

CARNEGIE MELLON UNIVERSITY

School of Architecture College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

[Master of Science in Computational Design]

TITLE:

Co-op Tile World: A Benchmarking Environment for
Human-AI Cooperative Interaction

AUTHOR:

Naeun (Nik) Kim

ACCEPTED BY ADVISORY COMMITTEE:

 Daragh Byrne	Principal Advisor	May 6th, 2025 _____ DATE
_____ Vernelle A. A. Noel	Advisor	May 7, 2025 _____ DATE
_____ Paul Pangaro	Advisor	May 6, 2025 _____ DATE

Abstract

Cooperative AI research is widely regarded as a next milestone in the development of advanced AI systems—where AI agents do not operate in isolation, but instead act as autonomous and social members of a larger network.

This research begins by examining cooperative behaviors through the lens of reinforcement learning. Unlike autonomous agents that perform tasks independently within an environment, cooperative tasks introduces a broader notion of **context**. In cooperative scenarios, the environment expands beyond the tangible task space to include intangible social elements such as norms and cultural expectations. Also, agents should now consider the presence of other agents as a group or individual, while the conditions for cooperation also plays into. Accordingly, an agent’s action set must include both autonomous and social behaviors. In response to these actions, agents receive not only task-related rewards, but also social rewards, and both the task state and social state evolve over time.

Social abilities are especially crucial in cooperative tasks between humans and agents, since successful cooperation hinges on providing participants with a comprehensive combination of task-related feedback and social cues. Feedback and cues allow participants to dynamically adjust their behaviors in pursuit of shared goals. Humans, for instance, do not evaluate cooperation solely based on final outcomes; they continuously adapt based on the perceived quality of interaction. Therefore, it is essential to develop agents with diverse social capabilities—built on top of reliable autonomous capabilities—and to study how those agents cooperate with real human partners.

Yet, there is currently no suitable environment for conducting cooperative AI research in a human-in-the-loop setting. While existing Multi-Agent Reinforcement Learning (MARL) environments are commonly used to benchmark agent performance, they primarily focus on AI-to-AI interactions and lack the necessary mechanisms to showcase an agent’s social capabilities—an essential component when interacting with humans.

To address this gap, this research introduces a custom benchmarking environment called Co-op Tile World, specifically designed to support experimentation on socially cooperative AI and human cooperation. Furthermore, this study leverages the environment to evaluate cooperative

performance between human players and AI agents, parameterizing the agent’s social feature—its alignment with the human’s decisions in the collective decision-making process—across two levels of task difficulty.

This work lays the groundwork for developing socially grounded cooperative agents that generate rich social feedback throughout the course of interaction. Through empirical studies using this environment, the research offers foundational insights into the nature of human–machine cooperation. For instance, our user study revealed that task difficulty had a significant main effect on cooperative performance. We also observed trends indicating that the agent’s alignment mode—and its interaction with task difficulty—influenced cooperative outcomes. These findings from human-in-the-loop experiments with diverse social agents carry broad implications for designing AI companions, socially assistive robots, and AI partners in cooperative games—agents that not only perform tasks autonomously, but also engage in context-aware social interaction.

Acknowledgments

Daragh, thank you for being there from the very beginning to the very end of this thesis. You were always quick to provide support—whether it was connecting me with resources, people, or just the right ideas. I’m especially grateful to have worked with you in the WHY Research Lab. The lab’s warm and supportive culture turned our meetings into something I genuinely looked forward to in the middle intensive writing.

Vernelle, thank you for the seriousness and integrity you bring to research, and for constantly reminding me that research isn’t just a personal pursuit—it’s a contribution to the field. When I became too caught up in my own perspective, you brought me back to the bigger picture.

Paul, thank you for embracing my far-fetched idea of exploring cybernetics through games and supporting it as an independent study. And thank you for your patience and encouragement when I struggled to put thoughts into words. Your support gave me the confidence to keep going.

Daniel, your advice during the early phase of this project helped me find my footing. Thank you.

Joshua, I loved your feedback during presentations—it always cut right to the heart of things and left me no choice but to solve the problem.

Code Lab friends, I feel incredibly lucky to have shared space with such brilliant and creative minds. Sohyun, thank you for being my desk buddy. I’ll miss our sushi party in the middle of endless assignments.

Kim, thank you for pulling me out of my room and off campus. Through you, I met so many amazing people through you, and I guess that just shows how amazing you are too.

Mom and Dad, thank you for believing in me and supporting my decision to study abroad. It’s incredibly reassuring to know that you trust me, no matter what path I take. Of all the privileges I’ve had, being your child is the greatest of all. Dad, you’re the person I learn the most from. Mom, you’re the wisest person I know.

Little brother, thank you for helping me with technical advice over Zoom. You always listened with patience and an open mind. You’re thoughtful, brilliant, and have a beautiful mind.

Byeolddong, I’m sorry I couldn’t be there at the end. If we meet again someday, let’s go everywhere together.

To all my family members, with endless gratitude and love.

Contents

Abstract	i
Acknowledgments	iii
List of Acronyms	vii
List of Figures	viii
List of Tables	x
1 Introduction	1
1.1 Games as Technical Benchmarking Environment	4
1.2 Socially Grounded Cooperative Interaction	6
1.3 Research Questions	7
1.3.1 Design: Co-op Tile World	7
1.3.2 Assessment: Game Theory	8
1.4 Structure of the Thesis	9
2 Background	11
2.1 Cooperative AI	12
2.1.1 Cooperation	12
2.1.2 Intelligence	14
2.1.3 Cooperative Intelligence	16
2.2 Robot Learning	18
2.2.1 Embodiment and Real-Time Adaptation	18
2.2.2 Learning from Action and Environment Interaction	18
2.2.3 Grounded Language	19
2.3 Human Robot Interaction	21
2.3.1 Autonomous Agent	21
2.3.2 Social Agent	23
2.4 Game Theory	28
2.5 Summary	30

3 Method	32
3.1 Overview	32
3.2 Human Machine Guitar Hero	33
3.2.1 Design	34
3.2.2 Expanding Human-Machine Guitar Hero	40
3.3 Designing Co-op Tile World	41
3.3.1 Supporting Co-op Game Play	42
3.3.2 Implementing Game Mechanics	45
3.3.3 Designing Two Levels	46
3.3.4 Communication Protocols	48
3.3.5 Developing Agents	49
3.4 User Experiment	62
3.4.1 Experiment Design	62
3.4.2 Experiment Evaluation Criteria	64
3.5 Data Collection	66
4 Result and Findings	67
4.1 Results	67
4.1.1 Demographics	67
4.1.2 Descriptive Statistics	70
4.1.3 Inferential Statistics	74
4.2 Findings	76
5 Conclusion	81
5.1 Summary	81
5.2 Contributions	83
5.2.1 Development of Custom Benchmarking Environment	83
5.2.2 User Test and Verification	85
5.3 Limitations	86
5.3.1 Agent Design	87
5.3.2 Level Design	87
5.3.3 Limited sample size	88
5.4 Future Work	89
5.5 Discussion	91

A Co-op Tile World Game Images	94
B Reinforcement Learning Agent	97
C User Experiment Raw Data	102
Bibliography	104

List of Acronyms

NLP Natural Language Processing

MDP Markov Decision Process

RoL Robot Learning

MARL Multi Agent Reinforcement Learning

HRI Human-Robot Interaction

CAI Cooperative AI

ToM Theory of Mind

RL Reinforcement Learning

IL Imitation Learning

BC Behavior Cloning

IV Independent Variable

DV Dependent Variable

OPTIME Operation Time

List of Figures

1.1 A diagram demonstrating a framework of reinforcement learning.	2
1.2 A diagram demonstrating a cooperative actions using reinforcement learning frame- work	3
1.3 A figure explaining agent's autonomous and social features.	3
1.4 A diagram explaining agent's cooperative actions adjusted by comprehensive task related and social data.	4
2.1 How required level of autonomy differs depending to the human's control of robots	22
3.1 Human-Machine Guitar Hero feedback model	35
3.2 Game Scenarios	37
3.3 Frontend and backend game structure diagram	38
3.4 Playstyle decision quadrant	39
3.5 Communication protocol between human and machine players	40
3.6 Screenshots from original Tile World 2	43
3.7 Differing team reward based on remaining time	44
3.8 Player inventory and individual reward bubble	44
3.9 Supported tile mechanics for Co-op Tile World	46
3.10 Google Spread Sheet as a level design pipeline	47
3.11 Resulting JSON file by running the script	47
3.12 Two levels demonstrating different task difficulties	48
3.13 A figure showing different items assigned to the agent and to the human player	48
3.14 Communication protocol UI	50
3.15 A diagram showing layered structure of agent design	51
3.16 A weighted graph where each edge representing cost	53
3.17 Dijkstra vs A* Algorithm comparison	54
3.18 Iterative BC pipeline with surrogate agent data	57
3.19 A diagram showing how the agent's internal planning model works	58
3.20 A diagram showing three different alignment modes of agent	59
4.1 Participant Demographics	68

4.2	Prior Gaming Experience	68
4.3	Prior Puzzle Game Experience	69
4.4	Prior Cooperative Gaming Experience	69
4.5	Bar chart comparing normalized final rewards across conditions	71
4.6	Box plot of normalized final rewards by level and agent alignment	72
4.7	Box plot of Stag-Hunt model choice counts across levels and agent alignment	73
4.8	An interaction plot showing level and alignment effect on normalized final reward.	75
A.1	Communication process	95
A.2	Demonstrating alignment mode via agent feedback	96
B.1	Training progress across different experimental conditions.	100
B.2	Evaluation trajectories of trained agents across six different runs	101
C.1	Raw experimental data collected from the Co-op Tile World user test	103

List of Tables

3.1 Machine's Action Table	40
3.2 Game mechanics used in each Level	45
3.3 Layered agent design	49
3.4 Progress of Dijkstra's algorithm showing tentative distances from start node A . .	53
3.5 Six experiment scenarios of two levels of task Difficulty and three levels of agent's alignment mode	62
3.6 Participant assignment within a single alignment mode across two task difficulties	63
3.7 Stag-Hunt game payoff matrix	64
3.8 Modified Stag-Hunt model payoff matrix M = Maximum team reward based on remaining time; I = Individual reward based on items collected	65
4.1 Mixed ANOVA Results for the Effect of Agent Alignment and Task Difficulty on Cooperative Performance.	74
B.1 PPO Training Hyperparameters	99

1

Introduction

Contents

1.1 Games as Technical Benchmarking Environment	4
1.2 Socially Grounded Cooperative Interaction	6
1.3 Research Questions	7
1.3.1 Design: Co-op Tile World	7
1.3.2 Assessment: Game Theory	8
1.4 Structure of the Thesis	9

Citations in this chapter: [1]–[9].

Acronyms in this chapter: Cooperative AI (CAI), Human-Robot Interaction (HRI), Reinforcement Learning (RL).

Among the many forms of interaction, cooperative interaction resonates with me the most—particularly the shared experiences of playing co-op video games with siblings and friends. Those moments—navigating challenges together, sharing laughter and shouts of victory, and building trust over time—are deeply meaningful. Such rich and nuanced interactions prompt the question: Can we emulate these dynamics with machine partners?

Supporting high-fidelity cooperative human-machine interaction is one of the next major challenges in Cooperative AI. As J. C. R. Licklider envisioned in *Man-Computer Symbiosis* [6], human-machine collaboration should evolve into an intellectual partnership in which both agents contribute to decision-making, problem-solving, and situational control. This vision is becoming increasingly tangible with recent advancements in Artificial Intelligence (AI) and machine learning. With this exciting task in mind, this thesis explores fundamental questions about the nature of cooperation between humans and machines and proposes a practical tool for Cooperative AI (CAI) and

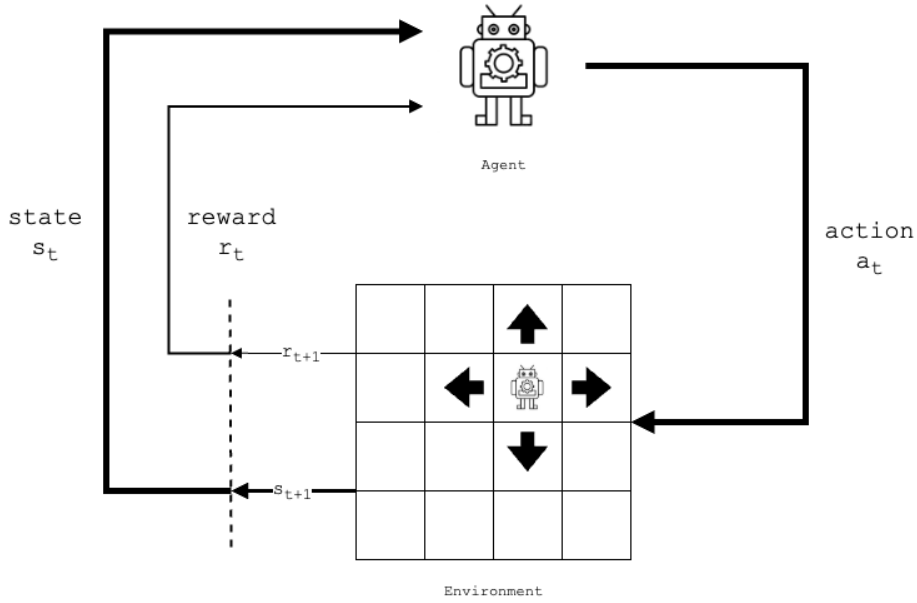


Figure 1.1: Reinforcement learning framework

Human-Robot Interaction ([HRI](#)) research. Ultimately, it aims to contribute to the development of *socially grounded* cooperative interactions, a concept that will be further elaborated in the following section.

To investigate cooperative interactions systematically, we adopt the Reinforcement Learning ([RL](#)) framework—a foundational paradigm in machine learning in which an agent learns an optimal policy to achieve a given goal. As illustrated in [Figure 1.1](#), in classic reinforcement learning, an agent interacts with the environment by taking actions, observing the resulting state changes, and receiving associated rewards. Over time, the agent adjusts its behavior to maximize the cumulative reward based on these experiences. Central to this process is the formation of a **policy**—a strategy that the agent uses to decide which action to take in a given state.

How does this framework apply to cooperative actions? [Figure 1.2](#) demonstrates the core argument of this thesis that in cooperative scenarios, the environment must be expanded to a broader “context” that includes not only the tangible task space but also intangible environment factors such as social norms, cultural expectations, other agents as individuals or groups, and conditions to cooperate. In addition, the action set of an agent should encompass both autonomous and social behaviors. In response to these actions, agents receive not only task-related rewards, but also social rewards, and the task state and the social state evolve accordingly.

What are social actions, and how do they differ from an agent’s autonomous, task-related actions? Before moving forward, it is useful to distinguish between an agent’s autonomous

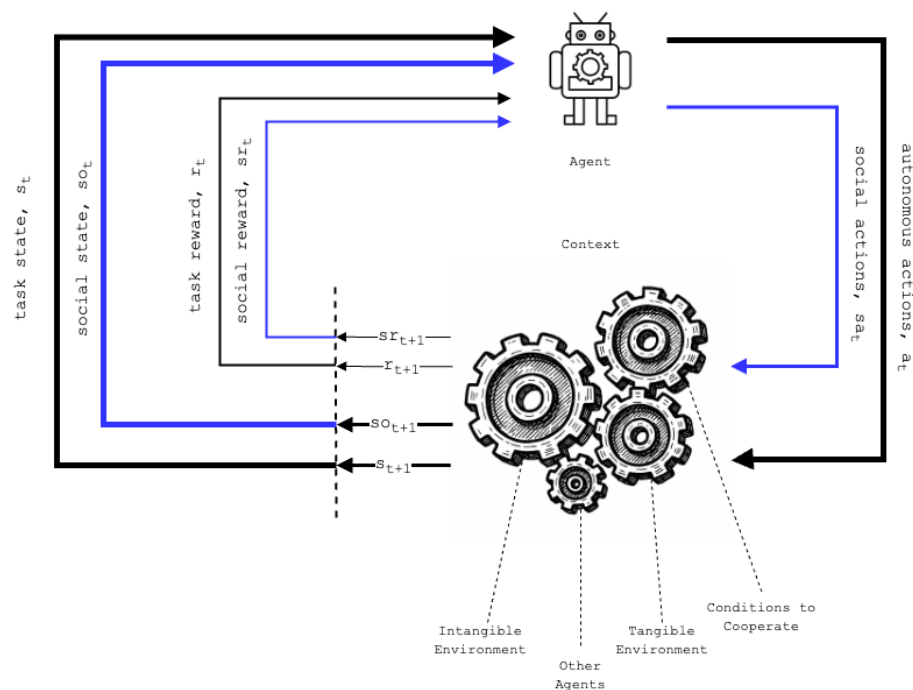


Figure 1.2: Interpreting nature of cooperative actions using reinforcement learning framework

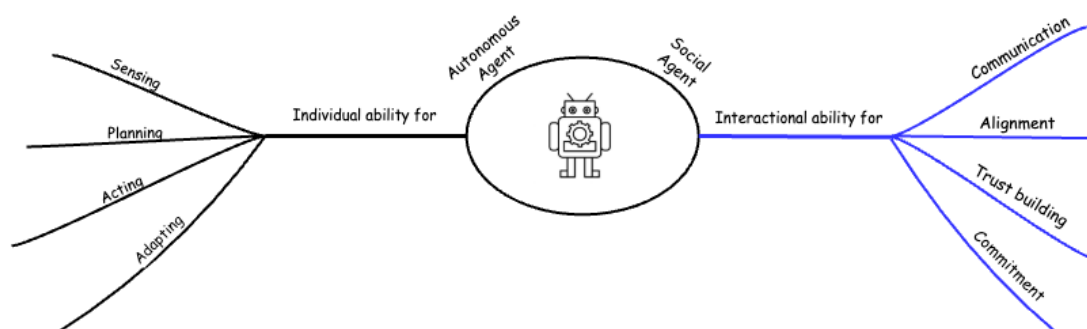


Figure 1.3: Agent's autonomous and social features.

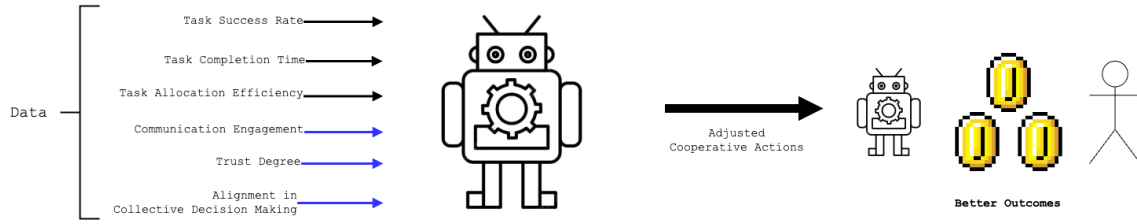


Figure 1.4: Social agent's cooperation and feedback system

actions and its social actions. In the Human-Robot Interaction ([HRI](#)) research domain, autonomous actions refer to behaviors an agent performs independently to regulate its own functioning—for example, perceiving the environment, planning next steps, executing movements, and adapting to new conditions. In contrast, social actions are behaviors that take other agents into account. These include communicating, coordinating, aligning goals, building trust, and making or responding to commitments. Figure [1.3](#) illustrates facets of agent's autonomous and social actions.

For agents performing cooperative tasks with human partners, social abilities are especially crucial. Humans do not evaluate cooperation solely based on final outcomes; rather, they continuously adapt their behavior in response to the perceived social quality of the interaction—for example, the fluidity of communication, a partner's sense of commitment, and the degree of mutual trust. As illustrated in Figure [1.4](#), successful cooperation hinges on providing participants with a comprehensive combination of task-related and social feedback, enabling them to dynamically adjust and improve their cooperative performance. As Dafoe et al. [4](#) point out, cooperative intelligence requires more than just understanding—the ability to perceive, plan, and act toward a goal. It also involves core social competencies such as communication, making commitments, and sensitivity to social norms and institutions.

1.1 Games as Technical Benchmarking Environment

In the development of AI agents, games have been widely used as technical benchmarking tools to assess the performance of agents. This is because games provide virtual environment where agents can perform on and games have set of rules to evaluate the outcome of actions. Games could be used effectively to represent scenarios in the real world, the game could be simple as classic Atari Games or it could be complex as games like Chess [3](#), GO [8](#), and Starcraft [1](#).

This trend continues in the development of cooperative agents. For instance, the Multi-Agent

Reinforcement Learning library PettingZoo [9] offers a wide range of benchmarking environments. While games like Basketball Pong, Boxing, and Tennis [2] support competitive multi-agent training, environments such as Emtobed: Cooperative [2], Multiwalker [5], Pursuit [5], and Waterworld [5] are specifically designed to model cooperative scenarios. These pre-built environments enable researchers to focus on developing and tuning algorithms, without the overhead of designing custom testing platforms. Typically, researchers only need to process observation inputs using their algorithm and generate corresponding actions, allowing for streamlined experimentation.

Despite their cleverness and utility in developing AI agents, these benchmarking environments have limitations, particularly for human-agent interaction assessment. The limitations include:

1. Lack of Human-Agent Interaction Focus: These environments are not designed to assess interactions between humans and agents.
2. Fixed Design: They are often fixed, providing little room for researchers to test their own cooperative task.
3. No Embedded Communication Protocol between players: There is typically no built-in mechanism for human-AI communication.
4. Difficulty in visualizing agent’s social features: It is challenging to visualize detailed cooperative social features such as alignment, commitment, trust, or the agent’s adaptability to human partners.

Benchmarking Environment for Humans-AI Cooperation

Based on the identified limitations, this research proposes a benchmarking environment designed specifically for real-time human-agent interactions. Our objectives for this environment are that it should:

- a) Be intuitive enough to operate easily for human players and at the same time, it should involve a room to train simple to sophisticated agent’s behavior.
- b) Be easily designable for researchers to fine-tune the purpose of the environment based on their research purpose.
- c) Provide communication protocol between human and AI agent.

- d) Be flexible enough to accommodate key social features of agents, such as alignment with human partners, commitment to shared tasks, trust formation, and explainability of actions.

The need for a dedicated benchmarking environment for human-machine cooperation stems from the inherently social nature of cooperative interactions. For AI to effectively support these dynamics, it is essential to examine how the social attributes of agents, including their ability to build trust, align and adapt to human partners, and respond to social norms and institutions, influence cooperative performance and enable socially grounded cooperative interaction.

1.2 Socially Grounded Cooperative Interaction

Cooperation is a highly advanced form of social capability. When we cooperate, we not only try to maximize mutually beneficial decision-making and action, but also generate rich social dynamics throughout interaction.

When collaborating, humans engage in social interactions in real time, continuously evaluating factors such as communication quality, goal alignment, and mutual trust—abilities that vary depending on the context. These dynamic cues shape individuals' cooperative behavior. For example, if communication becomes unclear or misaligned, one partner might adjust their actions to reestablish clarity and coordination, whereas in interactions where communication is smooth, goals are aligned, and trust is high, participants are more likely to maintain their roles and remain consistently engaged.

Furthermore, the expectations and expressions of these social abilities vary across contexts, influencing the trajectory and quality of cooperation. For example:

- In a friendly, intimate setting, a cooperative AI should engage in clear, empathetic communication—actively listening, confirming alignment with the human partner's goals, and fostering trust by consistently offering supportive feedback.
- In a competitive, team-based challenge, an AI teammate might adopt a more dynamic approach by critically evaluating the human partner's strategies and offering constructive challenges or counterarguments to ensure optimal strategies and the best possible outcomes through active debate.

