the complex logic of games and player interactions.

## 6.2 Beyond Action Planning

This paper primarily focuses on the integration of SUI (Spatial User Interface) with LLM-based (Large Language Model-based) agents in tabletop game scenarios. However, the HCI (Human-Computer Interaction) field has already introduced many instances where SUI is combined with other interactive scenarios, including AR (Augmented Reality) games [12], serious games [20], and remote interactions [9]. This reveals the potential for our AI-Gadget Kit to inspire further research and applications of SUI in scenarios with complex interaction tasks. In this paper, we take the first step by incorporating LLM-based agents for automated action planning in SUI. Future expansions of our kit could enhance understanding and generation of SUI's other interactive modalities. For instance, by integrating Add-ons like the Mechanical Shell [19], our kit could transform SUI action planning into planning for other interactivities (e.g., Multi-Dof, Aggregation). Similarly, by generating virtual information, our kit could extend SUI action planning into dynamic interactions that blend virtual and real elements [27]. Moving forward, we aim to explore more LLM-generated SUI interaction modalities, advancing research and application of SUI in scenarios with complex interaction tasks.

## 6.3 Availability and Applicability

Our proposed AI-Gadget Kit aims to assist researchers, board game designers, and players in creatively designing interactive experiences with automated gadgets using natural language. By leveraging the two-agent system and various add-on prompts included in the kit, users can easily modify the rules for generating sequences of interactive actions (for example, testing can be conducted directly on the OpenAI GPT-4 web client).

However, we have identified obstacles encountered when attempting to rapidly debug the dynamic effects of action sequences generated by the Kit. Users must either conduct live tests with a corresponding number of gadgets for different scenarios or plot the trajectories on a coordinate system based on the content of the action sequences to evaluate the agent-generated content. This impedes the visualization of creative interactive ideas of the users.

Recently, WRLKits[23] demonstrated a tool based on the interactive computational design approach, which visually assists designers in rapidly building personalized prototypes. Similarly, works like Habitat-Sim[24] and AI2-THOR[14] have proven the viability of simulation platforms for robot interaction design. In the future, we plan to develop simulation and design tools for the AI-Gadget Kit, enabling visualizations on web or client platforms for designing personalized interactions, thus making it more accessible and usable by many.

In addition, we have noted that the hardware costs associated with SUI capable of driving gadgets for interaction may hinder the widespread adoption of this work in board game scenarios. This is because most board game café may find it challenging to afford the costs of bulk purchasing expensive hardware platforms, such

as the Sony Toio platform. To further promote our work, we will explore the development of lower-cost hardware platforms for the AI-Gadget Kit in the future.

## 7 CONCLUSION

While Swarm User Interfaces (SUIs) have succeeded in enriching tangible interaction experiences, their limitations in autonomous action planning have hindered the potential for personalized and dynamic interaction generation in tabletop games. In this paper, We proposed an AI-Gadget Kit, a multi-agent SUI tabletop gaming system, which is designed to facilitate dynamic and complex interaction tasks in tabletop games. We first introduced the system architecture of the AI-gadget Kit, which includes a set of swarm robots to perform the gadget behaviors, and a multi-agent system responsible for executing the game and generating action plans for the swarm robots. We then elaborated the design of the multi-agent system, comprising a series of meta-motions for individual robots, two LLM-based agents for complex action planning, and a set of add-on prompts aimed at reinforcing the understanding and reacting capabilities of the agents. At last, we demonstrate four application examples using AI-gadget Kit to showcase the effect of the multi-agent-driven SUI on executing complex interaction tasks in tabletop games. We aimed to use our work as a case study to explore and inspire the application of LLMs on action planning of SUI in multiple scenarios with complex interaction tasks.

## REFERENCES

- as the Sony Toio platform. To further promote our work, we will explore the development of lower-cost hardware platforms for the AI-Gadget Kit in the future.

## 7 CONCLUSION

While Swarm User Interfaces (SUIs) have succeeded in enriching tangible interaction experiences, their limitations in autonomous action planning have hindered the potential for personalized and dynamic interaction generation in tabletop games. In this paper, We proposed an AI-Gadget Kit, a multi-agent SUI tabletop gaming system, which is designed to facilitate dynamic and complex interaction tasks in tabletop games. We first introduced the system architecture of the AI-gadget Kit, which includes a set of swarm robots to perform the gadget behaviors, and a multi-agent system responsible for executing the game and generating action plans for the swarm robots. We then elaborated the design of the multi-agent system, comprising a series of meta-motions for individual robots, two LLM-based agents for complex action planning, and a set of add-on prompts aimed at reinforcing the understanding and reacting capabilities of the agents. At last, we demonstrate four application examples using AI-gadget Kit to showcase the effect of the multi-agent-driven SUI on executing complex interaction tasks in tabletop games. We aimed to use our work as a case study to explore and inspire the application of LLMs on action planning of SUI in multiple scenarios with complex interaction tasks.