These context-grounded social abilities are vital because they provide real-time feedback that influences cooperative behavior. By continuously adjusting actions based on dynamic cues—such as changes in communication quality, goal alignment, and mutual trust—participants can fine-tune their cooperative efforts before a final outcome is reached. This process underscores the importance of designing AI systems that are not only effective in strategic decision-making but also adept at supporting the rich, situational dynamics inherent in human social interactions. In this sense, I define **socially grounded cooperative interaction** as an interaction that generates rich social data throughout the course of cooperation, so that participants can adjust their behavior to achieve better results.

1.3 Research Questions

This thesis is structured into two main parts: first, the development of the custom benchmarking environment, and second, the assessment of cooperative performance between humans and agents using this environment.

- RQ1 (Design): How should we design a benchmarking game environment for cooperation between humans and socially intelligent agents?
- RQ2 (Assessment): How can we assess cooperative performance between humans and socially intelligent agents using this custom environment?

1.3.1 Design: Co-op Tile World

In response to the limitations of existing game environments for benchmarking cooperative performance that involves real human—and recognizing the need for environments that support socially grounded human-AI interaction—this thesis introduces **Co-op Tile World**. Unlike conventional reinforcement learning benchmarks, which are primarily designed for AI-to-AI interaction or rely on proxy humans.

Co-op Tile World is built on top of the open-source game Tile World 2 [7], originally developed by Brian Raiter, Madhav Shanbag, and Eric Schmidt. Based on this foundation, Co-op Tile World was redesigned as a two-player cooperative game. One powerful feature of this benchmarking environment is its level editor, which operates through Google Sheets. This tool allows researchers to easily design custom game environments tailored to their specific research objectives, freeing

them from the limitations of pre-defined levels. The game includes rules that emphasize cooperative gameplay, various tile mechanics, and user interfaces that support human interaction with the system.

The second part of this research involves using Co-op Tile World to conduct a user study that explores the research question: How does an agent’s alignment in strategic decision-making affect cooperative performance under varying task difficulty? This question investigates whether cooperative performance differs depending on how the agent aligns with its human partner during collective decision-making, and whether this effect interacts with the difficulty of the task.

The degree of players’ alignment in decision-making will be categorized into three levels:

- **Aligned** – The agent always follows the human’s decisions.
- **Merged** – The agent proposes its own plan to the human partner.
- **Diverged** – The agent always rejects the human’s plans.

The task difficulty will be classified into two levels:

- **Easy** – A task requiring only a basic understanding of the game rules.
- **Difficult** – A task involving more complex rules and coordination.

This configuration results in six scenarios: (Easy, Aligned), (Easy, Merged), (Easy, Diverged), (Hard, Aligned), (Hard, Merged), and (Hard, Diverged). This structured experimental setup examines how the social behavior of agents—in particular, the degree of alignment in decision making—interplays with different task difficulties to affect cooperative performance.

1.3.2 Assessment: Game Theory

Second objective of this thesis is to assess the cooperative performance of human and machine partners using a custom-designed Co-op Tile World environment. The experiment systematically varies two key factors: task difficulty (Easy and Difficult) and the alignment mode in collective decision-making (Aligned, Merged, and Diverged). Using a mixed experimental design—with alignment mode as a between-subjects variable and task difficulty as a within-subjects variable—the study collects players cooperative performance as how much reward they gained throughout the play.

For evaluation, we'll use game theories. Game theories are one of the most foundational research area in Cooperative AI, since it have provided theoretical foundations to interpret complicated and dilemmatic cooperative scenarios in quantifiable way.

In this study, we particularly adopt the Stag Hunt model. The structure of this model is straightforward: when two hunters cooperate, they can hunt a stag, which offers the highest incentive. However, if one hunter defects while the other chooses to cooperate, the defector hunts a rabbit while the cooperator gains nothing. When both hunters choose to defect, they each end up hunting a rabbit, yielding smaller incentives.

Through this empirical study, the research offers insights into the nature of cooperation and provides implications for designing socially cooperative agents. In addition, the experiment serves as a concrete use case of how Cooperative AI (CAI) and Human-Robot Interaction (HRI) research can be conducted using the Co-op Tile World benchmarking environment.

1.4 Structure of the Thesis

Up to this point, we have examined the need for a dedicated benchmarking environment for human-AI cooperation, discussed key principles for designing such an environment, and explored methods for evaluating cooperative performance within it.

With this foundation set, the rest of the thesis is structured into four key sections:

1. **Background** This section explores the foundational research fields that inform this study, examines their interconnections, and identifies the key research domains upon which this thesis is built. The thesis is positioned at the intersection of Cooperative AI, Robot Learning, Human-Robot Interaction, and Game Theory—four extensive fields of study. While each of these areas is vast on its own, this section will break them down into relevant subtopics, establishing a clear territory map for the research.

2. **Method**

The method section begins with an overview of a preliminary experiment titled Human-Machine Guitar Hero, a modified two-player version of the original Guitar Hero game in which humans and AI agents must cooperate to achieve higher scores. This experiment provides foundational context for the research and illustrates how early insights shaped the development of this thesis.

Following this, the thesis addresses RQ1: How can we effectively design a benchmarking environment for human-AI cooperative interaction? This part of the methodology focuses on identifying the limitations of existing benchmarking environments in AI research and explains how the proposed Co-op Tile World addresses these shortcomings. In addition, I introduce an experimental design built on top of Co-op Tile World, demonstrating how the environment can be applied to cooperative AI research.

The next section describes the development of cooperative AI agents designed to play the game alongside human partners. For this thesis, agents were implemented using Imitation Learning and Path Planning algorithms. The specific implementation details and rationale for choosing each technique are discussed.

Lastly, the thesis addresses RQ2: How can we assess cooperative performance between humans and socially intelligent agents using this custom environment? This section outlines the evaluation strategy, including the use of a game-theoretic framework for assessing cooperation and the statistical methods used to analyze the results.

3. **Results and Findings** This section presents the results of the experiment conducted using the Co-op Tile World benchmarking environment, as described in the Method section. Both descriptive and inferential statistical approaches are used. Descriptive analysis includes supplementary graphs and visualizations to illustrate the results. For inferential analysis, a Mixed Analysis of Variance (Mixed ANOVA) is employed to evaluate the effects of experimental conditions on cooperative performance.
4. **Conclusion** The final section summarizes the contributions of this study and acknowledges its limitations. It also explores potential directions for future research and concludes with a discussion of the utility of this research in the fields of [CAI](#) and [HRI](#).

2

Background

Contents

2.1 Cooperative AI	12
2.1.1 Cooperation	12
2.1.2 Intelligence	14
2.1.3 Cooperative Intelligence	16
2.2 Robot Learning	18
2.2.1 Embodiment and Real-Time Adaptation	18
2.2.2 Learning from Action and Environment Interaction	18
2.2.3 Grounded Language	19
2.3 Human Robot Interaction	21
2.3.1 Autonomous Agent	21
2.3.2 Social Agent	23
2.4 Game Theory	28
2.5 Summary	30

Citations in this chapter: [4], [10]–[43].

Acronyms in this chapter: **CAI**, **HRI**, Markov Decision Process (**MDP**), Natural Language Processing (**NLP**), Robot Learning (**RoL**), Theory of Mind (**ToM**)

This background introduces four key domains—Cooperative AI, Robot Learning, Human-Robot Interaction, and Game Theory—that together form the conceptual territory map for this thesis. We begin by examining what Cooperative AI entails, first unpacking the concepts of cooperation and intelligence individually. Building on this, we turn to Robot Learning, a field that emphasizes embodiment and real-time adaptation in intelligent systems, and serves as a crucial technical foundation for realizing Cooperative AI. Next, we explore Human-Robot Interaction, which examines the social qualities of robots through empirical studies with humans. Finally, we visit Game

Theory, a foundational pillar in Cooperative AI that provides theoretical frameworks for analyzing complicated and dilemmatic cooperative scenarios in quantifiable way.

2.1 Cooperative AI

Cooperation is a fundamental challenge in Artificial Intelligence (AI), as it is essential for enabling AI agents to work alongside humans and other AI systems effectively. Throughout history, cooperation has been key to human survival and success, shaping our societies, economies, and technological advancements. [38] As AI continues to evolve, we envision a future where AI agents do not merely function in isolation but instead form large-scale cooperative networks, collaborating with both other AI systems and human users. This raises a critical question: How should cooperation between humans and AI be structured and facilitated?

According to Dafoe, Allan, et al. in their paper *Open Problems in Cooperative AI*, problems of cooperation are defined as “problems in which agents seek ways to jointly improve their welfare.” [17] Research in Cooperative AI focuses on designing intelligent agents capable of effective collaboration, developing frameworks that promote teamwork between AI systems and humans, and drawing insights from social science, psychology, and behavioral studies to better understand the mechanisms behind cooperation and collective problem-solving. [17]

More specifically, Cooperative AI research spans multiple domains, including multi-agent systems, game theory, social choice theory, human-machine interaction and alignment, natural-language processing, and the development of social tools and platforms. [17] However, Cooperative AI is not merely a union of these research fields—it introduces a new paradigm that views intelligence not as an isolated capability but as fundamentally social intelligence. This shift in perspective creates vast research opportunities, particularly in areas where Cooperative AI interacts with other disciplines.

2.1.1 Cooperation

What is Cooperation?

According to Joel Sachs in *The Evolution of Cooperation*, [39] cooperation is an action by one individual that benefits one or more others. There are two main perspectives on how cooperation has evolved:

- The Evolutionary Perspective – Cooperation emerges through direct and indirect benefits, as well as shared genetic ties.
- The Cultural Perspective – Cooperation is a product of cultural learning, reinforced and maintained by social norms and institutions.

The study of cooperation is a vast field, but the key takeaway is that cooperation evolves through repeated interactions between individuals and is shaped by environmental reinforcement. One of the best examples of this iterative nature of cooperation is the Repeated Prisoner's Dilemma.

In the classic Prisoner's Dilemma, the most individually rational choice is to betray one's partner to maximize personal gain. However, when the game is played repeatedly, allowing players to remember past interactions and adjust their strategies accordingly, the optimal solution shifts from betrayal to cooperation. Over multiple iterations, players recognize that sustained cooperation yields the best long-term outcome for both parties.

This dynamic reflects real-world social interactions. We live in a networked society where interactions are repeated, observed, and recorded. This repetition allows individuals multiple opportunities to evaluate past interactions and adjust their behavior accordingly, maximizing their own long-term benefits. Because cooperation provides a survival advantage, memories of cooperative behavior are ingrained in our genetic inheritance and passed down through generations. At the same time, it is embedded as a cultural norm to promote social stability.

Cooperation Beyond Biology: A Social Model

Given the complexity of modern society, cooperation is no longer just a biological survival strategy but has become a fundamentally social behavior model. According to Henrich and Muthukrishna, cooperation is not solely a product of biological evolution but is shaped by cultural transmission, institutions, and learned social norms. [24]

This is evident in game-theoretic experiments that examine cooperation in varied social and cultural contexts. These studies reveal that certain social mechanisms play a crucial role in sustaining and reinforcing cooperation. Specifically, reputation, punishment, and signaling were effective in maintaining and reinforcing cooperative behavior, building social trust, and resolving free-rider problems. [24]

These findings extend beyond human-human cooperation and can be applied to AI-AI and

human-AI cooperation, providing insights into how artificial agents can be designed to foster sustainable collaborative interactions.

The Necessity of Empirical Research in Cooperation Studies

If we define cooperation as a process in which individuals, through repeated interactions, come to recognize that mutual benefit yields the most favorable outcomes and refine their decision-making models accordingly, then empirical research becomes essential for uncovering the factors that trigger, reinforce, evolve, and sustain cooperative behavior. This empirical nature of cooperation studies underscores the importance of experiments in which humans and machines engage in real-time cooperative interactions.

2.1.2 Intelligence

The concept of intelligence has been widely discussed and defined across multiple disciplines, with significant efforts made to replicate it in machines. What constitutes intelligence, and how has it been formulated across different domains? This section examines intelligence from various perspectives, outlining its foundational principles and implications in artificial intelligence research.

Intelligence as a System

One prevalent approach conceptualizes intelligence as a *system*—a structured process encompassing *sensing, recognition, decision-making, and execution*—as described by Andrew et al. [32]. A key characteristic of this perspective is its **hierarchical structure**, where intelligence is organized into layered components. However, for a system to be truly intelligent, it must exhibit three critical properties: **modularity**, **adaptability**, and **autonomy**.

- Modularity refers to the system's composition of smaller, independent units that collectively form a larger architecture.
- Adaptability denotes the system's flexibility in responding to new and changing environments.
- Autonomy emphasizes minimal reliance on external intervention.

This systems-based perspective is particularly useful for analyzing intelligence as a structured, process-driven mechanism, allowing for decomposition into distinct functional components.

Intelligence without Representation

Contrasting the systemic approach, Rodney Brooks introduced the concept of **intelligence without representation** in the 1980s. In his paper *Intelligence without Representation*, he coined the term **creature** to describe intelligence as an emergent property of behavior rather than a structured system. He argued,

“The Creature itself has none; it is a collection of competing behaviors.” (p.149) [12]

This perspective is provocative, as it fundamentally rejects the recognition and decision-making layers present in system-based models. Instead, it proposes that intelligence arises solely from the direct mapping between perception and execution. According to Brooks, complex intelligence is not predefined but rather a byproduct of interacting with a complex environment.

While this approach has faced limitations—directly mapping perception to execution has not yet led to significant breakthroughs—it introduces a crucial insight: **situated intelligence**. Intelligence cannot be fully understood or realized in isolation; it must be examined within the context and environment in which an agent operates.

Universal Intelligence

Attempts have also been made to define a **universal measure of intelligence**, particularly in machines. Legg et al. [29] proposed a formula to quantitatively assess intelligence across diverse agents and environments. They highlight that existing benchmarks, such as the *Turing Test* or game-based assessments, are limited in scope—one fails to test general problem-solving abilities, while the other lacks applicability to broader real-world scenarios.

To address this, they define intelligence as an agent’s ability to achieve its **goals** across **diverse** environments. Their proposed mathematical formulation is as follows:

$$\Upsilon(p) = \sum_{e \in E} 2^{-K(e)} V_e^p \quad (2.1)$$

where E represents all possible environments, $K(e)$ denotes the complexity of an environment, and V_e^p measures the agent p ’s reward in environment e . This equation encapsulates the ambition of modeling intelligence as a generalized framework applicable to all possible environments. It

remains a foundational pillar in AI research, reflecting ongoing efforts to develop comprehensive and universal models of intelligence.

Cybernetics Perspective

Cybernetics offers another perspective on intelligence, focusing on feedback loops and communication systems. Wang et al. define cybernetics as “the science of communication and autonomous control in machines and living things,” originally proposed by Norbert Wiener in 1948 [42].

Unlike system-based or representation-free approaches, cybernetics interprets intelligence as a **dynamic conversational state**, where agents and environments continuously influence each other within a complex feedback loop. This perspective extends beyond individual intelligence, providing a framework to analyze intelligence at the level of organizations, societies, and even governments.

One practical application of this approach is the concept of an *Intelligent Organization*, which applies cybernetic principles to enable systemic self-regulation and adaptation in large-scale entities [40]. By structuring and fine-tuning feedback loops, cybernetics suggests that intelligence does not merely reside within individual agents but emerges from the interactional dynamics of an entire system.

Conclusion

The various perspectives on intelligence—whether as a structured system, an emergent behavior, a universal capability, or a cybernetic process—each contribute unique insights into its nature. While system-based and universal intelligence models attempt to formalize intelligence into structured mechanisms, representation-free and cybernetic perspectives emphasize the importance of interaction and environmental context. Understanding these diverse viewpoints provides a more comprehensive foundation for studying and developing cooperative intelligence in AI agents.

2.1.3 Cooperative Intelligence

Now, we have gone through the concepts of intelligence and we need to identify what *Cooperative Intelligence* is. Is it different from the definitions of intelligence? If so, how is it different and what are the core features of Cooperative Intelligence?

According to *Cooperative AI: Machines Must Learn to Find Common Ground*, Allan Dafoe states: [4](#)

“But autonomy is rarely regarded as the sole goal for humans. Similarly, certain kinds of autonomy in AI systems are useful precisely because they enable the system to contribute effectively to broader cooperative efforts.”

This implies the research necessity of Cooperative AI in specific. The literature also defines the four elements of Cooperative Intelligence, which are **understanding**, the ability to reflect, behave, and reason, **communication**, the ability to explicitly and credibly share information with others, **commitment**, the ability to make credible promises, and **norms and institutions**, social infrastructure that reinforces cooperation. Below is the detailed expected features for cooperative intelligence that Allan Dafoe itemized [4](#).

- **Understanding** The ability to take into account the consequences of actions, to predict another’s behavior, and the implications of another’s beliefs and preferences.
- **Communication** The ability to explicitly and credibly share information with others relevant to understanding behavior, intentions, and preferences.
- **Commitment** The ability of make credible promises when need for cooperation
- **Norms and Institutions** Social infrastructure – such as shared beliefs or rules – that reinforces, understanding, communications, and commitment.

These four elements serve as a valuable reference point for envisioning cooperative AI. The understanding component emphasizes an agent’s ability to perceive, plan, and take new actions in real time based on its observations of an ongoing situation. The communication element aligns with the cybernetic perspective, highlighting the importance of continuous information exchange. The last two elements incorporate key findings from cooperation studies, demonstrating that cooperation emerges through repeated interactions and is ultimately sustained and reinforced by social and cultural norms.

Bringing these aspects together, we can infer that cooperative intelligence is built upon autonomous intelligence but further requires an agent’s ability to communicate, engage in iterative interactions, and navigate social environments. Therefore, achieving perfect autonomy in an agent does not necessarily translate into cooperative intelligence. This distinction underscores the need for dedicated research into AI’s cooperative intelligence, separate from general AI autonomy.

2.2 Robot Learning

Unlike traditional machine learning approaches, Robot Learning (RoL) focuses on embodiment and real-time adaptation, making it highly relevant for creating AI agents that must engage in interactive and dynamic cooperation.

Although RoL is a subset of Machine Learning (ML), it differs significantly from standard statistical pattern recognition models [27]. It introduces two core features that are particularly relevant to the implementation of CAI, described in the following sections.

2.2.1 Embodiment and Real-Time Adaptation

Since robots operate in real-world environments under physical constraints, learning to interact dynamically within these constraints is essential. Unlike static models, robots must react in real time rather than merely predict outcomes. This focus has led to advancements in real-time learning, one-shot learning, and agent communication, which are critical for developing cooperative multi-agent systems. Key research areas at the intersection of RoL and Cooperative AI include emergent cooperation, trust development, task allocation, multi-agent learning, and shared reward and decision-making in teams.

2.2.2 Learning from Action and Environment Interaction

Unlike traditional ML models, where training data is pre-collected and fixed, RoL agents actively generate their own training data through interaction with their environment. This characteristic highlights the importance of action space design and environmental constraints, as these factors directly influence what the agent learns. To restate, bounded action sets that robots can perform and defined environment that robots act on, impact the data agent is collecting and eventually their learning outcome. This remains one of the key challenges in robotics—developing general-purpose, well-functioning robots capable of operating effectively in the physical world.

However, from a designer’s perspective, these constraints present not just limitations but valuable opportunities. By strategically designing the environment, researchers can influence how agents interact with their surroundings, enabling them to learn, adapt, and perform tasks more efficiently. In other words, clever environmental design can enhance an agent’s ability to act intelligently and effectively, even in complex behaviors such as cooperation.

While **RoL** has primarily been applied in physical robotics, its principles are equally valuable for virtual agents operating in digital environments. Even though this thesis does not develop physical robots, the **RoL**-inspired approach provides a strong foundation for modeling cooperative AI agents in interactive virtual spaces. Cooperation inherently requires an iterative feedback loop, continuous adaptation, and shared communication, making **RoL** a powerful framework for designing AI systems that learn cooperation dynamically rather than executing pre-programmed strategies.

2.2.3 Grounded Language

The concept of Grounded Language illustrates how **RoL** distinctly differs from general AI learning. To illustrate this distinction, this section explains how grounded language sets itself apart from traditional Natural Language Processing (**NLP**).

A key difference between **NLP** and grounded language—also referred to as robotic language, is that robots must map language to their physical environment and corresponding actions—a process known as grounding. This adds another layer of complexity to an already intricate **NLP** process, making grounded language processing noisier and more data-intensive. The core objectives of robotic language include:

- Understanding natural language requests in relation to the physical environment in which the robot operates.
- Leveraging language as a tool to enhance learning about the physical world.
- Engaging in collaborative discussions with humans to facilitate interaction and cooperation.

To achieve these goals, Tellex et al. explore three key approaches to realizing robotic language—learning-based methods, logic-based methods, and human-robot interaction (**HRI**) methods [41]. Compared to general **NLP**, robotic language requires additional processing layers such as grounding, grounded representation, and physical action execution, along with enhanced perception and world representation [41].

Lexically-Grounded Methods

Lexically-grounded methods focus on defining word meanings within a symbolic system, utilizing logic, grammars, and other linguistic structures to ground language. Interestingly, this approach

employs grounding tokens, such as knowledge bases or cognitive architectures, as a foundational language system.

On top of these grounding tokens, it adds a layer of formal reasoning, which typically attaches semantic structures to lexical systems. This approach provides the most structured method for analyzing grounded language. However, this characteristic becomes a limitation particularly in handling unexpected inputs that deviate from formal language rules and in scalability, as it struggles to generalize effectively across diverse scenarios.

Learning Methods

Robots learn word and utterance meanings from large datasets, drawing inspiration from machine learning and computational linguistics. This method is currently the most widely adopted approach in academia, benefiting from rapid advancements in deep learning algorithms. When trained on high-quality and diverse datasets, this approach proves to be robust and scalable, allowing robots to generalize across varied real-world scenarios.

Specific techniques include CNN, RNN, and DQN, which serve as foundational models for more advanced language architectures such as Seq2Seq, LSTM, and Transformers. However, this approach faces challenges, particularly in the mismatch between training data and real-world robotic tasks, as well as in defining an effective domain of possible word meanings.

Recent advances in machine learning-driven approaches mitigate these challenges. For instance, systems that learn latent spaces directly from data, rather than human-labeled annotations, are improving adaptability and reducing reliance on predefined linguistic structures.

HRI Methods

This approach explores how **NLP** research both supports and benefits robots that interact directly with humans. More specifically, it highlights three key aspects: (1) using language as a tool to support robots in learning other tasks, (2) leveraging robots as an ideal testbed for understanding physically situated language, and (3) integrating these two efforts to develop systems that use language for robust human-robot interaction.

Languages that facilitate robot learning are particularly compelling, as they often involve a human-in-the-loop approach, where human guidance plays a role in the robot's learning process. This perspective is valuable in addressing human-robot alignment and shared autonomy.

Furthermore, grounded language serves as a crucial modality for human-robot communication, especially in social and cognitive support applications. Examples include tutoring robots for children’s education and socially assistive robots for the elderly. These application-driven approaches not only advance physically situated language understanding, but also pave the way for designing more engaging and natural verbal interactions between humans and robots.

The grounded language example provides a clear overview of how robotic’s learning approach differs from unembodied AI processes and reversely how robot learning approaches can contribute to grounded AI research.

Furthermore, it is plausible to argue that grounded conversation will eventually become the dominant paradigm for human-machine verbal interaction, leveraging the multimodal sensory capabilities embedded in the devices we use every day.

2.3 Human Robot Interaction

While [RoL](#) provides the technical algorithms by which AI agents acquire cooperative behaviors, Human-Robot Interaction ([HRI](#)) offers a theoretical framework for structuring and understanding cooperation between humans and AI agents.

2.3.1 Autonomous Agent

One distinct feature we expect from well-performing robots is autonomy. Technically, designing autonomous machines requires receiving inputs from the surrounding environment and linking those inputs to output actions—executed by actuators or motors in the case of physical robots [\[20\]](#). Conceptually, according to Goodrich et al., a system with a high level of autonomy is one that can be left unattended for extended periods without human intervention [\[20\]](#). As shown in [Figure 2.1](#) in [HRI](#) the level of autonomy is determined by the function and task the robot is designed to perform. In other words, from an [HRI](#) perspective, the required level of autonomy varies depending on the means of operation. The [Figure 2.1](#), originally suggested by Goodrich et al., illustrates these variations.

Although robot autonomy can be implemented using various approaches—including control theory, artificial intelligence, signal processing, cognitive science, and linguistics [\[20\]](#)—the core objective of these technologies remains the same: enabling robots to adjust their behavior in real

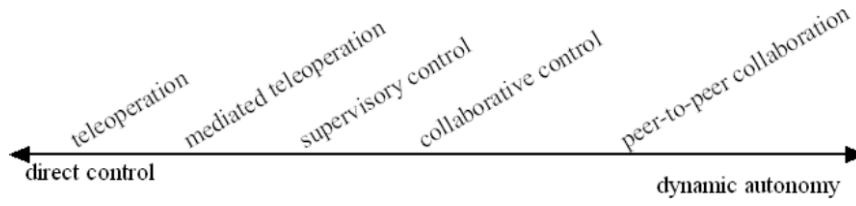


Figure 2.1: How required level of autonomy differs depending to the human’s control of robots.
 Original diagram presented by Goodrich et al., [20],
 Fig. 4.2 Levels of Autonomy

time based on changes in their environment. The environment might be static, changing only as a result of the robot’s actions, or dynamic, involving interactions among multiple agents. When the environment includes real human actions and interactions with robot agents, the adaptation problem extends to envisioning robots as social agents, not merely autonomous ones. In this section, we focus specifically on the adaptive properties from an HRI perspective.

Adaptation

As discussed above, robot adaptation is essential not only for task-specific learning and behavior design but also for shaping the interaction dynamics between humans and robots [19]. A distinctive feature of adaptation in the context of HRI is its bi-directional nature—meaning adaptive capabilities must be designed with sensitivity to both the context and the specific needs of the user.

For instance, in the case of edutainment robots, the aim is to create systems that require minimal specialized training. These robots are designed to be intuitive and accessible to a broad user base, often relying on simple instructions or built-in guidance to facilitate interaction. This approach reduces user burden and promotes broader accessibility.

Conversely, in high-stakes or complex domains—such as military operations, law enforcement, or healthcare—human operators must undergo extensive training due to increased task complexity and higher risks. In such settings, adaptation must occur in an attended and controlled manner, with additional safeguards in place to mitigate the risks of unintended outcomes resulting from the robot’s autonomous decision-making [20].

2.3.2 Social Agent

Communication

When discussing interactions between AI and humans, communication is an integral component. According to Guzman et al. in *Artificial Intelligence and Communication: A Human Machine Communication Research Agenda* [22], AI as a communicator should be examined from three distinct perspectives—functional, relational, and metaphysical—which may differ from traditional communication theories that focus solely on human subjects.

The functional aspect refers to how people perceive the role of AI as a communicator. In human-to-human communication, interactions are often shaped by individuals’ conceptualizations and evaluations of their communication partners [26]. However, in human-AI interaction, the focus tends to shift from “Who is on the other end?” to “What is the medium or content being exchanged?” In this context, attention centers on the operational role of the AI system as a communicator—specifically, the signals, language, and information it transmits [21].

The relational aspect acknowledges that human communication naturally unfolds within social contexts. As a result, when interacting with AI, people often perceive and respond to it as a social partner. This perception gives rise to the metaphysical dimension of AI communication, which engages with deeper questions about the nature of being, agency, and interaction. As the boundaries between humans and machines become blurred, there is a growing expectation for AI systems to function as ethical agents and conform to social norms—norms that have traditionally been grounded in anthropocentric assumptions [22].

Communication plays a critical role in facilitating effective human-robot interaction. Crandall et al. explore the role of communication in long-term, repeated cooperative scenarios [16]. The study poses the question: “How can we build AI algorithms that cooperate in arbitrarily long-term relationships modeled as Repeated Stochastic Games (RSG)?” To address this, the authors first evaluate several state-of-the-art algorithms designed for RSGs. They identify $S++$ as the top-performing algorithm and enhance it by incorporating a partner signaling system into its expert-selection mechanism, resulting in a new algorithm called $S\#$.

In addition, they evaluate $S\#$ with a feature known as *cheap talk*—a mechanism that allows the agent to send simple, non-binding messages, similar to the informal signals humans use to foster cooperation. Through extensive game-theoretic simulations, the researchers demonstrate

that S not only performs robustly across a wide range of game types but, when equipped with cheap talk, also achieves levels of mutual cooperation comparable to, or even exceeding, those observed in human–human pairings.

Alignment

Making robots align with human intent is another crucial aspect of human-robot interaction. Research on this topic begins with analyzing the elements of human behavior. In 1988, Bratman et al. suggested that a rational individual’s behavior can be explained by three components: **beliefs, desires, and intentions** [11]. This approach has since provided a useful theoretical framework for analyzing and predicting human behavior. Bratman proposed the Belief-Desire-Intention (BDI) architecture, where an agent’s beliefs capture its understanding of the world, its desires represent the goals or outcomes it wishes to achieve, and its intentions—often structured as plans—are the commitments it makes to achieve those goals [11]. Leveraging this framework, if we want robots to be aligned with a human’s next action, the problem converges into understanding the human partner’s underlying intention. This concept is well described by Bertram et al. in *The Distinction between Desire and Intention: A Folk-Conceptual Analysis*, where they state, “the intention, then, is just one step away from the action; all that remains is to put one’s decision into motion” [31]. This argument clearly shows the inferential relation between action and intention.

But how can robots discern what a human’s intentions are? How do they know what the human is thinking? This is the focus of a dedicated area of study known as Theory of Mind (ToM). Theory of Mind is the cognitive ability to attribute mental states—such as beliefs, desires, intentions, and emotions—to oneself and others [37]. This capacity allows an individual to recognize that others have their own perspectives and internal experiences that can differ from one’s own. For humans, ToM develops naturally over time as children learn to perceive complex contexts, interpret information as narratives, and adopt others’ perspectives to reconstruct situations as a whole.

Although ToM is a natural phenomenon for humans, enabling its realization in robots requires mathematical formalization. The next question, then, is: “How can we model ToM mathematically?” Interestingly, to build a robot’s internal model that predicts a human’s intention, we begin by formulating an objective function for observable human behavior given their intentions. This objective function operates on the assumption that if a human’s intention is $\mathbf{x}$, then their action will be $\mathbf{y}$.

According to Baker et al., intention reasoning process starts by first modeling the causal relation between beliefs, goals, and actions as rational probabilistic planning in Markov Decision Process (MDP) [10]. MDP is a mathematical framework for modeling decision-making in situations where outcomes are partially stochastic and partially under the control of a decision maker, such as environment, actions, transition functions, reward functions, etc. As such, in intention reasoning process, variables such as actions, environments, and goal are encoded in MDP. In a nutshell, MDP represents an agent's model of its interaction with its environment [10].

However, to infer an agent's goals from observed behavior, we invert the probabilistic MDP model using Bayesian inverse planning. In forward planning, an agent selects *actions* to maximize expected utility given its *goals*; in inverse planning, we compute the posterior probability of *goals* given observed *actions* via Bayes' theorem:

$$P(G|A) = \frac{P(G) * P(A|G)}{P(A)}, \quad (2.2)$$

where $P(G|A)$ is the posterior probability of a goal G given observed actions A , $P(A|G)$ is the likelihood of actions A if the agent has goal G , $P(G)$ is the prior probability of the goal G and $P(A)$ is the marginal probability of observing A . Then, how do values on the right-hand side given? Under the assumption of bounded rationality, we model the likelihood using a Boltzmann rational decision theory:

$$P(a | G) \propto \exp(\beta U_G(a)), \quad (2.3)$$

where $U_G(a)$ is the utility of action a under goal G , and β , the inverse temperature, controls how strongly the agent favors higher-utility actions. Common utility functions depend on factors such as distance to a target or time required to complete a task, which the agent seeks to minimize or maximize. The priors $P(G)$ and marginals $P(A)$ are set according to domain knowledge or estimated from data. By combining MDP planning with Bayesian inversion, the agent can reason about human intentions from their actions in a principled, probabilistic manner.

Trust

Trust is another primary factor that affects interactions between humans and agents. According to Lee and See, trust is an “attitude that an agent (automation or another person) will help achieve

an individual’s goals in a situation characterized by uncertainty and vulnerability” [28]. Trust is crucial because, for both humans and robots as social agents, the ability to trust one another promotes efficient transactions and cooperation [28]. Numerous social experiments have shown that when trust is appropriately calibrated, it reduces the need for excessive monitoring, negotiation, and coordination among team members, thereby lowering overall social costs [15], [30]. However, blind trust can harm the well-being of a society, as it may lead to over-reliance on the system or each other, potentially causing important warning signals to be ignored. Therefore, calibrating the right degree of trust within society is a significant challenge [35].

During trust formation, many factors come into play. Hancock et al. suggest classifying these factors into three groups—those related to robots, humans, and the environment [23]. The paper provides a detailed classification of factors for each group, with examples drawn from their research [23].

- Human-Related
 - Ability Based: Attention capacity, engagement, expertise, ...
 - Characteristics: Demographics, Personality traits, comfort with robot, ...
- Robot-Related
 - Performance Based: Behavior, level of automation, failure rates, ...
 - Attribute Based: Proximity, co-location, anthropomorphism, ...
- Environmental
 - Team Collaboration: Culture, in-group membership, shared mental models, ...
 - Tasking: Task type, task complexity, multi-tasking environment, ...

The authors then conducted a quantitative meta-analysis to measure the effects of each factor. As a result, the study found that among the three categories, robot-related factors had the strongest impact on trust formation. Specifically, robot performance had the greatest influence, followed by attributes such as proximity, anthropomorphism, and personality. Environmental factors—including team settings, task difficulty, and cultural background—also had a moderate effect, while human factors such as attention level and prior experience with robots had the least influence [23].

Trust between humans and robots significantly affects performance in collaborative tasks. For example, in the paper *Using Trust to Determine User Decision Making & Task Outcome During a Human-Agent Collaborative Task*, Herse et al. examine how user trust impacts decision-making and task outcomes in human-agent collaboration [25]. Using an adapted trust game to indirectly assess human trust in an agent, participants first completed a radiology diagnostics task alone and then with assistance from a virtual robot assistant named Assisto. The study manipulated both the accuracy of the assistant (correct vs. incorrect assistance) and task difficulty (easy vs. hard). As a result, correct assistance led to a significant improvement in task performance especially in the hard task, while incorrect assistance worsened performance. Overall, the study demonstrated that initial behavioral measures of trust have a significant impact on collaborative task performance and can be used to predict and potentially calibrate cooperative performance [25].

Commitment

What is commitment? According to Michael et al., commitment from a philosophical standpoint is defined as a relational construct involving two agents and a specific action. One agent takes on the obligation to carry out the action by giving an assurance to the other, who in turn acknowledges that assurance under conditions of common knowledge [34]. The outcome of such commitment between agents is a joint action toward a shared goal. As long as this commitment state persists, agents remain engaged until the goal is achieved or until all agents agree to interrupt the joint plan they had previously established.

An important consequence of credible commitments is that they make agents' behavior predictable, even in uncertain situations where interests and desires may dynamically change [33]. This predictability facilitates shared planning and coordination among multiple agents [34].

However, commitment is a dynamic and fragile component. The level of commitment can fluctuate during interactions, and any deviation from the shared objective may trigger a cascade of breakdowns in maintaining committed states. Therefore, Michael et al. argue that demonstrating a sense of commitment toward other agents is a crucial element in sustaining commitment [33]. Practically, a robot can show its sense of commitment to humans through various components, as detailed by Michael et al., [33].

- Reputation Management: Robots could be designed to build and maintain a reputation for reliability, which in turn builds human trust and commitment.

- Emotional Cues: Incorporating functional equivalents of emotions (like guilt or pride) in robots can help motivate them to honor commitments and signal commitment to humans.
- Eye Contact: Establishing eye contact (or analogous cues) may implicitly signal cooperative intent and commitment, as it does in human interactions.
- Signaling: Cooperative body language and predictable movement patterns serve as nonverbal signals that help maintain a mutual sense of commitment.
- Coordination and Mutual Supportiveness: Successful coordination and mutual support is an outcome of commitment but they could also feedback to contribute forming solid sense of commitment. The author argues fostering coordination and mutual support can further stabilize commitment in interactions.

2.4 Game Theory

According to Osborne et al., game theory is a collection of analytical tools designed to help us interpret the dynamics that arise when decision makers interact. It is based on two fundamental assumptions: first, that decision makers are rational, and second, that they reason strategically about others' actions based on their own knowledge and expectations [36].

Game theory has been commonly research tools in behavioural economics and psychology, allowing researchers to utilize its experimental games as a mean of providing an implied but *quantifiable* measure of the underlying state of social interactions such as trust and cooperation. [18]. Games are devised to abstractly represent certain critical features of real-life situations [18], where the outcomes that occur are determined by the choices of the players [14], [25].

Games and Solutions

Real-life situations are highly complex. However, in a game theoretic approach the situations could be boiled down to key abstract representations that governs the critical interaction mechanisms.

A game is a description of a strategic situation. It includes information about the players, the constraints on the actions that the players can take, and how much each player can make, their *payoffs*. However, it not specify the actions that the players will actually take [36].

A solution is a systematic description of the outcomes that may emerge in a family of games, given that the players are rational and know how to play the game. The most commonly used solution concept in game theory is the Nash equilibrium. The existence of such a solution makes the interaction outcomes quantifiable and amenable to analysis and interpretation [36].

Then what is Nash equilibrium? According to Spaniel, *Game Theory 101: The Complete Textbook*, “A Nash equilibrium is a set of strategies, one for each player, such that no player has incentive to change his or her strategy given what the other players are doing [43].” In simpler terms, a Nash equilibrium is a state where no player can improve their payoff by unilaterally changing their strategy—so no one has an incentive to deviate, resulting in an *equilibrium* of decision-making among the players. Importantly, a Nash equilibrium is stable but not necessarily optimal: it ensures that no individual can do better on their own, yet the collective outcome may still fall short of the best possible and, in some cases, be quite poor.

Cooperative Games

Because the Prisoner’s Dilemma is the most famous example, one might assume that all games are adversarial. However, cooperative games also exist. Non-cooperative and cooperative games can be distinguished by their action primitives: in non-cooperative games, each player’s actions are taken individually, whereas in cooperative games, also called as coalitional games, action primitives involve joint actions by groups of players [36]. Analogous to the Nash equilibrium in non-cooperative games, the *Core* serves as a solution concept in cooperative games. A solution lies in the core when no set of players can break away and achieve a joint payoff that is better, resulting an outcome stable [36].

Along with the concept of the core, *Sharpey value* is a solution that describes how participants divide the value fairly, in other words, how the value $v(N)$ should be divided among the player N [13]. Sharpey value is consisted of four axioms [13]:

1. Efficiency: The entire value of the grand coalition is fully distributed.
2. Symmetry: Players who contribute equally receive the same payoff.
3. Dummy player: Players who make no contribution should receive nothing.
4. Additivity: The value should be additive over the set of all games.

Overall, combining the Core and the Shapley value as solution concepts provides an analytical framework for understanding which cooperative agreements are stable and how the cooperative surplus is fairly distributed.

2.5 Summary

This background section provides an integrated overview of the key domains that underpin our exploration of Cooperative AI. It establishes how cooperation is not only a biological and cultural *phenomenon*, but also a complex *design challenge* in the development of intelligent systems. By interrelating findings from Cooperative AI, Robot Learning, Human-Robot Interaction, and Game Theory, we construct a conceptual territory map that grounds the foundation of this research.

Cooperative AI (CAI): Cooperative AI focuses on enabling agents to work together—whether with one another or with humans—by fostering task coordination, mutual understanding, and commitment to shared goals. Rooted in the evolutionary foundations of cooperation from both biological and cultural perspectives, this field redefines intelligence as inherently social. The concept of Cooperative Intelligence incorporates key elements such as understanding, communication, commitment, and adherence to social norms, emphasizing that effective cooperation requires more than just autonomous behavior.

Robot Learning (RoL): RoL emphasizes the importance of embodiment and real-time adaptation in AI agents. Unlike traditional machine learning, RoL requires agents to actively generate and refine their own training data through direct interaction with dynamic environments. This unique approach is particularly evident in grounded language processing, where robots map linguistic inputs to physical actions and perceptions, making them better suited for complex, real-world tasks. Overall, RoL offers essential technical methodologies for developing cooperative AI agents.

Human-Robot Interaction (HRI): HRI provides a theoretical and empirical framework for structuring interactions between humans and robots. It addresses key research topics such as alignment, trust building, and communication in human-machine interaction. By approaching these issues through both computational modeling and empirical studies, HRI research emphasizes the importance of incorporating human factors—particularly when designing social robots.

Game Theory: Game Theory provides a set of analytical tools to quantify and examine cooperative behavior within structured experimental settings. Through game-theoretic frameworks—such as the Repeated Prisoner’s Dilemma and adapted trust games—researchers can in-

investigate underlying social factors like trust, commitment, and coordination. These models offer a structured way to interpret real-world interactions in a quantifiable manner, yielding insights into how cooperative dynamics emerge, evolve, and can be systematically assessed.

Together, these four perspectives form a holistic framework for understanding and advancing research in Cooperative AI, forming the foundational basis for the work presented in this thesis.

3

Method

Contents

3.1 Overview	32
3.2 Human Machine Guitar Hero	33
3.2.1 Design	34
3.2.2 Expanding Human-Machine Guitar Hero	40
3.3 Designing Co-op Tile World	41
3.3.1 Supporting Co-op Game Play	42
3.3.2 Implementing Game Mechanics	45
3.3.3 Designing Two Levels	46
3.3.4 Communication Protocols	48
3.3.5 Developing Agents	49
3.4 User Experiment	62
3.4.1 Experiment Design	62
3.4.2 Experiment Evaluation Criteria	64
3.5 Data Collection	66

Citations in this chapter: [2], [5], [7], [9], [12], [32], [40], [44], [45].

Acronyms in this chapter: Behavior Cloning (BC), CAI, HRI, Imitation Learning (IL), Multi Agent Reinforcement Learning (MARL), Operation Time (OPTIME), RL

3.1 Overview

Up to this point, we have discussed the context of this research, addressed the importance of cooperation with machines in today’s landscape, and explained the need to develop socially intelligent agents capable of engaging in real-time interaction with humans. This section outlines how the research addresses two central research questions:

- **RQ1 (Design):** How should we **design** a benchmarking game environment for cooperation between humans and socially intelligent agents?
- **RQ2 (Assessment):** How can we **assess** cooperative performance between humans and socially intelligent agents using this custom environment?

To begin answering these questions, I introduce a preliminary research project titled **Human-Machine Guitar Hero**, which laid the groundwork for this thesis and offered key insights that informed the design and development of the main study.

The next step is the main research project of this thesis, **Co-op Tile World**. Co-op Tile World is a custom benchmarking environment specifically designed to evaluate cooperative performance between real humans and machine agents during real-time gameplay. The development of Co-op Tile World is divided into two major parts: designing the environment itself and developing a machine agent capable of operating within it.

Finally, we address the second research question concerning assessment. Using the environment developed in the previous stage, we define quantitative metrics to measure cooperative performance. These metrics are based on the rewards received by players at the end of each game and are interpreted through the lens of the **Stag-Hunt** game theory model. Real human participants interact with the machine agent, and their cooperative behaviors are recorded. The results are then analyzed through both descriptive and statistical methods, yielding findings that support, verify, and extend insights from existing research in Cooperative AI (CAI) and Human-Robot Interaction (HRI).

3.2 Human Machine Guitar Hero

Human-Machine Guitar Hero is a game in which a human and a machine agent must strategically cooperate to achieve the highest possible score. A key feature of the game is that the machine agent continuously observes the human player's performance and selects an optimal mode of cooperation based on the *Play Style Decision Quadrant*, with each quadrant labeled as Master, Strategic, Novice, and Solo. Based on this classification, the machine determines its behavior using a predefined actions table—a set of algorithms that guide its cooperative responses. The game also includes a real-time *communication system* that allows both players to share strategies during gameplay.

One of the most intriguing aspects of the cooperative interaction observed in this research is that the machine’s actions were firmly grounded in a **shared objective**, rather than adopting a *supportive* role aimed at *assisting* the human player’s learning—a behavior more commonly expected from machines. Occasionally, the machine’s strictly goal-driven actions felt like interruptions to some players. However, these moments reflected the natural dynamics of cooperative play, mirroring the negotiation, misalignment, and adaptation that often arise in human-to-human cooperation.

3.2.1 Design

The aim of the game, Human-Machine Guitar Hero, is to implement a **Play with** cooperative model between humans and machines by locating them in a gaming environment. This game is built on the same mechanics as the well-known Guitar Hero video game. However, the extraordinary touch of Human-Machine Guitar Hero is that there is a machine agent with which the human player should cooperate. Throughout the game, human and machine players will strive together to achieve higher scores.

Specifically, players have two action modes:

Players individually strive to play the game accurately to earn base points by pressing the button at the precise moment when notes are falling. On top of this, players work cooperatively to earn bonus points by managing their contributions to the total score equally throughout the game.

Initially, their cooperation may not be effective, but after multiple iterations of the game, humans and machines will learn how to cooperate effectively. This is accomplished as humans acknowledge the machine as a reliable teammate, while the machine builds a mental model of the human player’s play style.

Agent’s Feedback Model

Let’s get into how the cooperative guitar hero machine player is developed. The below diagram shows the agent’s learning and cooperation model.

First, the game involves two players: a human and a machine. Both players interact with the environment by performing actions and receiving rewards based on the outcomes of those actions.

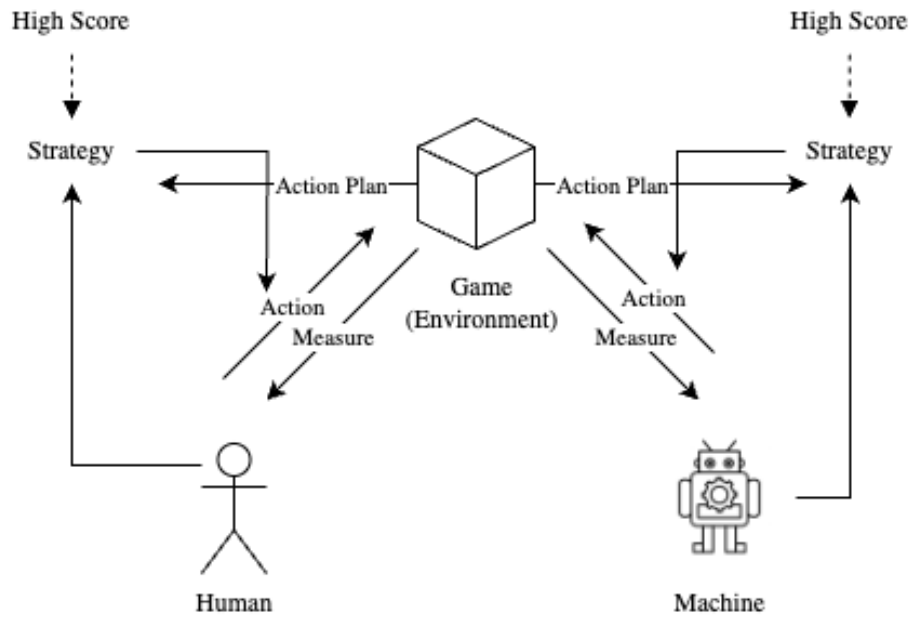


Figure 3.1: Human-Machine Guitar Hero feedback model

Their interaction is guided by the game’s objective, which is to achieve a high score. This naturally introduces the need for strategic decision-making, and because success depends on cooperation, the players are required to **share** their strategies. These strategic plans are communicated through the game display.