## REFERENCES

  - [1] Heike Brock, Selma Šabanović, and Randy Gomez. 2021. Remote You, Haru and Me: Exploring Social Interaction in Telepresence Gaming With a Robotic Agent. In *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction* (Boulder, CO, USA) (*HRI ’21 Companion*). Association for Computing Machinery, New York, NY, USA, 283–287. <https://doi.org/10.1145/3434074.3447177>
  - [2] Artem Dementyev, Hsin-Liu (Cindy) Kao, Inrak Choi, Deborah Ajilo, Maggie Xu, Joseph A. Paradiso, Chris Schmandt, and Sean Follmer. 2016. Rovables: Miniature On-Body Robots as Mobile Wearables. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) (*UIST ’16*). Association for Computing Machinery, New York, NY, USA, 111–120. <https://doi.org/10.1145/2984511.2984531>
  - [3] Peter Felsman, Colleen M. Seifert, and Joseph A. Himle. 2019. The use of improvisational theater training to reduce social anxiety in adolescents. *The Arts in Psychotherapy* 63 (2019), 111–117. <https://doi.org/10.1016/j.aip.2018.12.001>
  - [4] Chuang Gan, Yiwei Zhang, Jiajun Wu, Boqing Gong, and Joshua B Tenenbaum. 2020. Look, listen, and act: Towards audio-visual embodied navigation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 9701–9707.
  - [5] Sarah Gillet, Wouter van den Bos, Iolanda Leite, et al. 2020. A social robot mediator to foster collaboration and inclusion among children. In *Robotics: Science and Systems*.
  - [6] Yijie Guo, Zhenhan Huang, Ruohan Wang, Chih-Heng Li, Ruoyu Wu, Qirui Sun, Zhihao Yao, Haipeng Mi, and Yu Peng. 2023. Sparkybot: An Embodied AI Agent-Powered Robot with Customizable Characters and Interaction Behavior for Children. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (<conf-loc>, <city>San Francisco</city>, <state>CA</state>, <country>USA</country>, </conf-loc>) (UIST ’23 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 90, 3 pages. <https://doi.org/10.1145/3586182.3615804>
  - [7] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. 2023. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973* (2023).
  - [8] Xiaoyu Huang, Dhruv Batra, Akshara Rai, and Andrew Szot. 2023. Skill transformer: A monolithic policy for mobile manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10852–10862.
  - [9] Keiichi Ihara, Mehrad Faridan, Ayumi Ichikawa, Ikkaku Kawaguchi, and Ryo Suzuki. 2023. HoloBots: Augmenting Holographic Telepresence with Mobile Robots for Tangible Remote Collaboration in Mixed Reality. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (<conf-loc>, <city>San Francisco</city>, <state>CA</state>, <country>USA</country>, </conf-loc>)*. Association for Computing Machinery, New York, NY, USA, Article 119, 12 pages. <https://doi.org/10.1145/3586183.3606727>
  - [10] Chattriya Jariyavaje, Arnon Visavakcharoen, Preeyaphond Sirimaha, Booncharoen Sirinaovakul, and Jumpol Polvichai. 2018. A Practical interactive chess board with automatic movement control. In *2018 Global Wireless Summit (GWS)*. IEEE, 246–250.
  - [11] Keith Johnstone. 2012. *Impro: Improvisation and the theatre*. Routledge.
  - [12] Hiroki Kaimoto, Kyzyl Monteiro, Mehrad Faridan, Jiatong Li, Samin Farajian, Yasuki Kakehi, Ken Nakagaki, and Ryo Suzuki. 2022. Sketched reality: Sketching bi-directional interactions between virtual and physical worlds with ar and actuated tangible ui. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–12.
  - [13] Lawrence H. Kim and Sean Follmer. 2019. SwarmHaptics: Haptic Display with Swarm Robots. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland UK) (*CHI ’19*). Association for Computing Machinery, New York, NY, USA, 1–13. <https://doi.org/10.1145/3290605.3300918>
  - [14] Eric Kolve, Roozbeh Mottaghi, Winston Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. 2022. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv:1712.05474 [cs.CV]*
  - [15] Jianan Li, Mauricio Sousa, Chu Li, Jessie Liu, Yan Chen, Ravin Balakrishnan, and Tovi Grossman. 2022. ASTEROIDS: Exploring Swarms of Mini-Telepresence Robots for Physical Skill Demonstration. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (<conf-loc>, <city>New Orleans</city>, <state>LA</state>, <country>USA</country>, </conf-loc>) (CHI ’22)*. Association for Computing Machinery, New York, NY, USA, Article 111, 14 pages. <https://doi.org/10.1145/3491102.3501927>
  - [16] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What Makes Good In-Context Examples for GPT-3? *arXiv preprint arXiv:2101.06804* (2021).
  - [17] Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2023. Discuss before moving: Visual language navigation via multi-expert discussions. *arXiv preprint arXiv:2309.11382* (2023).
  - [18] Cynthia Matuszek, Brian Mayton, Roberto Aimi, Marc Peter Deisenroth, Liefeng Bo, Robert Chu, Mike Kung, Louis LeGrand, Joshua R Smith, and Dieter Fox. 2011. Gambit: An autonomous chess-playing robotic system. In *2011 IEEE international conference on robotics and automation*. IEEE, 4291–4297.
  - [19] Ken Nakagaki, Joanne Leong, Jordan L. Tappa, João Wilbert, and Hiroshi Ishii. 2020. HERMITS: Dynamically Reconfiguring the Interactivity of Self-propelled TUIs with Mechanical Shell Add-ons. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (Virtual Event, USA) (UIST ’20)*. Association for Computing Machinery, New York, NY, USA, 882–896. <https://doi.org/10.1145/3379337.3415831>
  - [20] Yu Peng, Yuan-Ling Feng, Nan Wang, and Haipeng Mi. 2020. How children interpret robots’ contextual behaviors in live theatre: Gaining insights for multi-robot theatre design. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 327–334.
  - [21] Yanyuan Qiao, Yuankai Qi, Zheng Yu, Jing Liu, and Qi Wu. 2023. March in chat: Interactive prompting for remote embodied referring expression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 15758–15767.
  - [22] Shreyas Sundara Raman, Vanya Cohen, Eric Rosen, Ifrah Idrees, David Paulius, and Stefanie Tellex. 2022. Planning with large language models via corrective re-prompts. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.
  - [23] Artin Saberpour Abadian, Ata Otaran, Martin Schmitz, Marie Muehlhaus, Rishabh Dabral, Diogo Luvizon, Azumi Maekawa, Masahiko Inami, Christian Theobalt, and Jürgen Steinle. 2023. Computational Design of Personalized Wearable Robotic Limbs. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (<conf-loc>, <city>San Francisco</city>, <state>CA</state>, <country>USA</country>, </conf-loc>)* (UIST ’23). Association for Computing Machinery, New York, NY, USA, Article 68, 13 pages. <https://doi.org/10.1145/3586183.3606748>
  - [24] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9339–9347.
  - [25] R. Keith Sawyer. 2000. Improvisational Cultures: Collaborative Emergence and Creativity in Improvisation. *Mind, Culture, and Activity* 7, 3 (2000), 180–185. [https://doi.org/10.1207/S15327884MCA0703\\_05](https://doi.org/10.1207/S15327884MCA0703_05)
  - [26] Diana Schwenke, Maja Dshemuchadse, Lisa Rasehorn, Dominik Klarholter, and Stefan Scherbaum. 2021. Improv to Improve: The Impact of Improvisational Theater on Creativity, Acceptance, and Psychological Well-Being. *Journal of Creativity in Mental Health* 16, 1 (2021), 31–48. <https://doi.org/10.1080/15401383.2020.1754987>
  - [27] Ryo Suzuki, Rubaiat Habib Kazi, Li-yi Wei, Stephen DiVerdi, Wilmot Li, and Daniel Leithinger. 2020. RealitySketch: Embedding Responsive Graphics and Visualizations in AR through Dynamic Sketching. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (Virtual Event, USA) (UIST ’20)*. Association for Computing Machinery, New York, NY, USA, Article 119, 12 pages. <https://doi.org/10.1145/3586183.3606727>