The agent’s cooperation model in this game can be analyzed through the lens of cybernetic system classifications: First-order systems, Second-order systems, and Conversing systems—the latter being a combination of the first two. According to Dubberly et al., a first-order system is a simple, self-regulating system with a single feedback loop. It cannot adjust its own goal; instead, its goal must be set or modified by something external to the system. A classic example of a first-order system is a thermostat, which adjusts temperature in response to the external environment but cannot change its target temperature on its own [44].

An interesting phenomenon in cybernetics is that a second-order system can emerge by stacking two first-order systems. Second-order systems extend the capabilities of first-order systems by adding an additional layer. This added layer observes the effects of the original system on the environment. Based on these observations, it can adjust the system’s objectives, either aligning with the initial purpose or choosing to diverge from it [44].

This structure is analogous to meta-planning, where an agent not only executes a plan but also evaluates and modifies the plan itself. Crucially, this is where true learning occurs—shifting from merely **reacting** to the environment based on fixed goals to **adapting** those goals based on

experience. In this way, the mental and behavioral model of a game agent can be framed in terms of first-order and second-order system dynamics.

First-order system: In this model, the human and machine agents independently perceive the game environment and respond accordingly. Each agent initiates actions by reacting to environmental stimuli.

Second-order system: In this model, the agents engage in a higher level of cooperation by formulating a shared strategy or a **next action plan** aimed at achieving a higher score. They evaluate the outcomes of their actions and adapt their strategies accordingly.

An even more advanced model can be designed by parallelizing two second-order systems. This gives rise to a Conversing system, where the output of one learning system becomes the input of another. As Dubberly et al. describe, “Furthermore, the systems learn from each other, not just by discovering which actions can maintain their goals under specific circumstances (as with a standalone second-order system), but by exchanging information of common interest. They may coordinate goals and actions [44].”

This conversing system plays a central role in the gameplay and supports the core objective of this prototype. It is embedded directly into the game mechanics. Within this system, the human and machine provide feedback to one another about the next action plans. Through this mutual exchange, each player influences the other’s next actions in order to maximize their score.

Game Rules

So, what are the rules of the game? The game has a single winning condition: earn as many rewards as possible. However, achieving this goal requires more than just playing well.

First, players must perform well *individually* by playing their own guitar parts accurately. But that alone is not enough. Second, players must also perform well *collectively*, cooperating with their partner to earn bonus rewards. To maximize these bonus rewards, each player’s contribution to the total score must remain balanced—ideally a 50/50 split.

Game Scenarios

Figure 3.2 illustrates the example gameplay scenario:

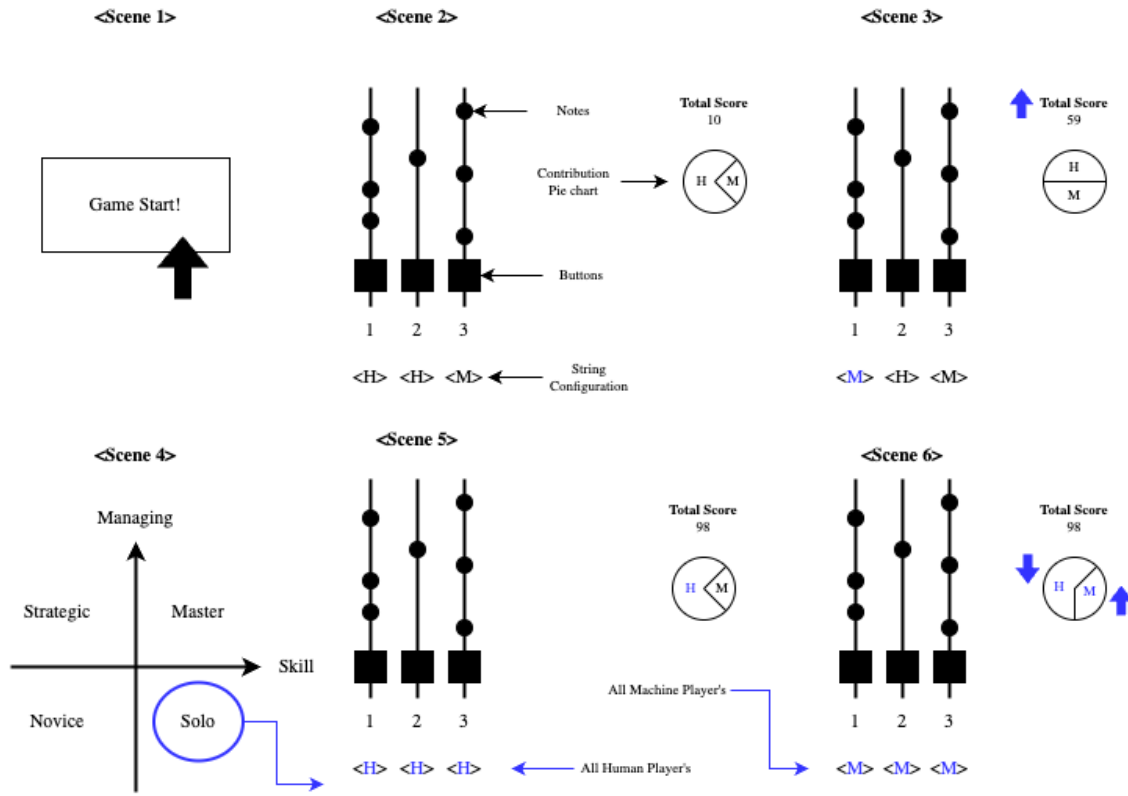


Figure 3.2: Game scenarios

<Scene 1> The human player (HP) initiates the game by clicking the “Game Start” button on the screen.

<Scene 2> At the start, HP is assigned to control strings 1 and 2. As this is their first attempt, HP frequently misses notes and struggles to maintain the desired 50/50 contribution balance.

<Scene 3> Recognizing HP’s difficulties, the machine player (MP) classifies HP as a *Novice* and intervenes to support engagement. MP takes over string 1, allowing HP to focus solely on string 2. This minimizes score loss while preserving the minimum cooperation threshold.

<Scene 4> After several rounds, HP gradually improves and begins to hit notes with more accurate timing. Observing this progress, MP updates HP’s classification from **Novice** to **Solo**, demonstrating trust in HP’s increased skill level.

<Scene 5> With growing confidence in HP’s abilities, MP allows HP to handle even the most complex note patterns, giving them greater responsibility.

<Scene 6> Despite this progress, MP remains attentive to contribution balance. Given HP’s

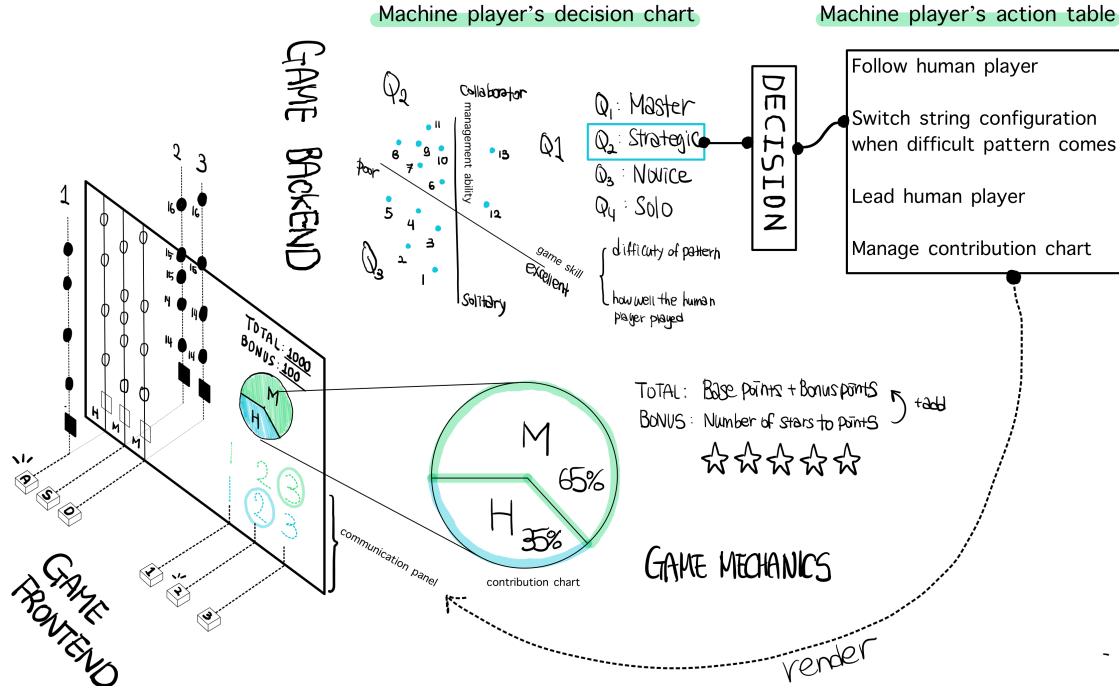


Figure 3.3: Frontend and backend game structure diagram

limited experience in managing dynamic gameplay, MP occasionally intervenes to reassign string responsibilities when cooperation levels drop below the target threshold.

Over time, HP learns to anticipate MP's actions and adjusts their play accordingly. This evolving coordination results in more effective collaboration and higher overall scores.

Modeling Cooperative Machine

Based on this setup, the most interesting part is left: “How should the gameplay be designed in terms of the machine player?” To model a machine player's play, we should answer the following three questions. First, **what to learn?** Second, **how to model decisions?** Third, **how to model actions?**

1. **What to learn?** In an ideal scenario, the machine agent would also learn and adapt to the game environment over time. However, in this prototype, the machine's knowledge of the environment is pre-programmed—in other words, it functions as a perfect player from the outset. Instead, the focus of this project is on the second aspect: learning about its human counterpart. The machine player closely monitors the human player's every in-game action, using this information to determine the human's playstyle.

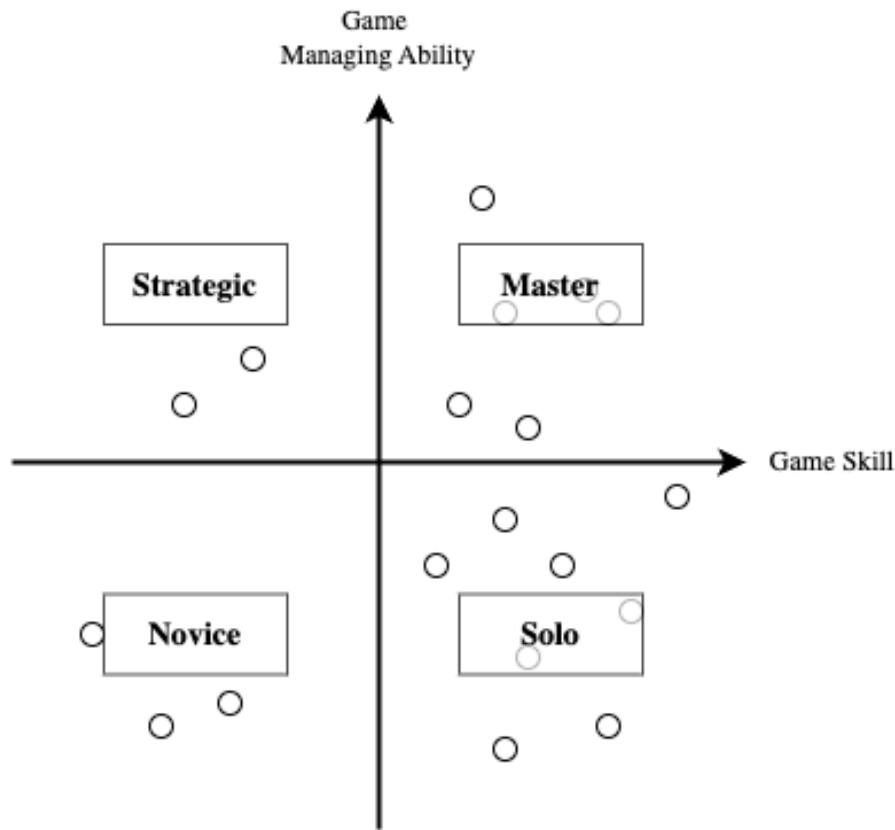


Figure 3.4: Playstyle decision quadrant

2. How to model decisions?

So, how can we model such decisions? Refer to Figure 3.4. The agent uses two numerical values to assess the human player. The skill level reflects the player's accuracy in hitting notes at the correct timing, while the managing level represents the player's ability to maintain a balanced contribution between the human and machine players.

Using these two dimensions, the machine maps the player's behavior onto a playstyle quadrant. The x-axis corresponds to the skill level, and the y-axis represents the managing level. Based on the human player's gameplay data, the machine plots the data on the quadrant and classifies human players into one of four playstyles: Master, Strategic, Novice, or Solo.

3. How to model actions?

One last piece remains. how to model actions? Based on the previous decision, the machine player will act based on its action table, as shown in Table 3.1. For example, if the machine player decides the human player is a Novice, then it will change the string configuration in a way that human play is minimized. On the other hand, when the player is a Solo, then the machine will let the human player be engaged actively ensuring the contribution level is

Machine’s Decision	Machine’s Behavior
A: Master	Follows the human player’s action plan.
B: Strategic	Minimize human play when difficult beat patterns are coming up. Otherwise, follow the human player’s management decision.
C: Novice	Leads the play session, by minimizing human player’s play.
D: Solo	Suggest string configuration that maximizes human player’s play.

Table 3.1: Machine’s Action Table

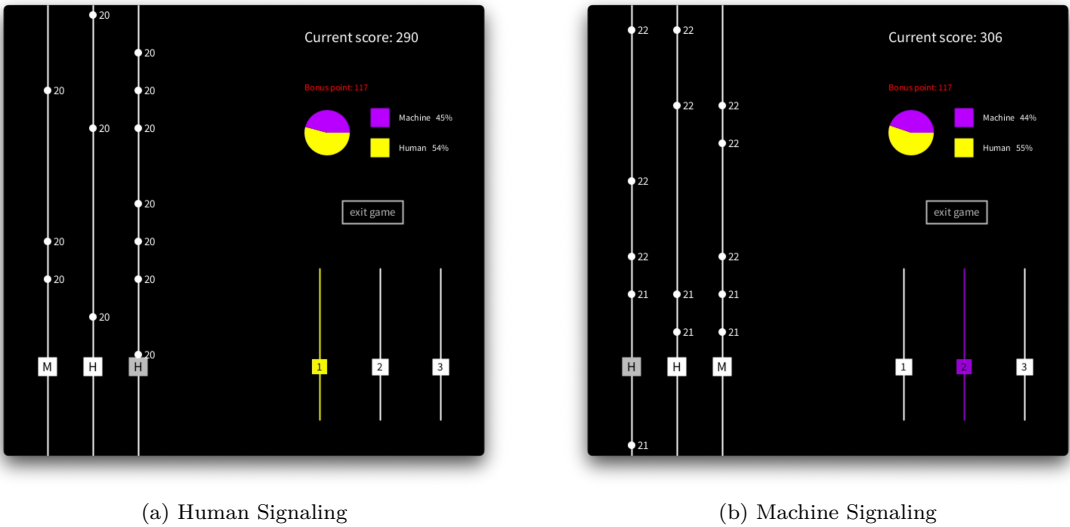


Figure 3.5: Communication protocol between human and machine players

maintained. When the player is a Master, the machine will follow its counterpart’s action plan.

Supporting Conversing System

As the final component, the game must provide a means for communication between the human and machine players. This prototype supports an intuitive conversing system. In Figure 3.5a, the human player presses key 1 to indicate that they will take responsibility for string number 1. Figure 3.5b illustrates a scenario where the machine player signals that it will take over string number 2 from the human player. In both cases, the selected string lights up with a color corresponding to the player, visually reinforcing the communication.

3.2.2 Expanding Human-Machine Guitar Hero

From this project, I could derive two meaningful approaches that should be continued.

1. The game should support real-time interaction between human and machine players, with the machine adapting dynamically to its human partner.
2. The agent should support a communication system that enables both players to share their strategies in order to achieve a common goal.

Building on the preliminary project Human-Machine Guitar Hero, I introduce Co-op Tile World—an expanded environment that increases task complexity by transitioning from the one-dimensional space of Human-Machine Guitar Hero to a two-dimensional grid world. This environment provides researchers with greater flexibility to design custom cooperative tasks by incorporating diverse tile mechanics. It also showcases the development of agents capable of tackling more complex challenges using various learning and path-planning algorithms. Finally, the project presents a framework for assessing human-machine cooperative performance, enabling researchers to conduct [HRI](#) and [CAI](#) studies that generate insights grounded in real-time cooperation between humans and machines.

3.3 Designing Co-op Tile World

Co-op Tile World is a customized game environment designed to assess the cooperative performance of humans and agents. While there are several pre-built environments for benchmarking cooperative behavior [\[2\]](#), [\[5\]](#), [\[9\]](#), these platforms are not specifically designed to support real-time interaction between humans and AI agents. To address this gap, this research introduces a new benchmarking environment dedicated to testing cooperative interactions between humans and machines. Through this foundational effort to actively involve real human interaction in the development of cooperative agents, the thesis aims to advance [CAI](#) and [HRI](#) research in a more human-in-the-loop direction.

Co-op Tile World is built on top of the open-source game *Tile World 2*, originally developed by Brian Raiter, Madhav Shanbag, and Eric Schmidt [\[7\]](#). Based on this foundation, Co-op Tile World was redesigned as a two-player cooperative game. Using *Tile World 2* as the code base offers a significant advantage: its robust and well-developed game mechanics allowed me to focus on designing cooperative interactions rather than building the entire system from scratch. Furthermore, the code base is actively maintained by multiple contributors, ensuring ease of maintenance and long-term compatibility for Co-op Tile World. In addition, the original game is relatively well-known

and not overly complex in operation. This familiarity and simplicity help reduce the cognitive load for participants, making it an ideal platform for studying human-machine cooperation.

Design Process of Co-op Tile World

Developing the Co-op Tile World required the following seven steps:

1. Adapt existing Tile World 2 into a two-player cooperative game.
2. Implement core game mechanics, including navigation, item interactions, and win/loss conditions.
3. Design levels using the Co-op Tile World level editor tool.
4. Design a communication system and implement an in-game communication protocol.
5. Develop AI agents capable of playing each level.
6. Build a strategic planning model for each agent.
7. Deploy the agent in the environment and evaluate the cooperative performance between the AI agent and human participants.

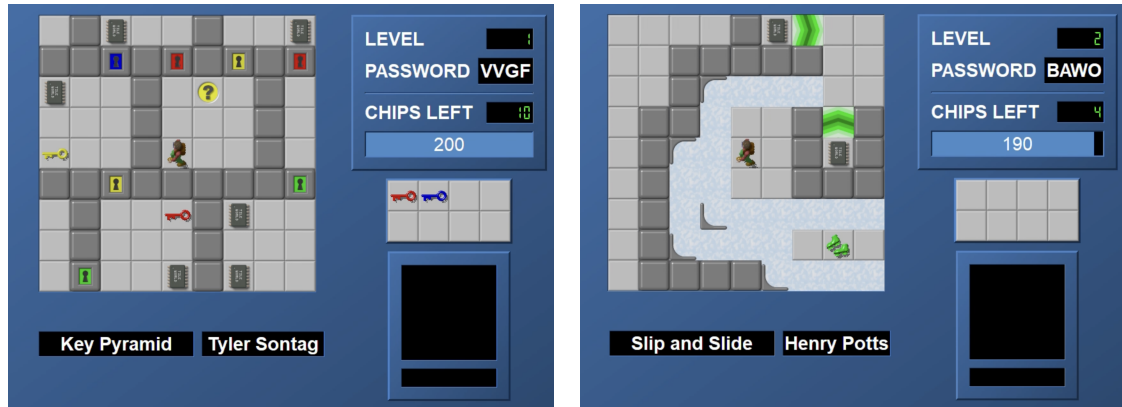
In the following section, each step will be discussed in detail, covering both the implementation process and the outcomes. The goal is to provide sufficient detail to enable replication of this approach by other researchers.

3.3.1 Supporting Co-op Game Play

Before diving into the cooperative gameplay rules, let us first take a brief look at the overall game design of the original Tile World 2.

Tile World is built on a 32×32 tile grid, populated with a variety of tiles and objects. These include hazardous tiles such as fire, water, ice, and force floors, as well as collectible items like (computer) chips, keys, and boots. Each tile serves a specific function—for example, fire and water tiles cause the player to lose a life, while ice and force floors force the player to move in a fixed direction, making it more difficult to control movement.

Collectible items play an essential role in gameplay. Keys are used to unlock doors of matching colors, granting access to new areas. Boots allow players to safely walk over specific hazard tiles.



(a) A screenshot of Lesson 1

(b) A screenshot of Lesson 2

Figure 3.6: Screenshots from original Tile World 2

The primary objective of this game is to collect all the chips scattered throughout the level in order to unlock a socket and reach the exit tile. Along the way, players must solve puzzles and navigate obstacles that require careful collection and use of these items.

Although the core mechanics are simple and intuitive, the combination of diverse tile types allows designers to create levels ranging from easy to highly challenging. In the original Tile World 2, each level is referred to as a “lesson,” and players must complete it within a set time limit to succeed.

Based on the original game mechanics, Co-op Tile World was transformed into a two-player cooperative game. The fundamental rules of this conversion are as follows:

- Each player maintains their own item inventory.
- Only the player who possesses an item in their inventory can interact with the corresponding tile.
- A more sophisticated time rule has been implemented to support richer cooperative scenarios and enhance gameplay dynamics:
 - The level must be completed within a given time limit. If neither player reaches the goal tile in time, the level fails.
 - If one player reaches the exit tile, a three-seconds count down begins. During this time, the other player must also reach the goal tile to receive the reward.
 - If only one player reaches the exit within the time limit, only that player receives the reward.

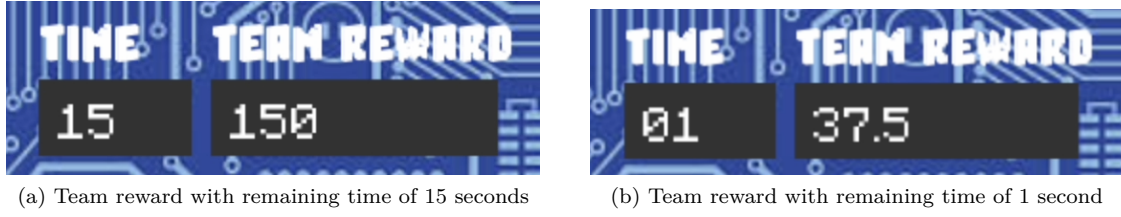


Figure 3.7: Differing team reward based on remaining time

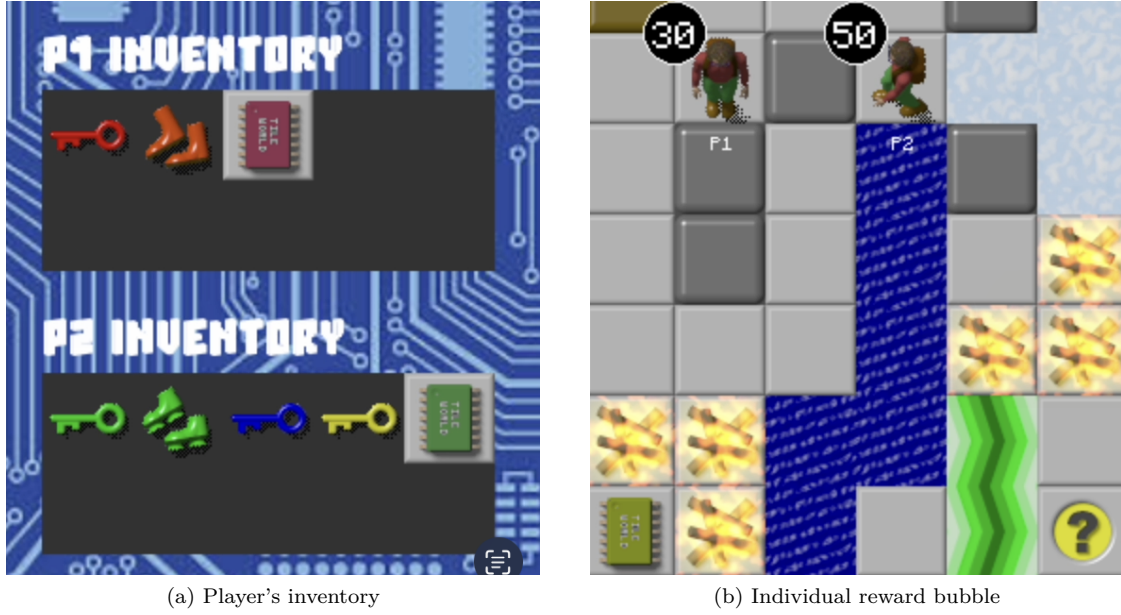


Figure 3.8: Player inventory and individual reward bubble

- If both players successfully reach the exit tile within the time limit and three-seconds countdown, they receive the highest possible reward.
- An advanced reward mechanism has been implemented to monitor cooperative play:
 - The team reward is calculated based on the time remaining when the level is completed. For example, as illustrated in Figure 3.7, the two plots show that team rewards are inversely proportional to the time left, encouraging players to cooperate and complete the level as efficiently as possible.
 - Individual rewards are determined by the number of collectible items each player obtains during gameplay. As shown in Figure 3.8, a black bubble follows each player, visually representing the individual reward accumulated so far.