- Event, USA) (*UIST '20*). Association for Computing Machinery, New York, NY, USA, 166–181. <https://doi.org/10.1145/3379337.3415892>
- [28] Ryo Suzuki, Clement Zheng, Yasuaki Kakehi, Tom Yeh, Ellen Yi-Luen Do, Mark D. Gross, and Daniel Leithinger. 2019. ShapeBots: Shape-changing Swarm Robots. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) (*UIST '19*). Association for Computing Machinery, New York, NY, USA, 493–505. <https://doi.org/10.1145/3332165.3347911>
- [29] Albert van Breemen, Xue Yan, and Bernt Meerbeek. 2005. iCat: an animated user-interface robot with personality. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems* (The Netherlands) (*AAMAS '05*). Association for Computing Machinery, New York, NY, USA, 143–144. <https://doi.org/10.1145/1082473.1082823>
- [30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
- [31] Xin Eric Wang, Vihan Jain, Eugene Ie, William Yang Wang, Zornitsa Kozareva, and Sujith Ravi. 2020. Environment-agnostic multitask learning for natural language grounded navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*. Springer, 413–430.
- [32] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.
- [33] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155* (2023).
- [34] Lilith Yu, Chenfeng Gao, David Wu, and Ken Nakagaki. 2023. AeroRigUI: Actuated TUIs for Spatial Interaction using Rigging Swarm Robots on Ceilings in Everyday Space. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–18.
- [35] Jichen Zhu, Jennifer Villareale, Nithesh Javvaji, Sebastian Risi, Mathias Löwe, Rush Weigelt, and Casper Harteveld. 2021. Player-AI interaction: What neural network games reveal about AI as play. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–17.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009