These new mechanics introduce cooperative gameplay elements, such as the ability to distribute and assign items strategically to optimize task completion. For example, players can coordinate

their actions by assigning specific items to the player or by having one player wait at the exit tile to ensure a faster level completion.

3.3.2 Implementing Game Mechanics

The initial idea behind using Tile World 2 as a code base was to reuse its existing game mechanics. However, this approach proved to be suboptimal because the original code was written in C++. Given that most Machine Learning and Reinforcement Learning (RL) libraries—such as Gym, PettingZoo, and PyTorch—are written in Python, it became evident that implementing the game environment in Python would be the best practice for future agent training.

As a result, instead of directly porting the entire Tile World 2 codebase, this research focused on implementing a confined set of game mechanics that are essential to the Co-op Tile World environment. The following table outlines the game mechanics that are commonly or exclusively used in Level 1 and Level 2.

Mechanics	LV1	LV2
Moving	v	v
Burning Fire	v	v
Drowning Water	v	v
Slippery Ice		v
Force Floor		v
Chip Collection	v	v
Socket Unlocking	v	v
Monster Collision		

Mechanics	LV1	LV2
Key Collection		v
Door Unlocking		v
Boot Collection		v
Boot Effect		v
Movable Dirt Block		v
Dirt Tile Effect		v
Time Attack	v	v
Communication Protocol	v	v

Table 3.2: Game mechanics used in each Level

The variation in applied game mechanics across levels is designed to create a clear difference in task difficulty for experimental purposes. Although collision with monsters is supported in the current version of the game, agents are not programmed to interact with them. During the development and testing of AI agents, avoiding collisions with moving monsters proved to be significantly more challenging than anticipated. After multiple rounds of trial and error, this research chose to focus on static tile effects as the primary mechanism for shaping agent behavior and task complexity.

As shown in Figure 3.9, the current version of Co-op Tile World supports game mechanics of collecting 9 collectible items, unlocking 5 tiles, interacting with 7 interactable tiles, collision detection with 1 monster type (beetle), and reaching the exit tile.

Collectible Items														
Unlockable Tiles														
Interactable Tiles														
Monsters														
Exit Tile														

Figure 3.9: Supported tile mechanics for Co-op Tile World

3.3.3 Designing Two Levels

Co-op Tile World includes a valuable tool called the Co-op Tile World Level Editor, originally developed as part of an assignment for the course *Programming for Game Designers*. The starter codebase for the assignment was developed by instructors David Culyba and Phil Light.

Using this tool, designers can easily create custom levels using Google Sheets. Once all tiles are placed on the spreadsheet, a custom Google Apps Script is used to convert the level data into a JSON file format. This approach offers an efficient workflow for prototyping, testing, and validating different level designs before finalizing them for experimental use.

The figure 3.10 illustrates the two levels designed in Google Sheets and the figure 3.11 shows corresponding JSON file generated by executing the `Save all levels` script within the sheet. By simply loading this JSON file into the game, the designed level can be rendered and played directly in Co-op Tile World.

Using this *Co-op Tile World Level Editor*, I created two levels with the primary objective of ensuring a clear distinction in task difficulty, allowing participants to perceive the difference immediately.

As shown in Figure 3.12, Level 1 is relatively simple, requiring players to collect chips while avoiding water and fire tiles. In contrast, Level 2 introduces significantly greater complexity. Players must navigate hazardous tiles, understand tile dependencies, and know how to use movable tiles and their interactions with other elements. Level 2 is intentionally designed to allow for multiple solutions, encouraging players to communicate and strategize in order to succeed.



Figure 3.12: Two levels demonstrating different task difficulties

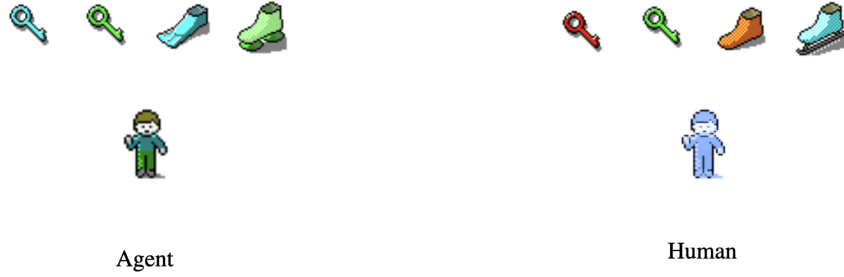


Figure 3.13: Item assignment as strategic decision making

3.3.4 Communication Protocols

The next crucial feature the benchmarking environment should support is a communication protocol between human players and agents. Among diverse possible implementations of this system, here, we decided to leverage the core game mechanics of the game, which is to collect items to win. Based on this mechanics, players share their strategic planning by assigning collectible items to their partners or to themselves [3.13]. This assignment mechanism serves as an implicit communication channel for strategy exchange.

Based on this communication logic, it was essential to establish a clear way to visualize strategic exchanges between players. This is where the **communication protocol UI** comes into play. The UI not only enables players to interact efficiently but also allows experiment participants to visually inspect how decision-making processes evolve.

Figure 3.14 provides a detailed screenshot of the communication protocol UI for each level. At the top of the interface, all collectible items on the current map are displayed. Players can *assign* an item to themselves or to their partner by right-clicking on its icon. Once an item is assigned, it appears in a separate box, showing which items have been allocated to each player. Additionally, players can right-click on items within their own inventory box to *unassign* them if needed.

To accommodate the *Time Attack* feature of the game, the game clock pauses while the players engage in strategic decision making. Specifically, players can press the ***Operation Time (OPTIME)*** button on the right side of the UI, which temporarily stops the game and allows time for communication.

Refer to Figure A.1 for a detailed explanation of how this communication game mechanic functions. The figures illustrate the sequential process of communication.

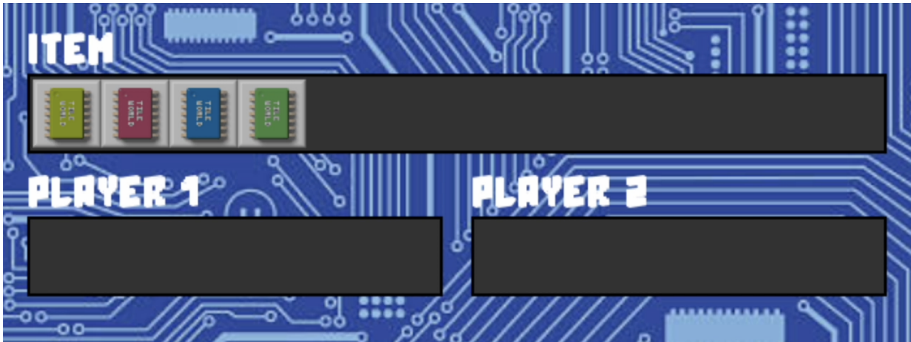
3.3.5 Developing Agents

Figure 3.15 shows the high-level architecture of the agent. The agent is structured into four layers. The lowest layer is the operation layer, where the agent play the game on its own. Above that is the cooperation layer, where the previously solo-playing agent learns to collaborate with another agent. The communication layer follows, integrating the environment’s predefined communication protocol. Finally, at the top, the alignment layer defines three different degrees of the agent’s alignment in decision-making with human partners.

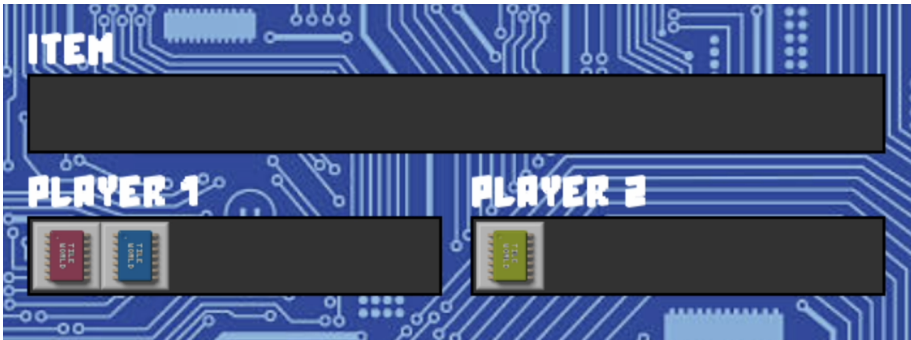
Table 3.3 shows how each layer is implemented. The original intention was to fully support a **RL** agent at Level 2 as well. However, ultimately, the agent was implemented by using A* path planning algorithm to ensure reliable agent performance at this level. The sequential design of Level 2 – collect keys, unlock associated doors, gain items and chips – resulted in sparse rewards, making it difficult for **RL** agents to successfully learn the optimal policy.

Layer	Level 1	Level 2
3	Given alignment mode – aligned, merged, diverged models	
2	Proximity-based planning	Proximity and tile-dependency-based planning
1	Additional BC training with surrogate partner	Recalculate A* path every game frame
0	Behavioral Cloning model	A* path-planning algorithm

Table 3.3: Layered agent design



(a) Level 1 collectible items list



(b) Level 1 strategic assignment made



(c) Level 2 collectible items list



(d) Level 2 strategic assignment made

Figure 3.14: Communication protocol UI

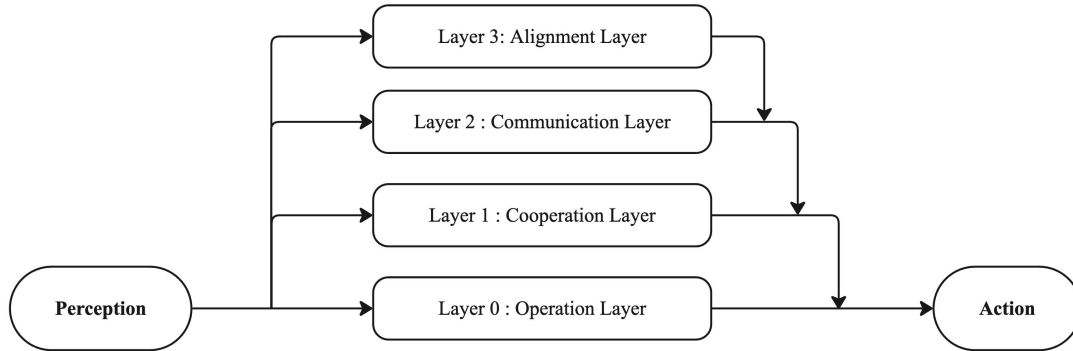


Figure 3.15: Layered structure of constructing the agent

Layer 0: Operation Layers

BC is a subset of Imitation Learning **IL** and takes the form of supervised learning. In Behavior Cloning, we train an agent to imitate an expert’s behavior. To do so, we collect a dataset of state-action pairs from a human playing the task and use this data to train a neural network. The network is trained to predict the output action based on the input state. After training, the game loop provides the agent with a game state—structured identically to the training data—and the agent outputs actions.

The **BC** method has both advantages and limitations. Since data is directly provided by a human expert, the agent can achieve reasonably good performance with a very short training time compared to other unsupervised approaches such as **RL**. However, this strength also becomes a weakness: the agent is extremely fragile to small deviations from the expert’s data. Because the expert’s data always contain only *correct answers*, the agent fails dramatically with unseen situations. Additionally, noise in the data set may cause agents’ undesirable actions.

Therefore, there are various techniques developed to overcome the limitations of **BC**. One such method is **Dagger** (Dataset Aggregation). **Dagger** begins by training the agent on an initial dataset of expert demonstrations, similar to **BC**. The agent is then deployed to act in the environment. During its execution, if the agent deviates from optimal behavior, the expert intervenes and provides corrective actions. These new state-action pairs are aggregated with the original dataset. Over successive iterations, the agent is then trained not only on ideal expert states, but also on the actual situations it encounters when performing alone, giving an agent more robust performance.

As an alternative approach to developing an agent at the operation layer, **RL** was also explored. However, fully supporting an **RL** agent proved more challenging than anticipated due to limited computational resources and the design of the levels themselves. The sequential nature of the levels led to sparse rewards, making it difficult for **RL** agents to effectively learn a policy. This required a meticulously designed reward mechanism and a large number of training episodes. Details of the **RL** training experiments can be found in Appendix Section B.

Level 1

For Level 1, the agent was initially trained using the **BC** (Behavior Cloning) method. After implementing all game mechanics, gameplay data from a human player was recorded. The state vector input size is 191, constructed as a concatenated set of tuples: `position`, `chips collected`, `total chips collected`, `socket unlocked`, `nearest chip`, `exit location`, `full grid tensor`, `local grid tensor`, `alive`, `remaining chips`, `other player position`. Each state was paired with one of four discrete actions: `up`, `down`, `left`, `right`.

This state vector was then fed into a neural network. The network for Level 1 consists of a fully connected feedforward architecture with five hidden layers of size 128, each followed by a ReLU activation. The input layer has size 191, and the final output layer is a linear projection to four outputs, corresponding to the action space.

However, this initial **BC** agent performed poorly, as the training data did not account for the partner’s dynamic position changes. To address this, additional gameplay data was collected by deploying the initially trained agent into the game environment. As a result, the newly collected data incorporated dynamic positions of the other player which simulates the movement of human player that it will actually work with. This additional **BC** training allowed agent to learn state distributions of human partner’s movements.

Level 2

To develop an agent capable of playing Level 2, I researched path planning algorithms—specifically Dijkstra’s algorithm and A* (A-star)—as they are well-suited for the grid-based environment of Co-op Tile World.

Dijkstra’s algorithm is a graph traversal method that, given a weighted graph, finds the shortest path from a starting node to all other vertices. It operates using a greedy approach, incrementally building the optimal solution by always selecting the vertex with the currently known shortest distance from the start. The algorithm then updates the distances to neighboring vertices by

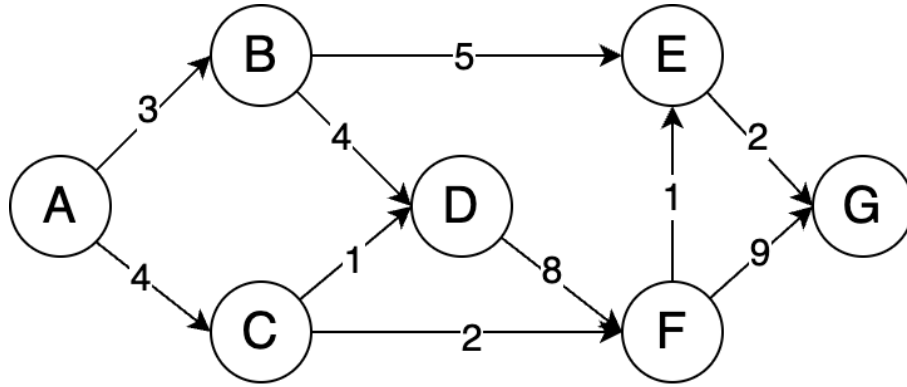


Figure 3.16: A weighted graph where each edge representing cost

adding edge costs where *applicable*, leaving unreachable nodes with an infinite distance value.

Here, *applicable* means that starting from vertex u , there exists a directed edge to vertex v , and the current distance to u plus the cost from u to v is less than the current known distance to v .

```

if (d[u] + c(u,v) < d[v]):
    d[v] = d[u] + c(u,v) // update distance
else:
    pass // no update

```

Given the example directed graph as [3.16](#) where edge weights represent the costs from one node to another, and assuming that the start node is A, the distance to the start node is initialized to 0 while all other nodes are initialized to `inf` (infinity). The table below illustrates the step-by-step process of how distances are updated using Dijkstra's algorithm:

Explored \ Unexplored	B	C	D	E	F	G
A	3	4	∞	∞	∞	∞
B	3	4	7	8	∞	∞
C	3	4	5	8	6	∞
D	3	4	5	8	6	∞
F	3	4	5	7	6	15
E	3	4	5	7	6	9

Table 3.4: Progress of Dijkstra's algorithm showing tentative distances from start node A

Building on Dijkstra's algorithm, the A* algorithm introduces a heuristic function to optimize pathfinding by guiding the search toward the goal. The choice of heuristic depends on the design and objectives of the specific problem. A* minimizes the sum of the actual cost to reach a node and a heuristic value that estimates the remaining cost from the current position to the goal, effectively biasing the search toward more promising paths.

In our Co-op Tile World case, I used the A* algorithm for the agent to find the path for its next action, using the Manhattan distance from point **a** to point **b** as the heuristic function. In a grid world, the Manhattan distance provides a straightforward and effective heuristic because the agent's movement is restricted to horizontal and vertical directions; it can only move up, down, left, or right.

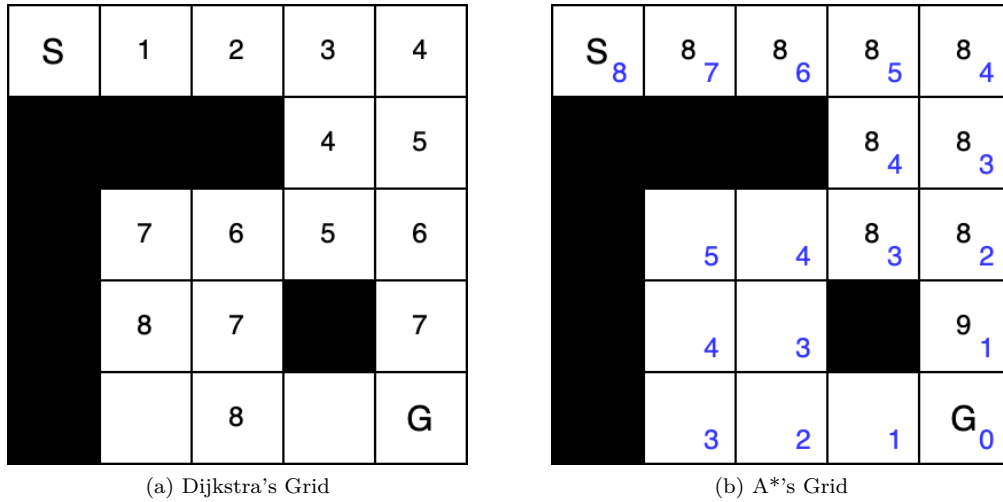


Figure 3.17: Dijkstra vs A* Algorithm comparison

Figure 3.17 illustrates a comparison between Dijkstra's algorithm and the A* algorithm. In the example grid, **S** denotes the start node, **G** denotes the goal node, and black cells represent obstacles. For simplicity, the edge cost is uniformly set to 1. Additionally, in Figure 3.17b, the blue numbers indicate the heuristic values, calculated as the Manhattan distance from each cell to the goal. The value inside each cell represents the total cost (i.e., actual cost plus heuristic, in the case of A*).

From the figures, we observe that Dijkstra's algorithm visits 14 nodes to find the shortest path, whereas A* visits only 9 nodes. This reduction in explored nodes is due to A*'s bias toward nodes with lower heuristic values, which guides the search more directly toward the goal.

However, the graph representation of Level 2 is more complex than the simple grid shown in Figure 3.17. This complexity comes from the presence of different types of tiles within the level. In particular, because some tiles depend on certain conditions—such as requiring specific items to pass—the graph must also include information about these tile dependencies and determine accessibility based on that information.

Pseudocode 1 presents a high-level overview of how the agent performs path planning in Level 2 using the A* algorithm.

Algorithm 1 Level 2 Agent Path Planning High Level Pseudocode

```

1: function CAN_TRAVERSE(tile)
2:   Determine if the agent can step onto tile given tile dependencies.
3: end function
4: function BUILD_GRAPH_FROM_BOARD
5:   Scan all tiles and connect each traversable tile to its neighbors.
6:   return graph
7: end function
8: function HEURISTIC(a, b)
9:   return  $|a.x - b.x| + |a.y - b.y|$ 
10: end function
11: function A_STAR_SEARCH(start, goal)
12:   Use  $A^*$  ( $f = g + h$ ) on the navigation graph.
13:   return path or None
14: end function
15: function CHOOSE_TARGET
16:   As below priority order:
17:   Exit if ready
18:   Preliminary items like boots and keys
19:   Nearby chips
20:   Remaining chips
21:   Otherwise, explore unexplored tiles
22:   return (position, type, path)
23: end function
24: function STEP
25:   if actions list empty OR stuck then
26:     (position, type, path)  $\leftarrow$  CHOOSE_TARGET
27:     actions  $\leftarrow$  PATH_TO_ACTIONS(p)
28:   end if
29:   Execute next action
30: end function

```

Layer 1: Cooperation Layers

Beyond the operational layer, a key question arises: how does the agent know how to act cooperatively with its human partner? In this thesis’s implementation, the agents do not explicitly learn what it means to act “cooperatively.” Instead, they are designed to respond rapidly and appropriately to changing game states. Let me elaborate on what this approach entails in the contexts of Level 1 and Level 2, respectively.

Level 1

For example, in Level 1, when training the `BC` agent, the positions of the remaining chips were provided as part of the training data. As a result, the agent learned to associate chip collection with moving toward the locations where chips were present. Consequently, the agent would avoid visiting positions where its partner had already collected the item. Similarly, based on the training data, once the `sockets_unlocked = true`, the agent learned to navigate to the exit, enabling it to respond appropriately when all chips had been collected.

This setup enabled the `BC` agent to simulate cooperative behavior to a certain extent. However, due to the inherent limitations of `BC` approaches—discussed earlier—the agent remained highly vulnerable to unfamiliar situations that deviated from its training data. In particular, the performance of the `BC` algorithm was sensitive to whether the agent was positioned on the right or left side. To improve robustness and reduce this positional dependency, I trained the `BC` agent multiple times using an accumulated dataset. After each training cycle, the resulting model was used as a surrogate agent, progressively improving the overall performance of the `BC` agent. This was a simple yet effective approach to training the `BC` agent in environments involving multiple agents. This iterative pipeline is well explained in Figure 3.18

Level 2

In Level 2, the agent operates based on a directed graph that represents accessible positions given the current game state, along with the cost of each path. As such, maintaining an up-to-date graph is crucial for simulating cooperative behavior. For example, if the human player acquires the red boots, the graph must quickly reflect this change by removing the need to visit the red boot location. To remain responsive to such dynamic updates, the agent recalculates the directed graph every game frame, allowing it to adapt its behavior in real time.

This approach does not involve the agent explicitly learning how to cooperate. Rather, it

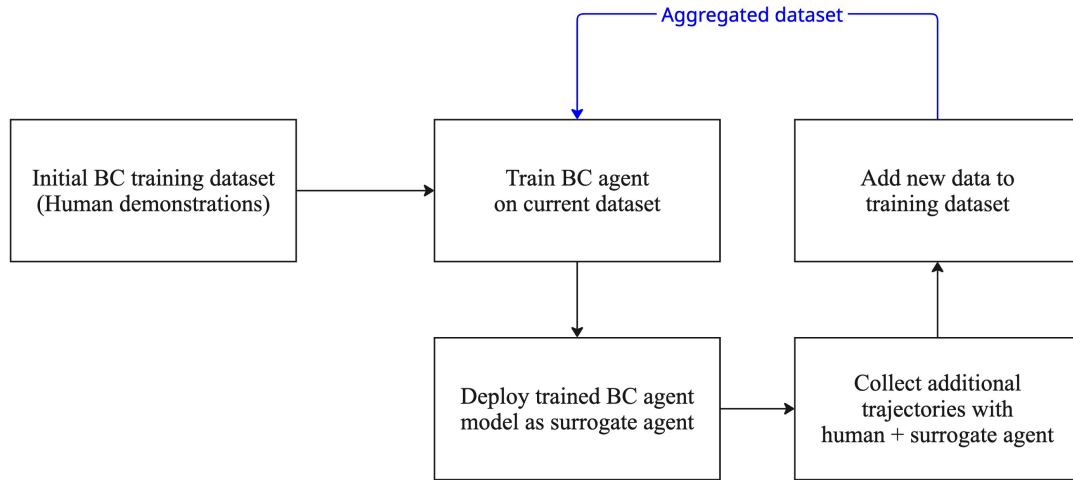


Figure 3.18: Iterative BC pipeline with surrogate agent data

simulates cooperative behavior by continuously adapting its strategy in response to the changing game state.

The agent is able to simulate cooperative behavior without explicitly learning how to cooperate, due to the way the levels are designed. Since both levels are solvable by a single player, the agent can respond to real-time game states in a manner that appears cooperative. This design choice provides a practical benefit in developing cooperative agents, as it avoids the complexity and computational overhead typically associated with training agents in multi-agent environments.

Conversely, levels can also be intentionally designed to require joint action—where cooperation is essential to complete the task. In such cases, the agent must learn explicit strategies for coordination under certain game state. Ultimately, the design of the cooperative layer is closely tied to the structure of the level itself: the more interdependent the task, the greater the need for agents to model and learn true cooperative behavior.

Layer 2: Communication Layers

For communication layer, the game’s communication logic is integrated with agent’s internal planning model. Agent has a separate planning model that plans the most optimal item assignment and the game renders agent’s item assignment when the function is called.

Level 1

Figure 3.19 illustrates how the alignment model is implemented for Level 1. In essence, the

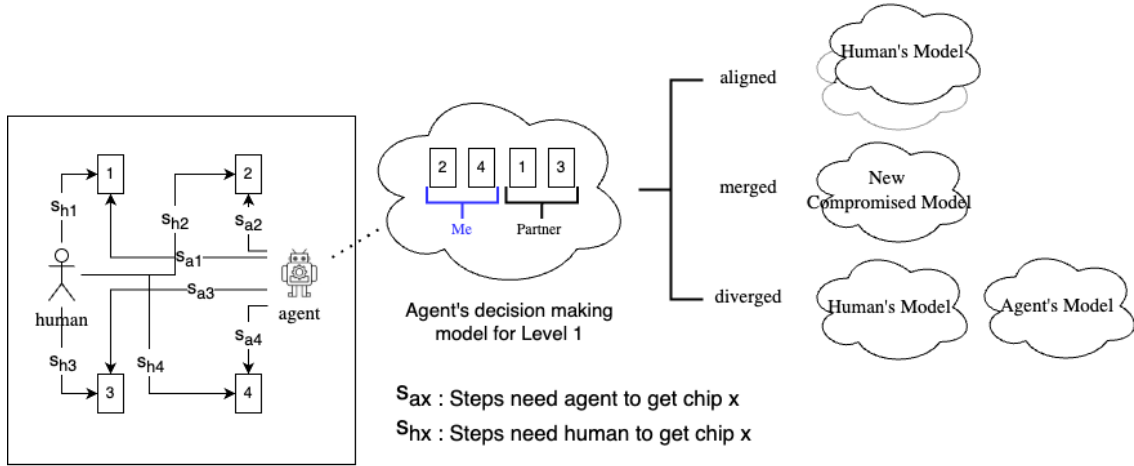


Figure 3.19: Agent's internal planning model for Level 1

agent assigns items based on their proximity to both its own location and that of its human partner. In the diagram, S_{ax} represents the number of steps the agent needs to reach item x , while S_{hx} indicates the number of steps required for the human to reach the same item. In the given configuration, items 2 and 4 are closer to the agent, while items 1 and 3 are closer to the human player. Accordingly, the agent's internal planning model assigns items 2 and 4 to itself and items 1 and 3 to the human. However, this initial assignment is subject to modification by the final alignment layer, which ultimately determines the agent's decision. The details of this decision-making process will be discussed in a later section.

Level 2

However, for Level 2, the internal planning model becomes more complex, incorporating tile dependency rules in addition to proximity-based planning. These dependencies are explicitly defined within the planning model. The **lhs** (left-hand side) represents the prerequisite items required to access those listed in the **rhs** (right-hand side). For example, if the agent already possesses **KEY_GREEN**, the planning algorithm will assign both **BOOT_FORCE** and **CHIP_GREEN** to the agent.

```

dependency_rules = [
    {"lhs": ["KEY_RED"], "rhs": ["BOOT_FIRE"]},
    {"lhs": ["KEY_GREEN"], "rhs": ["BOOT_FORCE", "CHIP_GREEN"]},
    {"lhs": ["KEY_BLUE"], "rhs": ["BOOT_WATER"]},
    {"lhs": ["BOOT_FIRE"], "rhs": ["CHIP_YELLOW"]},
    {"lhs": ["BOOT_WATER"], "rhs": ["CHIP_BLUE"]},
    {"lhs": ["BOOT_FIRE", "BOOT_WATER"], "rhs": ["CHIP_PINK"]},

```

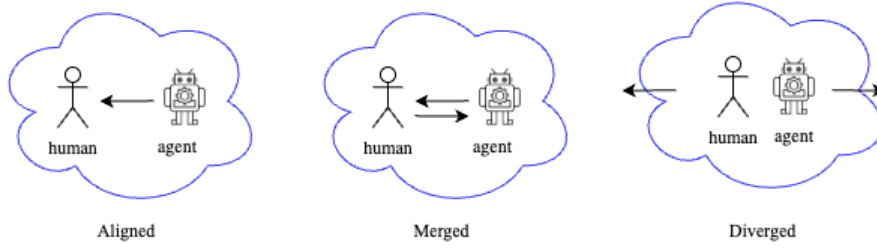


Figure 3.20: Three different alignment mode in collective decision making process

```

{"lhs": ["BOOT_FORCE", "BOOT_WATER"], "rhs": ["KEY_YELLOW"]},
]

```

As shown in Algorithm 2, the agent triggers `get_optimized_assignments` when `OPTIME` is activated. The function begins by retrieving all collectible items available in the current game state. From this list, the algorithm simulates assigning items in a turn-based manner, alternating between the two players.

In Step 1, the function selects the player whose turn it is and evaluates their current inventory. If a dependency rule is triggered—i.e., the inventory contains a prerequisite item listed in the rule’s left-hand side (LHS)—then corresponding right-hand side (RHS) collectibles is assigned to that player.

If no dependency rule applies, the algorithm proceeds to Step 2, where it selects the nearest accessible collectible based on proximity. After an item is assigned, the player’s simulated state is updated, and the turn is passed to the other player. This process repeats until all available collectibles have been assigned.

Layer 3: Three Alignment Modes

For this thesis specifically, we designed three different alignment modes in strategic decision making process. This is the social feature of the agent that the thesis chose to examine more closely using the custom benchmarking environment, Co-op Tile World. Figure 3.20 show three different types. As the names suggest, in the aligned model, the agent always follows the human’s decisions. In the diverged model, the human and agent ultimately fail to reach a common strategy, choosing to act independently. Meanwhile, in the merged model, the agents re-suggest its own plan to its human partner.

1. Aligned

Algorithm 2 Level 2 Agent Get Optimized Assignment High Level Pseudocode

```

1: function GETOPTIMIZEDASSIGNMENTS
2:   Initialize list of available collectibles
3:   Initialize simulated states for Player 1 and Player 2
4:   Provide dependency rules (LHS  $\rightarrow$  RHS)
5:   Set current player turn and track
6:   while  $\exists$ unassigned collectibles do
7:     Select player with current turn
8:     assignment_made  $\leftarrow$  False ▷ If True turn switches to the other player
9:     for each rule do ▷ Step 1: Check dependency rules
10:      if player has an item from rule.LHS then
11:        if a rule.RHS item is available then
12:          Assign it to the player
13:          Update player state and available list
14:          assignment_made  $\leftarrow$  True
15:          break
16:        end if
17:      end if
18:    end for
19:    if not assignment_made then ▷ Step 2: Fallback to nearest accessible item
20:      Find nearest accessible collectible
21:      if one exists then
22:        Assign it to the player
23:        Update state and remove item
24:        assignment_made  $\leftarrow$  True
25:      end if
26:    end if
27:    Switch to the other player's turn
28:  end while
29:  Save final assignments
30: end function

```

- **Process:** Player 1 suggests → Player 2 accepts → Both execute the plan together.
- **Outcome:** The agent always conforms to the human player's suggestion and performs the task collaboratively.

2. Merged

- **Process:** Player 1 suggests → Player 2 proposes an alternative → Player 1 decides whether to accept or reject the suggestion.
- **Outcome:** The agent proposes a plan generated by its own planning model. Depending on the human player's response, the task may be performed either collaboratively or individually.

3. Diverged

- **Process:** Player 1 suggests → Player 2 rejects → No common strategy is established.
- **Outcome:** The agent consistently rejects the human player's suggestion and proceeds to perform the task independently.

As previously discussed, the agent possesses its own internal planning model; however, its final decision is influenced by the selected alignment mode. In the aligned mode, the agent's decisions are entirely overridden by the human player. In the merged mode, the agent proposes a plan based on its own planning model, allowing the human player to either accept or reject the suggestion. In the diverged mode, the human and the agent each rely on their own models, operating independently without forming a shared strategy.

Integration with Communication Protocol

A key question arises: How do we reveal the agent's aligned mode using this communication protocol? In this game, accepting, rejecting, or re-suggesting an assignment reflects the degree of alignment between players.

For example, if Player 1 assigns a *fire boot* to Player 2, and Player 2 accepts the item and maintains the strategy throughout the game, this interaction represents the **aligned** type. Conversely, if Player 2 rejects the assignment and both players fail to converge on a common strategy, this interaction is classified as the **diverged** type.

Once human players finalize their decision, they click the *send* button on the right side of the panel. This triggers agent to run its internal model for strategic planning. In this process, the

thinking... UI appears to indicate that the agent is now processing your suggestion. Finally, when processing is done, the UI renders the other agent’s decision, which could be either *accept*, *reject*, or *new suggestion*. For each case, corresponding feed back is rendered on the screen, for example, “Player 1 Accepted”, “Player 1 Rejected”, or “Player 1 has a new suggestion” and printing the new assignment on the assignment list box on the top portion of the UI. Refer to [A.2](#) for detailed game screen shots for each case.

3.4 User Experiment

With the custom benchmarking environment Co-op Tile World and functional agent in place, the next step was to conduct a user study using the setup to evaluate players’ cooperative performance. This section outlines the experimental design and the criteria used to assess cooperative performance.

The objective of this experiment is to how the cooperative aspect differs from depending on the task difficulty and the alignment mode of agents in collective decision making process – aligned, merged, and diverged mode. To summarize, this experiment questions to this question: “How does the task difficulty and alignment in collective decision making affects the cooperative performance?”

3.4.1 Experiment Design

The experiment tests the cooperative performance of players in six test scenarios. There are two different levels of task difficulty and three different levels of degree of alignment in collective decision making process.

Task Difficulty	Alignment Mode		
	Aligned	Merged	Diverged
Easy	Easy-A	Easy-M	Easy-D
Difficult	Diff-A	Diff-M	Diff-D

Table 3.5: Six experiment scenarios of two levels of task Difficulty and three levels of agent’s alignment mode

Based on these variants, we are using a mixed experimental design, with alignment mode as a between-subjects factor and task difficulty as a within-subjects factor.

As shown in Table [3.6](#) the experimental design incorporates both between-subjects and within-subjects factors. **Alignment mode** is a between-subjects factor, as each participant is assigned to

Participants	Easy-A	Diff-A	Easy-M	Diff-M	Easy-D	Diff-D
P1	●	●				
P2			●	●		
P3					●	●
P4	●	●				
P5			●	●		
P6					●	●
⋮	●	●	●	●	●	●

Table 3.6: Participant assignment within a single alignment mode across two task difficulties

only one alignment condition, meaning that different participants experience different alignment modes. In contrast, **task difficulty** is a within-subjects factor, since each participant engages in both an Easy and a Difficult task. This allows for performance comparisons across two conditions within the same individual.

Statistical Analysis Method: Mixed ANOVA

ANOVA, which stands for Analysis of Variance, is a statistical method used to determine whether there are significant differences between group means. A mixed ANOVA is particularly useful when comparing means across groups that are cross-classified by two types of factors: between-subjects factors (independent categories) and within-subjects factors (repeated measures). This approach allows researchers to assess not only the main effects of each factor but also any interaction between them [45]. If the p-value for an effect is less than 0.05, the null hypothesis is rejected, indicating that the factor likely has a statistically significant influence. The p-value represents the probability of observing data as extreme as the current data, assuming the null hypothesis is true.

In this study, each participant is exposed to both levels of task difficulty (Easy and Difficult), representing a within-subjects factor, while interacting with only one agent alignment mode (Aligned, Merged, or Diverged), representing a between-subjects factor. Given this structure and the goal of analyzing cooperative performance across conditions, the design is well-suited for a mixed ANOVA analysis.

Using mixed ANOVA we can answer to below questions:

1. Does task difficulty affect cooperative performance?
2. Does an agent's alignment in decision-making affect cooperative performance?
3. Is there an interaction effect between task difficulty and an agent's alignment in decision-making?

After all, we either reject the null hypothesis—providing evidence in favor of the alternative—or fail to reject it, indicating insufficient evidence of an effect.

3.4.2 Experiment Evaluation Criteria

The next question is how to establish appropriate evaluation criteria for the experiment. In this study, the primary method for assessing cooperative performance is grounded in a game-theoretic approach, which provides a structured and quantitative framework for evaluating outcomes.

To complement this quantitative analysis, supplementary data are collected through participant surveys. These surveys provide qualitative insights into the nature of the cooperative interaction, capturing participants' reflections on their strategies, decision-making processes, and perceived dynamics of collaboration. This mixed-methods approach enables a more holistic understanding of both the measurable outcomes and the underlying rationale driving human-agent cooperation.

Quantitative Evaluation

For quantitative evaluation criteria, we use game theoretic approach. Specifically, the experiment suggests modified *Stag-Hunt* game. Below table shows the original game:

Player 2 / Player 1	Cooperate (Stag)	Defect (Rabbit)
Cooperate (Rabbit)	(5, 5)	(0, 3)
Defect (Rabbit)	(3, 0)	(3, 3)

Table 3.7: Stag-Hunt game payoff matrix

The Stag-Hunt game is a classic game theory model that represents a situation where two players can either cooperate (hunt a stag) or defect (hunt a rabbit). When two hunters cooperate, they can hunt a stag, which offers the highest incentive. However, if one hunter defects while the other chooses to cooperate, the defector hunts a rabbit while the cooperator gains nothing. When both hunters choose to defect, they each end up hunting a rabbit, yielding smaller incentives.

The game explores the trade-off between cooperation and individualism. In the game, both players must coordinate to succeed, but if one defects while the other cooperates, the cooperator gets nothing.

The Stag-Hunt game and the prisoner's dilemma differ primarily in how they balance cooperation and risk. In the Stag-Hunt game, mutual cooperation yields the highest payoff, but it requires both players to trust that the other will also choose to cooperate. If one player doubts the other's

commitment, they may choose the safer, lower-reward option instead. This creates two possible outcomes: one in which both players cooperate and receive a high reward, and another in which both defect, settling for a lower but more secure payoff.

In contrast, the prisoner’s dilemma presents a situation in which defection is the dominant strategy regardless of the other player’s decision. Although mutual cooperation would produce a better outcome for both, the fear of being exploited—and the desire to avoid the worst-case scenario—leads both players to defect. As a result, the game consistently settles into a single equilibrium of mutual defection, which is suboptimal from a collective standpoint.

We use Stag-Hunt framework to assess cooperative performance. Recall the game rule we set earlier.

- Each player can earn an individual reward based on the number of items that they individually collected.
- If both players touch exit tile within the time, then both players receive maximum payoff, which is calculated based on the remaining time.
- If only one player touches exit tile and the other fails, then only the exited player earns the individual reward, the other receives 0 reward.

The game rule modifies the original Stag-Hunt model to fit our game set up and give us quantitative tool to analysis the result.

Player 1 / Player 2	Exit	Fail to Exit
Exit	(M, M)	$(I, 0)$
Fail to Exit	$(0, I)$	(I, I)

Table 3.8: Modified Stag-Hunt model payoff matrix
 M = Maximum team reward based on remaining time;
 I = Individual reward based on items collected

Qualitative Evaluation

In addition to the primary quantitative reward reflecting the cooperative performance of the players, pre- and post-surveys are carried out to collect additional data. The pre-survey gathers information on the demographics and background of the participants, while the post-survey explores the rationale behind their chosen cooperative strategy and their perceptions of the agent’s cooperative behavior.

3.5 Data Collection

Running the code below generates the designated alignment mode for the agent, logs data for future analysis, and starts the user test.

```
python main.py --name "Participant Name" --lv1-alignment # --lv2-alignment #
```

Whenever the participant finishes the game .csv file will be written with following entries:

Participant, Level, Level Time Limit, Level Start Time, Level End Time, Play Duration, Level Cleared, Agent Alignment, P1 Ind Rew, P2 Ind Rew, Assignment Log, P1 Inventory, P2 Inventory, P1 Step Directions, P2 Step Directions, P1 Final Reward, P2 Final Reward

To elaborate functions for few key entries, P1 Final Points and P2 Final Points represent cooperative performance based on the game-theoretic framework. In addition, the assignment log indicates the result of players' strategic decision making. Lastly, the P1 and P2 Step Directions show the strategic movement of the players during the game. These data back up the cooperative decision making and behavior of agents during the game along with the quantifiable total reward.

The full set of raw data collected through user tests can be found in Figure [C.1](#).

4

Result and Findings

Contents

4.1 Results	67
4.1.1 Demographics	67
4.1.2 Descriptive Statistics	70
4.1.3 Inferential Statistics	74
4.2 Findings	76

Citations in this chapter: [12], [23], [32], [40], [46–50].

Acronyms in this chapter: CAI.

4.1 Results

The results is presented in two ways. First, through descriptive statistics, which aim to summarize and describe the collected dataset without making conclusions about a larger population. Second, the thesis reports inferential statistics, which allow us to draw conclusions or make inferences about the broader population based on the sample data. Specifically, this research conducted hypothesis testing using a Mixed ANOVA analysis.

4.1.1 Demographics

Experiments were conducted with a total of 18 participants, resulting in 18 data points for comparing the within-subject independent variable—task difficulty—and 6 data points for comparing the between-subject independent variable—agent alignment mode in decision making.

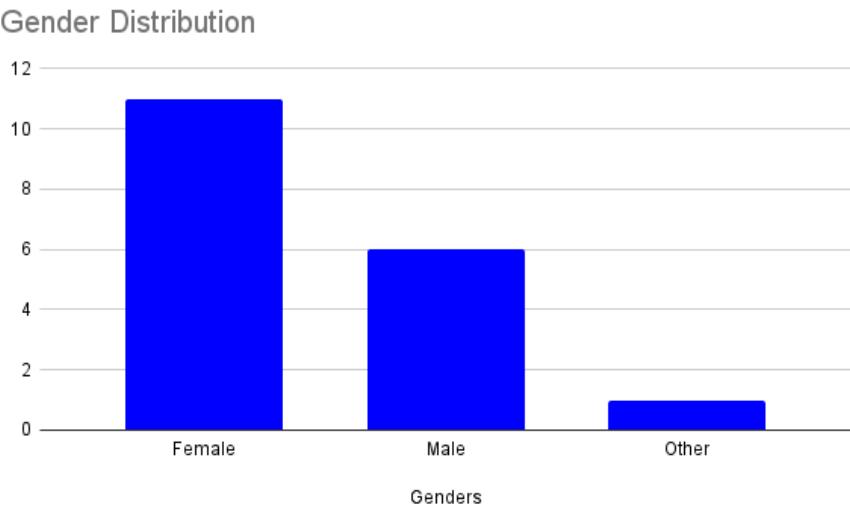


Figure 4.1: A chart showing gender of participants.

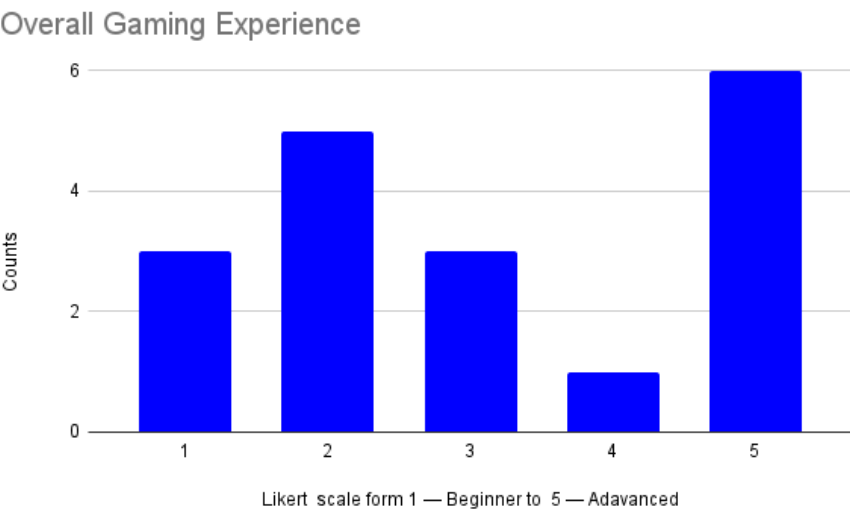


Figure 4.2: A chart showing overall gaming experience per participants.



Figure 4.3: A chart showing prior puzzle game experience.



Figure 4.4: A chart showing prior cooperative game experience.

Through a pre-survey, I collected demographic information from the experiment participants. As shown in in [4.1](#), among the 18 participants, 11 identified as women, 6 as men, and 1 as other. Additionally, participants were asked to self-report their general gaming experience on a scale from 1 (beginner) to 5 (advanced). The survey also inquired specifically about their experience with puzzle games (e.g., Minesweeper, Sudoku) and cooperative games (e.g., Mario Kart, Overcooked, It Takes Two). For detailed gaming experience refer to Figures [4.2](#), [4.3](#), [4.4](#).

Participants reported a broad range of overall gaming experience levels (Figure [4.2](#)). 6 participants rated themselves at the highest level (5 – Advanced), followed by 5 at level 2. 3 participants identified as beginners (1 - Beginners), 3 at level 3, and only one at level 4.

When it comes to puzzle games (Figure [4.3](#)), 7 participants indicated they had tried such games a couple of times, while 6 played them occasionally, and 4 considered themselves experienced. Only one participant reported that they had never played puzzle games.

Regarding the experience with cooperative games (Figure [4.4](#)), 9 participants had tried cooperative games a couple of times, 5 reported regular play and considered themselves experienced, and 4 played them occasionally. No participant reported having never played cooperative games.

4.1.2 Descriptive Statistics

In this section, I summarize and describe experiment results using different types of plots – bar graph, box plot, and fixed bar graph.

Bar Graph for Mean Comparison

As shown in Figure [4.5](#), agents operating in the aligned mode (Align 0) consistently achieve the highest normalized final rewards across both task levels. At Level 1, the aligned condition yields a mean reward close to 29%, followed by the merged mode (Align 1) at approximately 26%, and the diverged mode (Align 2) at 22%. A similar ordering is observed in Level 2, where aligned agents again lead with a reward near 29%, while merged and diverged agents drop to roughly 19% and 8%, respectively.

This pattern confirms a consistent ranking of cooperative performance: Aligned > Merged > Diverged. Furthermore, task difficulty introduces a noticeable performance decline from Level 1 to Level 2, but the magnitude of this drop differs by alignment mode. The aligned mode shows

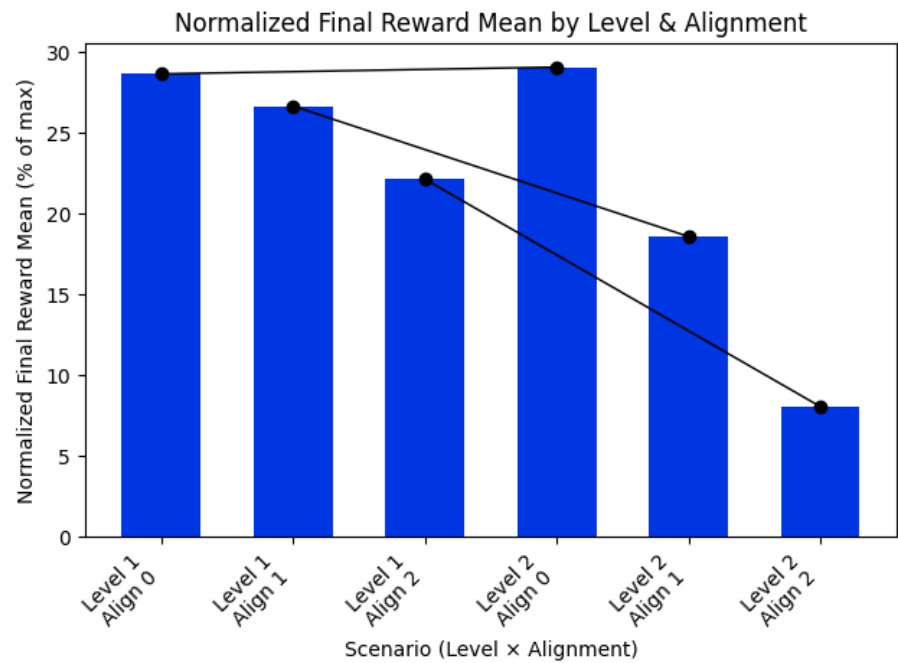


Figure 4.5: Bar chart comparing normalized final rewards across conditions

minimal decline, less than 1 percent point, suggesting robustness to increased difficulty. In contrast, merged agents see a moderate decrease of about 7 percentage points, and diverged agents experience a sharp performance drop of around 14 percentage points. These results highlight both the benefit of alignment and its interaction with task difficulty in shaping cooperative outcomes.

Box Plot

Figure 4.6 shows the box plot of the data distribution. The box represents the interquartile range (IQR) of the middle 50% of the data, from the 25th percentile to the 75th percentile.

In Level 1, both the aligned and merged modes exhibit a tall box with very short whiskers. This pattern indicates high variability within the IQR, meaning participants’ performances varied considerably within the middle 50% of the data. However, the short whiskers suggest that values outside the IQR are relatively concentrated, showing low variability in the tails. Additionally, the median is slightly skewed toward the upper end of the box, implying that cooperative performance is more densely clustered near the higher end of the distribution. Together, these characteristics suggest that while there was some variation in outcomes, overall cooperative performance in the aligned and merged modes was relatively strong in Level 1.

In contrast, the diverged mode displays a short box with long whiskers, indicating low variability

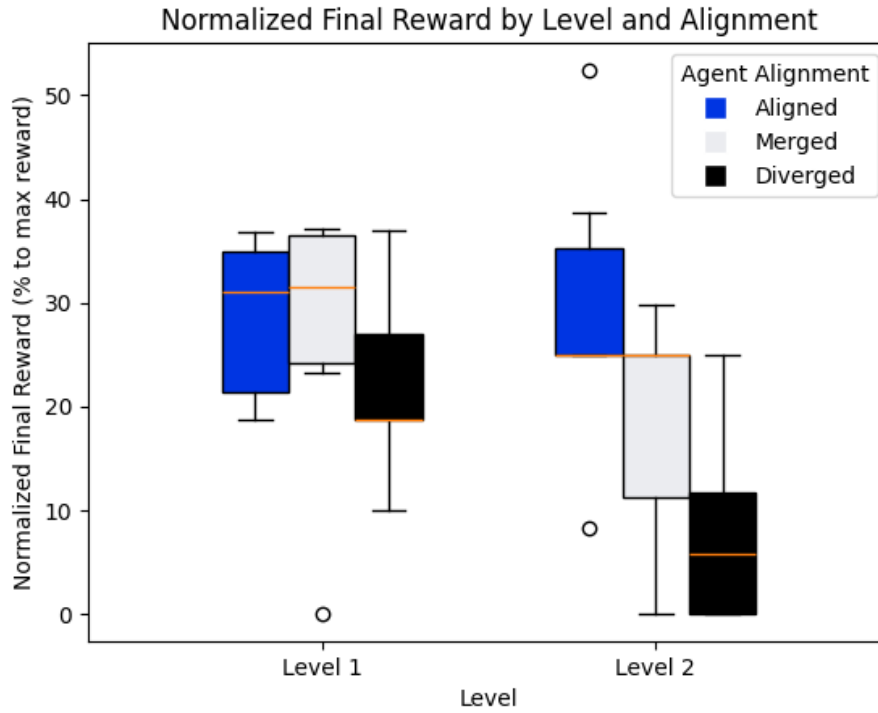


Figure 4.6: Box plot of normalized final rewards by task difficulty and agent alignment

within the IQR but high variability in the tails. The median is skewed toward the lower end of the box, suggesting that performance was more concentrated near the lower end of the distribution. This overall pattern reflects both a broader spread in participant outcomes and generally weaker cooperative performance in the diverged mode for Level 1.

In Level 2, the distributions across alignment modes diverge significantly. For the aligned mode, the box height is even shorter than in Level 1, and only a short upper whisker is present—no lower whisker is observed. This indicates extremely low variability in cooperative performance. Notably, the median lies at the bottom edge of the box, suggesting that a large portion of participants achieved identical or near-identical scores clustered at the high-performance end. This distribution reflects a performance plateau, where many participants reached similarly high outcomes. Overall, these characteristics suggest that the aligned strategy was not only effective but also highly robust under increased task difficulty.

For the merged mode in Level 2, both the IQR and the whiskers are tall, indicating high overall variability in participants' cooperative performance. The median is skewed toward the upper end of the box, suggesting that while a portion of participants achieved higher scores, the lower half of the distribution is more spread out. In other words, the variability is concentrated among the lower-performing participants, while the higher scores are more tightly clustered.

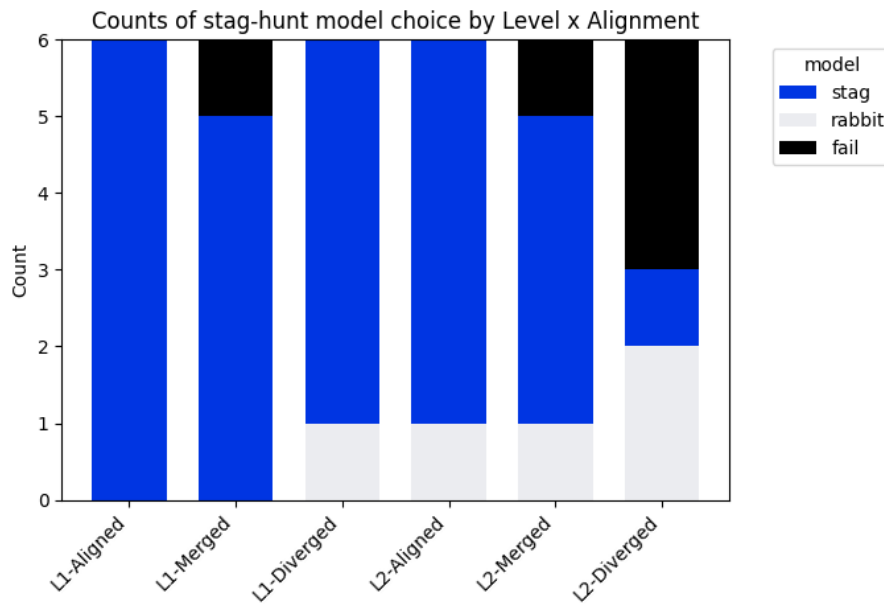


Figure 4.7: Box plot of Stag-Hunt model choice counts across levels and agent alignment

Lastly, the diverged mode in Level 2 exhibits a notable decline in cooperative performance. The box is anchored at a normalized reward of 0, with no visible lower whisker, indicating that a substantial portion of participants achieved minimal scores. The median lies near the center of the box, meaning that even the 75th percentile of participants scored below 10% of the final reward. In contrast, the long upper whisker reveals high variability among the top-performing participants, suggesting that while most struggled, a few achieved significantly higher outcomes.

Fixed Bar Graph for Stag-Hunt Model Choice

The results shown in Figure 4.6 are further illuminated by the fixed box plot in Figure 4.7, which illustrates the Stag-Hunt strategies chosen by participants in each scenario. As observed, in most cases, players successfully coordinated to hunt stags—indicated by the dominance of blue segments in the plot. The performance variability primarily reflects differences in how quickly each player completed the level while pursuing the stag strategy. However, a notable exception appears in Level 2 under the diverged mode: only one player succeeded in hunting a stag, while three players failed to receive any reward, and two were able to hunt rabbit instead. This pattern underscores the difficulty of maintaining cooperative behavior with diverged agent.

4.1.3 Inferential Statistics

ANOVA (Analysis of Variance) is a hypothesis testing method used to draw inferences about a population based on sample data. Because direct inference from samples can be unreliable, hypothesis testing typically begins with a null hypothesis— assuming that there is no effect of the independent variable x on the dependent variable y . ANOVA then evaluates whether the observed differences between group means are statistically significant enough to **reject** this null hypothesis.

In this study, we employ a Mixed ANOVA, which compares group means across two types of factors: between-subjects factors (independent groups) and within-subjects factors (repeated measures on the same participants). This method also enables the analysis of interaction effects, revealing whether the influence of one independent variable depends on the level of another.

Mixed ANOVA test was performed using python library `pingouin`.

Result

Source	SS	df	MS	F	p	η_p^2
Agent Alignment	1140.20	2	570.10	3.14	.073	.30
Task Difficulty (Level)	472.46	1	472.46	4.93	.042	.25
Alignment $\times$ Level Interaction	318.37	2	159.19	1.66	.223	.18

Table 4.1: Mixed ANOVA Results for the Effect of Agent Alignment and Task Difficulty on Cooperative Performance.

A final mixed ANOVA result is shown in table 4.1. A 2×3 mixed ANOVA was conducted on the dependent variable normalized cooperative performance data to find the main and interaction effect of independent variables task difficulty and alignment modes.

A mixed-design ANOVA revealed a significant main effect of task difficulty, $F(1, 24) = 4.93$, $p = .042$, partial $\eta^2 = .25$, indicating a large effect size. The main effect of agent alignment was not statistically significant, $F(2, 24) = 3.14$, $p = .073$, although the effect size was large, partial $\eta^2 = .30$. The interaction between alignment and level was also non-significant, $F(2, 24) = 1.66$, $p = .223$, however the effect size was large, partial $\eta^2 = .18$.

For the p-value threshold, 0.05 was used to determine statistical significance. To interpret the strength of observed effects, effect sizes were evaluated using Cohen's (1988) guidelines [47], [49].

Effect size is a quantitative measure of the magnitude of the difference or the strength of the association between variables. P -values tell researchers whether the effect is likely due to chance

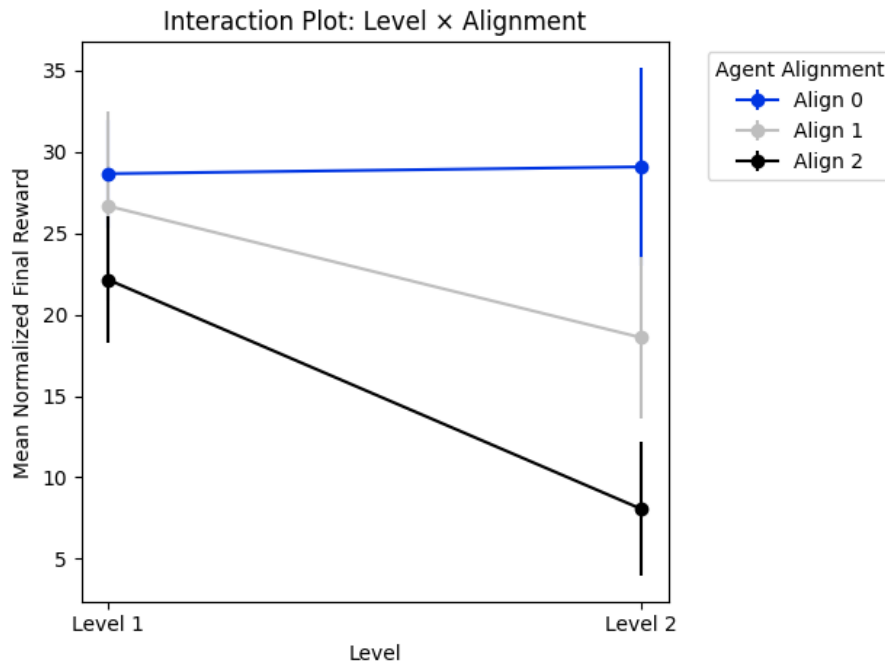


Figure 4.8: Level and Alignment Interaction Plot

taking into account the sample size, while the effect sizes tell how big the effect is, and is calculated independent of the sample size. Increasing sample size makes the statistical test results more likely to find a statistically significant effect, no matter how small the effect truly is in the world. Therefore, effect size is a complementary value p -values.

Effect size is a quantitative measure of the magnitude of a difference or the strength of an association between variables. [46] First, p -values indicate whether an observed effect is likely due to chance, depending on the sample size. However, effect size reflects how large or meaningful that effect actually is, independent of sample size. As sample size increases, even very small effects may reach statistical significance. Therefore, effect size serves as an important complement to p -values, providing insight into the practical significance of a result. [50]

The analysis revealed no significant main effect of Alignment, $p = .073$. There was a significant main effect of Level, $p = .042$. No significant interaction was found between Level and Alignment, $p = .223$. Therefore, we reject the null hypothesis that task difficulty has no effect on cooperative performance, as the p -value .042 is less than the standard significance threshold of 0.05. However, we fail to reject the null hypotheses for alignment mode and the interaction effect.

Limitations

In our experiment, the main effect of alignment mode and the interaction effect did not reach statistical significance, likely due to the small sample size. However, both effects exhibited meaningful effect sizes, suggesting that the associations between variables may be practically relevant. As shown in the interaction plot (Figure 4.8), cooperative performance consistently decreases across alignment modes—from aligned to merged to diverged. Additionally, there appears to be an interaction effect between the two variables, as cooperative performance declines more sharply in Level 2 compared to Level 1. These findings imply that with a larger sample size, statistically significant effects may emerge, supporting the potential value of conducting follow-up experiments with a larger participant pool.

4.2 Findings

Task Difficulty and Cooperative Performance

Almost all players chose to cooperate even when they were uncertain about the agent’s ability. No player was being greedy to collect more items to get more individual rewards. Their assignment strategy aimed to clear the level as fast as they can. Notably, experiencing low task difficulty during the initial interaction helped build successful cooperative experiences, which in turn promoted cooperation in Level 2 as well.

However, not only the success of cooperation but also the way it failed influenced future cooperation. When a failure was perceived as a lack of willingness or disregard for teamwork, humans often responded with a tit-for-tat strategy, retaliating against the agent in subsequent levels.

In this playtest, human players generally outperformed the agent, as they were able to move faster. The agent, constrained by a 400 ms cooldown between moves, rarely managed to finish the level before its human partner. However, in one case, the agent succeeded in reaching the exit tile before the human player. Due to the three-second countdown rule after the agent exited, the human player failed to exit in time and received none in Level 1.

Then, in the following round, the human player changed strategies—choosing to leave immediately after the agent collected the final chip, rather than waiting for the agent. This decision

effectively sacrificed a larger team reward in exchange for a smaller individual reward. This behavior is notable, as it demonstrates a shift in trust on cooperation.

The player chose to prioritize their own action as a direct response to the agent’s earlier behavior, not to maximize future outcomes. The human player appeared to withdraw cooperation intentionally, exhibiting what behavioral game theory identifies as a **tit-for-tat** strategy. When cooperation is violated or trust is perceived to be broken, humans often respond by mirroring that behavior—even if it means forgoing higher potential rewards. This reflects not just a breakdown in strategic alignment, but also a shift in the social dynamics of the interaction.

This observation aligns with findings on how trust is formed, violated, and how people respond in reciprocal game settings. According Ernst Fehr and Simon Gächter, altruistic punishment—where individuals penalize others at a personal cost without any material gain—is a key mechanism that promotes human cooperation, even among strangers and in one-shot interactions, as demonstrated in their research [48].

In contrast, when the failure was attributed to the agent’s limited ability rather than its intent, humans were more forgiving and willing to give the agent another chance. This time, the human player was waiting for the agent to complete the task so they could receive the team reward in Level 1. However, the agent malfunctioned and failed to complete its task. As a result, the human player had no choice but to exit the level alone, receiving only the smaller individual reward. What was particularly noteworthy was that, despite this failure, the human player continued to wait for the agent in the subsequent round—still aiming for team cooperation.

The contrast between these two scenarios highlights an important insight: how cooperation fails influences whether people choose to cooperate in subsequent rounds. When the failure of cooperation is perceived as a lack of willingness or disregard for teamwork, humans often respond with a tit-for-tat strategy. In contrast, when the failure is attributed to the agent’s limited ability rather than its intent, humans may be more forgiving and willing to give the agent another chance.

While these conclusions are drawn from just one observation in each case, they suggest valuable directions for future research: **The success or failure of a cooperative experience, and how that experience unfolds, shapes human willingness to cooperate in future cooperation.**

Alignment Modes and Cooperative Performance

How can we explain the differences in cooperative performance depending on alignment modes? The results may be interpreted as to whether the players have a shared plan or not. In essence, having a shared plan leads to improved cooperative performance. The aligned mode consistently achieved the highest performance because a shared plan was always present for both agents to follow. Similarly, in the merged mode, cooperative performance was higher when the human player compromised and followed the agent's new suggestions, rather than rejecting them and acting independently without a shared plan. Consequently, the diverged mode resulted in the poorest performance, as no shared plan was established at any point.

The agent's alignment mode was ultimately linked to whether an explicitly assigned shared plan was provided. When the team reached an agreement, the assignment UI clearly displayed which items each player was responsible for collecting. In contrast, when no shared plan was established, the UI remained empty. This seemingly minor difference had a noticeable impact on cooperative performance. With a predefined shared plan, players could simply follow the prescribed sequence of actions without having to consciously deliberate on their next move at every step.

The agents also performed well with shared plans. Since the shared plan is visible to both the human and the agent, the agent encountered fewer uncertain situations because both players acted in accordance with the predetermined plan. The handling of uncertainty is one of the key challenges in AI development. However, by implementing a shared plan under the assumption that both players are committed to executing it, the level of uncertainty the agent must manage is significantly reduced, resulting in better performance compared to situations involving unshared, individual plans.

Although the effect of alignment mode on cooperative performance was not statistically significant, a clear trend was observed within the sample, accompanied by a large effect size: performance declined from aligned to merged to diverged modes. This finding suggests that explicitly sharing plans can enhance cooperative performance.

In terms of diverged mode agent especially, participants hated the divergent model. They especially raged with the fact that agents don't even explain why they rejected the suggestion although their assignment plan seemed to be reasonable.

According to Hancock et al., robot performance had the greatest impact on trust formation [23]. Similarly, our playtest revealed a comparable pattern. Although most participants felt very

uncomfortable with the diverged agent, one participant remarked, “If their performance is exceptional and they are rejecting suggestions, I would reflect on that. However, their performance was not better than mine, yet they were still rejecting suggestions. I can’t accept that.”

The agent’s poor performance—specifically, its inability to dynamically adapt to changing environmental states in certain situations—negatively impacted participants’ satisfaction with the cooperative interaction, especially when coupled with the agent’s apparent unwillingness to cooperate (e.g., repeatedly rejecting the partner’s suggestions). In contrast, when the agent performed beyond expectations, participants reported greater satisfaction with the collaboration. While this outcome may seem intuitive, it underscores a critical point: an agent’s social capabilities are ultimately grounded in the reliability of its autonomous performance.

Task Difficulty and Alignment Modes Interaction Effect

Although the interaction effect between the two independent variables was not statistically significant, from Level 1 to Level 2, aligned agents exhibited a slight drop, merged agents showed a moderate drop, and diverged agents experienced a sharp decline in cooperative performance. As task difficulty increased, the absence of a shared plan led to a more dramatic decrease in cooperation. This suggests that **as tasks become more complex, it becomes even more crucial to explicitly communicate shared plans.**

However, the preferred style of communication may vary depending on the player’s skill level. Qualitative responses revealed that players generally enjoyed playing with aligned agents. More experienced participants appreciated the merged mode, often enjoying the process of interpreting the agent’s new suggestions. Some even expressed disappointment when the agent simply followed their plans without proposing alternatives.

In contrast, novice players perceived the agent’s suggestions as an added cognitive burden. Already focused on understanding the game’s rules and mechanics, these participants found the additional layer of communication cognitively taxing—especially as task difficulty increased. This increased cognitive load may have contributed to the decrease in cooperative performance.

These findings point to promising directions for future research. While strategy exchange and iterative negotiation may help experienced players arrive at more efficient strategies, they do not universally lead to better cooperative performance. For less experienced users, such exchanges may feel overwhelming or confusing. If this pattern repeats, novice users may eventually stop analyzing

the agent's suggestions, blindly relying on the agent's plans and, in doing so, relinquishing their own agency in the cooperative interaction. This suggests that an agent's alignment behavior in collective decision-making process should be dynamically adapted to the skill level of its human partner—particularly in tasks that involve high complexity.

5

Conclusion

Contents

5.1 Summary	81
5.2 Contributions	83
5.2.1 Development of Custom Benchmarking Environment	83
5.2.2 User Test and Verification	85
5.3 Limitations	86
5.3.1 Agent Design	87
5.3.2 Level Design	87
5.3.3 Limited sample size	88
5.4 Future Work	89
5.5 Discussion	91

Citations in this chapter: [7], [12], [32], [40], [51], [52].

Acronyms in this chapter: CAI, Dependent Variable (DV), HRI, Independent Variable (IV), OPTIME, RL, ToM.

5.1 Summary

This research aims to answer the following two questions:

- RQ1 (Design): How should we design a benchmarking game environment for cooperation between humans and socially intelligent agents?
- RQ2 (Assessment): How can we assess cooperative performance between humans and socially intelligent agents using this custom environment?

To address the first research question, we developed a customizable benchmarking environment called Co-op Tile World, built upon the core mechanics of the open-source game Tile World 2 [7], but redesigned as a two-player cooperative game. Unlike typical reinforcement learning benchmark environments—many of which are designed for AI-to-AI interaction or use proxy humans—Co-op Tile World is specifically devised for real-time cooperation between human players and AI agents. A key feature of the environment is its embedded communication protocol, which enables strategic opinion exchange between the human and the agent. This was implemented leveraging a game mechanic centered on *item collection*, allowing players to assign items either to themselves or to their partner as a way to express their strategic planning. Additionally, the environment supports a variety of tile types and collectible items enabling researchers to design diverse cooperative gameplay scenarios.

Building on this general setting, this research specifically designed two distinct levels with varying task difficulty. The difficulty was introduced by incorporating diverse tile types, requiring players to understand each tile’s effects and the dependencies between them. Additionally, the agent was implemented in three different alignment modes for collective decision-making: aligned, merged, and diverged. This design allows us to examine how task difficulty (IV 1) and the agent’s alignment in decision-making (IV 2) affect cooperative performance (DV).

Using the benchmarking environment proposed to address RQ1, I evaluated the cooperative performance between humans and agents by adopting a game-theoretic framework, specifically, the Stag Hunt model. The Stag Hunt is a classic scenario in game theory that illustrates the tension between cooperation and self-interest: two players must choose between cooperating to hunt a stag, which offers a high reward only if both commit, or acting independently to hunt a rabbit, which yields a smaller but guaranteed reward. This model highlights the trade-off between collective benefit and individual security, making it well-suited for evaluating human-agent cooperative task.

In our adapted version, if both players successfully exit the level within the time limit, they receive a large team reward. If only one player manages to exit, that player receives a smaller individual reward, while the other receives nothing. If neither player exits in time, both receive no reward. This setup provides a direct analogy to the Stag-Hunt dilemma, enabling us to measure cooperative dynamics under varying conditions.

Lastly, we analyzed the results using both descriptive and inferential statistics, specifically employing a mixed ANOVA. The analysis revealed that task difficulty had a statistically significant effect on cooperative performance. Although the main effect of alignment mode and interaction

effect was not statistically significant at the population level, a meaningful trend was observed within the sample. These findings offer valuable information for the development of cooperative agents that can achieve better performance within context-dependent environments. The detailed implications and broader utility of this research will be discussed in the later section, [5.5](#)

5.2 Contributions

The thesis, Co-op Tile World: A Benchmarking Environment for Human-AI Cooperative Interaction, contributes to the research domains of Cooperative AI, Robot Learning, and Human-Robot Interaction. Developing socially cooperative behavior grounded in strong autonomous capabilities represents a critical next step in AI research. As AI technologies continue to advance, intelligent agents are increasingly expected to operate not in isolation, but as part of interconnected systems—cooperating with both humans and other agents forming complex networks.

Historically, the continuous advancement of powerful AI algorithms has been driven by well-designed benchmarking environments. While several pre-built learning environments exist to evaluate cooperative performance, few are specifically designed to support real-time cooperation between real humans and AI agents. In this spirit, I propose Co-op Tile World—a benchmarking environment purposefully designed for real-time human–AI cooperation, rather than relying on AI-to-AI interactions or proxy human simulations.

5.2.1 Development of Custom Benchmarking Environment

Through this work, I delivered a benchmarking environment that tests human and computer interaction. This environment considers the following factors:

- a) The environment should be intuitive for human players to operate, while also allowing space to train agents with behaviors ranging from simple to sophisticated.
- b) The environment should be easily customizable, enabling researchers to fine-tune its design based on specific research goals.
- c) The environment should support a communication protocol between the human and the AI agent.

- d) The environment should be flexible enough to program agent’s key social dynamics such as alignment, commitment, trust, and explainability.

Considering above factors, I decided to work based on the existing open source game called *Tile World 2* [7]. *Tile World* was appropriate code base for the purpose of creating benchmarking environment for human and machine’s cooperative interaction for following reasons:

a) **Simplicity and Expandability:**

- The core game mechanics of the original *Tile World 2* are simple and intuitive, allowing human players to engage with the game easily.
- However, by combining various tile effects, collectible items, and monsters, designers can construct complex and challenging environments to support the training of sophisticated agent behaviors.
- The variety of tile mechanics enables researchers to design cooperative tasks tailored to their specific research goals.

b) **Customizable Environment Design:**

- With the Google Sheets-based *Tile World Level Editor*, researchers can easily create custom game environments aligned with their research objectives.
- By importing the JSON file generated by the Level Editor, researchers can instantly deploy the environment without needing to re-implement any game mechanics.

c) **Communication Protocol:**

- The item collection mechanic from the original *Tile World 2* integrates naturally with the communication system.
- Players can express their strategic intentions by assigning collectible items to themselves or to their partner, making item assignment as a form of strategic communication.

Building on the original *Tile World* game mechanics and assets, the environment was converted into a cooperative game where a human and an AI agent can play together. To enable this, several modifications were introduced: the system now supports human input via arrow keys for movement and mouse clicks for interacting with the communication UI; a reward-earning system was added to quantify cooperation; item assignment serves as a communication protocol between players; and each player maintains a separate item inventory to track individual rewards.

More importantly, the Co-op Tile World environment is designed to be extensible. Researchers can build upon the existing configuration by adding custom game rules or elements tailored to their research goals. As a result, the platform offers a scalable and structured benchmarking environment for both quantitative and qualitative assessments of human–AI cooperative performance.

5.2.2 User Test and Verification

This research not only introduces a benchmarking environment for evaluating human–AI cooperative performance but also demonstrates its application through a user study investigating the effect of agent alignment modes on collaborative decision-making under varying task difficulty levels.

The experiment utilized Co-op Tile World to measure cooperative performance across six distinct scenarios, each representing a unique combination of task difficulty (Level 1 or Level 2) and agent alignment mode (aligned, merged, or diverged). The study employed both descriptive and inferential statistical analyses within a game-theoretic framework.

Descriptive results revealed a consistent decline in cooperative performance as task difficulty increased. Moreover, agent alignment mode had a notable influence: human players paired with aligned agents achieved the highest performance, followed by those with merged agents, and finally, those with diverged agents. Performance also declined more sharply from Level 1 to Level 2 in the merged and diverged conditions, suggesting a potential interaction effect between task difficulty and alignment mode in shaping collaborative behavior.

However, these trends were not fully supported by inferential analysis. The mixed ANOVA revealed that only the main effect of task difficulty reached statistical significance ($p = .042$), falling below the conventional threshold of .05. As a result, we could only reject the null hypothesis that

task difficulty has no effect on cooperative performance, while the effects of agent alignment and the interaction between factors did not reach statistical significance—likely due to limited sample size.

Below are key findings from the analysis:

1. **Main Effect of Task Difficulty:** Successful cooperative experiences in Level 1 often led to continued cooperation in Level 2. However, the way in which cooperation **failed** also influenced how collaboration unfolded in subsequent rounds. When participants perceived the agent’s behavior as a betrayal—such as deliberately acting against the shared goal—they

often responded with a tit-for-tat strategy in the next round. In contrast, when the agent’s failure was attributed to a lack of capability rather than intent, participants tended to be more forgiving and were willing to give the agent another chance.

2. **Main Effect of Alignment Modes:** Shared plans led to more stable and successful co-operation. Aligned agents produced the best outcomes by making shared plans explicit and visible to the human player. Interestingly, shared planning also benefited the agents by reducing environmental uncertainty, highlighting the importance of mutual predictability in cooperative tasks.
3. **Interaction Effect:** As task difficulty increased, the absence of a shared plan led to a more dramatic decrease in cooperation. This suggests that as tasks become more complex, it becomes even more crucial to explicitly communicate shared plans. However, the preferred style of communication may vary depending on the player’s skill level. Novice players often perceived the added communication layer as an extra cognitive burden while more experienced participants tended to enjoy the added communication layer.

To summarize, this research contributes to the development of socially cooperative agents capable of engaging in **socially grounded cooperative interaction**—interactions that generate rich social data and feedback during the interaction. Despite growing interest in this area, there remains no comprehensive benchmarking environment specifically designed to evaluate the performance of socially intelligent agents interacting with real humans. Addressing this gap, I proposed a custom benchmarking environment, Co-op Tile World, and conducted an empirical study to examine how specific cooperative features and environmental configurations influence players’ cooperative behavior.

As previously discussed, the environment is both scalable and customizable, allowing researchers to tailor it to their specific research objectives. By leveraging these qualities, this thesis aims to support continued progress in Cooperative AI, particularly in the development of socially cooperative agents that emulate the complex dynamics of real human cooperation.

5.3 Limitations

This section outlines the limitations of this research in three areas: agent design (5.3.1), level design (5.3.2), and the limited sample size in user testing (5.3.3).

5.3.1 Agent Design

The first limitation emerged during the playtest, specifically in how participants perceived the merged agent’s feedback. The original intention behind designing the merged mode was to create an agent that generates its own suggestions and offers feedback to its partner—simulating a back-and-forth exchange of opinions, rather than simply accepting or rejecting the human player’s input. However, contrary to this intention, many participants perceived the agent’s new suggestions as outright “rejections” of their own ideas. As a result, for those participants, the merged mode felt similar to interacting with a diverged agent.

This limitation stems from two main factors: the design of the merged communication itself and the structure of the game. Let’s first discuss the communication design. In the merged mode, the sequence of events unfolded as follows:

OPTIME requested → Human player assigns collectible items → Send the assignment → Agent internal model renders new assignment plan on the UI → Now human player has choice to either reject or accept.

This is how a typical cycle in the merged mode unfolds. The core issue was diagnosed in the final step, where the human player is only given the option to either accept or reject the agent’s decision. This binary choice limited the opportunity for iterative suggestion exchanges, which is the essence of a truly merged decision-making process.

Secondly, the game’s tightly constrained time component further limited opportunities for extended communication. After the first automatic **OPTIME** trigger, players rarely initiated additional **OPTIME** requests to propose new suggestions or make adjustments. Once the **OPTIME** phase ended, players were either focused on executing their assignments or too occupied solving the level itself. As a result, the alignment mode functioned less as an expression of the agent’s social capabilities and more as a structural condition that determined players to achieve shared planning or not.

5.3.2 Level Design

Through multiple training sessions, it became evident that the 13×13 grid filled with diverse tile types was too complex for effectively training a reliable **RL** agent. The original goal of the Co-op Tile World environment was to develop a scalable **RL** benchmarking platform, similar to pre-built

games in OpenAI’s reinforcement learning framework, *Gym*. However, repeated training attempts revealed that the current map size and the complexity of its tile mechanics posed significant challenges for creating an easily trainable and testable environment.

Supporting an efficient agent training pipeline is essential, as the primary objective of this project is to enable the rapid development and evaluation of socially cooperative agents through real-time gameplay with human partners. To achieve this, the environment must not only provide rich and diverse game mechanics but also support robust and practical training pipelines.

Another factor that contributed to the difficulty of agent training was the textbfsequential design of the puzzles, particularly in Level 2. Sequential gameplay, in this context, refers to scenarios where the player must collect specific prerequisite items to unlock access to other necessary items in order to complete future tasks. For example, in Level 2, a player needs to obtain a red key to access the fire boot, which is then required to collect the pink chip. This layered dependency made it especially challenging to train **RL** agents, as it introduced even sparser rewards, as meaningful progress could only be made by completing a strict sequence of actions. Consequently, meeting the completion condition became significantly harder, making it difficult for **RL** agents to effectively learn the policy.

This ties to the type of cooperation that this game could support. The sequential design of the puzzle led to the cooperation to be a “effective path planning” to collect items. However, other types of cooperative playing could be imagined using diverse tile types. For example, movable dirt tile transforms hazardous tile into walkable tile. This tile could be utilized to make players cooperate in a different way, for example, one player moves the dirt tile and transforms the hazardous tile so that the other agent can proceed faster. Likewise, different types of tiles that supports relatively instant reward compared to sequential puzzles can be demonstrated.

5.3.3 Limited sample size

A key limitation of this study is the small sample size, which may have prevented statistically significant detection of the main and interaction effects of alignment mode. Despite this, observed trends and meaningful effect sizes suggest practically relevant associations that warrant further investigation with a larger participant pool.

5.4 Future Work

The future work should aim to mitigate the limitations identified in the section 5.3—specifically by enabling a true sense of merged decision-making mode and re-engineering the game environment to support a more approachable and efficient agent training pipeline.

Tuning Merged Mode Mechanism

As previously discussed, the current implementation of the merged mode needs refinement to truly represent agents’ diverse social abilities in the decision-making process. The key limitation lies in the final step of merged mode, where the human player is only given the binary choice to either accept or reject the agent’s suggestion. This can be improved by introducing a re-suggestion loop, in which the human player can counter the agent’s suggestion with an updated proposal. This cycle would continue until mutual agreement is reached—either the human adopts the agent’s plan or the agent aligns with the human’s plan through its internal model.

To fully realize this mechanism, future versions of the merged mode should incorporate the human’s re-suggestion into the agent’s next reasoning step. This may require the agent to infer the intentions behind human suggestions, a task well covered in the Theory of Mind (ToM), which was discussed in Chapter 2.

To provide a concrete example, the updated merged mode might unfold as follows:

1. The human proposes an assignment plan.
2. The agent runs its internal model and proposes an alternative.
 - (a) If the agent’s result is the same as the human’s, the agent accepts it directly and communication ends → *aligned mode*.
 - (b) If not, the agent proposes an alternative plan.
3. The human now has the choice to accept, reject, or re-suggest.
 - (a) If the human rejects the suggestion → *diverged mode*.
 - (b) If the human re-suggests, the human provide a modified plan.

4. The agent reasons about the human’s likely intent, updates its internal model, and outputs a revised plan.
5. This loop continues until the agent and human reach a shared plan $\rightarrow$ *merged mode*.
6. With given probability, agent rejects human’s plan $\rightarrow$ *diverged mode*.

Using this mechanism, we can capture the back-and-forth dynamics of merged communication, revealing the agent’s social capability of *merged alignment* in collective decision-making process.

Engineering Environment for Better **RL** Training

Supporting Partially Observable Grid Map: In original Tile World 2 game setting, the game feels more dynamic since the map is larger than what player can observe at a time. The maximum grid size of Tile World is 32 x 32, but players can only observe 9 x 9 tiles at once. Once the player moves to the edge, the camera moves to the direction of the player moves and then renders unobservable grid previously. This adds *exploration* sense of the game, because players cannot plan out the whole game from the start, rather they should solve the level by exploring.

This concept also aligns with the challenges of training AI agents. One of the key difficulties in reinforcement learning is balancing the amount of information available to the agent with the design of sparse reward signals that allow the agent to associate specific actions with outcomes. In my experiments, providing the agent with the full 13×13 grid as input made it difficult for the agent to learn meaningful action–state relationships. The abundance of information appeared to overwhelm the learning network. Conversely, when the agent was given only a local 5×5 grid, it successfully learned local navigation rules but struggled to explore beyond the immediate area—often remaining close to its starting position. As a final alternative, I experimented with combining both the full 13×13 grid and the local 5×5 grid. However, this also failed: the agent did not learn meaningful local associations, likely because the larger 13×13 input dominated and obscured the learning signal from the smaller local grid.

Through experimentation with developing **RL** agents, I found that implementing a **partially observable environment**, for both human player and agent may boost game experience for humans while also providing a more suitable setting for **RL** agents to learn meaningful state–action–outcome relationships.

Supporting Advanced Game Mechanics: In the future, Co-op Tile World needs to support

advanced game mechanics such as different types of monster and toggleable tiles like traps and clone buttons, that exist in original Tile World 2 [7] but not yet supported in Co-op Tile World.

Implementing different types of monsters will give researchers challenging environment to train agents that responds quickly in a complex environment since each monster has different movement style and speed. Also, the toggleable tiles are especially useful game mechanics in Co-op Tile World in that they facilitate cooperative game play. For example, the trap tile, when players step on it, they become trapped and cannot move until the trap is deactivated. However, the trapped object can be freed when the trap button is pressed and activated. Similarly, for clone tiles, they spawn a creature when activated. These toggling behavior naturally goes along with cooperative interaction. However, for the current version, these mechanisms are not yet applied, leaving room for future work.

5.5 Discussion

This thesis began with an intellectual aspiration: to develop machines capable of playing cooperative video games with humans. But why emulate the dynamics of playing co-op games with siblings and friends in interactions with machine partners? Because these interactions naturally produce a high resolution of social data and feedback—elements that are critical for understanding and evaluating cooperation.

The richness and fidelity of these social signals are particularly valuable, as they offer nuanced data for diagnosing the state of cooperation. Such social diagnostics are essential for promoting successful and sustained collaboration.

Below is an illustrative scenario demonstrating how the findings from this thesis could be applied in real-world settings:

Imagine a robot designed to play with children—a robotic companion marketed as a relief for parents who need a brief moment of rest from the demands of parenting. This robot can play board games, build LEGO structures, draw and paint, or engage in active games like hide-and-seek or tag.

To build such a robot, it is not sufficient for the robot to simply excel at the mechanics of each game or task. Cooperation with children requires more than just task execution—it requires the ability to guide, adapt, and communicate.

At the beginning, children may be unsure how to interact with the robot. To build trust and engagement, the robot could begin with simple, familiar games such as word chain games or basic LEGO tasks. Once a successful cooperative experience is established, the robot can gradually introduce more complex games like strategy-based board games or physically dynamic tasks like hide-and-seek.

However, the robot must also monitor and adapt to the child’s skill level. If it introduces games that are too difficult too quickly, the child may lose interest or become frustrated. Therefore, the robot should dynamically assess the child’s engagement and performance, adjusting task difficulty accordingly.

Furthermore, the robot’s manner of communication should be tailored to the child’s developmental level: being highly supportive and encouraging for younger or less experienced children, while allowing more open-ended, negotiable interactions with more capable children. This adaptive communication style helps sustain engagement.

In short, the success of such a robot depends not just on its individual ability to excel in performing specialized tasks, but also on how it builds trust, adapts to its partner, and communicates appropriately—echoing the findings from this research.

This scenario underscores the real-world need for empirically grounded social robots that can cooperate effectively with human partners by adjusting their behavior in response to social cues—and enabling humans to do the same in return.

Until now, significant progress has been made in developing machines capable of performing specific tasks. Benchmarks of such agents in games like Chess, Go, Atari, and StarCraft have demonstrated performances that exceed human capabilities [52]. These game environments have provided valuable platforms for AI researchers to assess task performance, compare algorithms, and accelerate the development of AI techniques.

Building on the proven utility of benchmarking environments in AI research, this study proposes a custom game environment specifically designed for real-time cooperation between human players and AI agents. Researchers can utilize the Co-op Tile World level editor to design customized cooperative tasks aligned with their specific research objectives. Once levels are configured, the environment offers integrated communication protocols, predefined game rules, and a rich set of interactive tile mechanics—providing a flexible and efficient foundation for studying human–AI cooperation.

Ultimately, Co-op Tile World offers an ideal testbed for conducting empirical research on machine behavior [52]. Through such empirical research, we gain insights into machine behav-

ior—which, at its core, is the manifestation of underlying algorithms—within the social contexts in which they operate [51]. More importantly, we can examine whether and how human interactions with these algorithms influence broader societal outcomes [52].

Co-op Tile World brings attention to the concept of *socially grounded cooperative interaction* in human-machine cooperation. Here, social grounding refers to the system’s ability to generate rich social data throughout the course of cooperation, enabling participants to tune their behavior to achieve better outcomes. As illustrated in the earlier example of the child caregiving robot, understanding the dynamic social interplay between algorithms and humans is essential for designing more adaptive and purposeful intelligent machines.

Moreover, high-fidelity social expressions and feedback mechanisms are especially important, as they not only promotes successful cooperation, but also for enhancing algorithmic transparency, establishing ethical thresholds, detecting risks, and ensuring safety in interaction.

However, beyond Co-op Tile World, I believe that developing effective benchmarking game mechanics integrated across diverse platforms—such as physical environments, robotic systems, and AR/VR interfaces—offers a powerful avenue for computational designers to contribute meaningfully to Cooperative AI (CAI) and Human–Robot Interaction (HRI) research.



Co-op Tile World Game Images



(a) Pressing the “OPTIME” button pauses the game, giving players time to assign tasks strategically.



(b) Players can click on collectible items in the UI to assign them to themselves or another player.

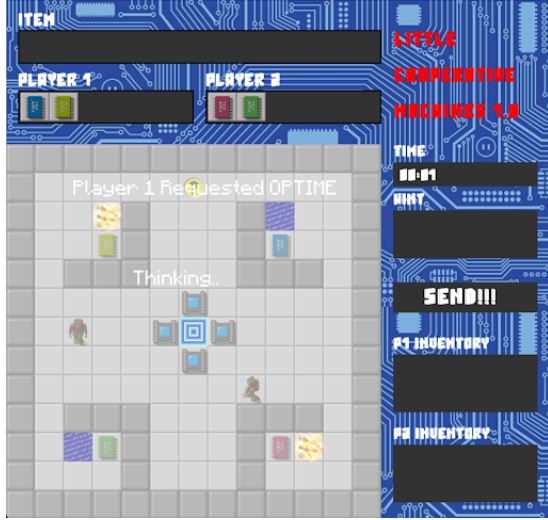


(c) Once assigned, the items appear in a separate UI box displaying each player's assigned items.



(d) However players can reject an assigned strategy by clicking on an item in their assignment box and selecting “Unassign item.” This introduces varying degrees of alignment in decision-making.

Figure A.1: Communication process



(a) Once human players finalize their decision, they click the send button on the right side of the panel. This triggers agent to run its internal model for strategic planning.



(b) **Aligned Mode** Agent directly accepts the suggestion.



(c) **Merged Mode** Agent resuggest new suggestion.



(d) **Diverged Mode** Agent directly rejects the suggestion.

Figure A.2: Demonstrating alignment mode via agent feedback

B

Reinforcement Learning Agent

Reinforcement Learning (RL) focuses on agent-based decision-making to maximize cumulative rewards within a given environment. Unlike supervised learning methods where models are trained on labeled datasets, RL agents must learn through interaction, guided only by sparse reward signals. [53] As noted by A.G. Barto, RL relies on two foundational components: search and memory. [54] Search involves trial and error to explore possible actions, while memory allows the system to retain and reuse information about what has worked well in past situations. Given a well-defined environment and reward function, RL can be a powerful framework for developing agents capable of aligning with intended goals—even learning subtle behaviors that may not have been explicitly designed.

However, RL also presents several challenges. Compared to supervised learning, it typically demands significantly more computational resources. Additionally, once an agent is trained in a specific environment, its learned policy often struggles to generalize to new or slightly altered environments.

To train RL agent on Level 1, the level was converted into a **Gymnasium** environment. Gymnasium is the actively maintained fork of OpenAI's Gym— a Python library for developing and testing RL algorithms [55], [56]. To create a custom Gym environment, the designer must configure the **observation** and **action** spaces, which define the possible inputs (actions taken by the agent) and outputs (observations returned by the environment) [56]. These spaces can be either discrete or continuous, depending on the task. In addition to these definitions, two core functions are required: the **reset** function, which initializes a new episode, and the **step** function, which

updates the environment state based on the agent’s action. The **step** function must compute the next observation, the reward signal, and whether the episode has ended. With these components in place, the Level 1 environment was implemented as a custom class, `GridWorldEnv`, as described in pseudocode [3].

Algorithm 3 `GridWorldEnv` Core Mechanisms

```

1: function INIT
2:   Define action space as 4 discrete actions (up, down, left, right)
3:   Define observation space as flattened grid of tile values
4: end function
5: function RESET
6:   Reset to initial grid configuration
7:   Reset step counter and collected chip set
8:   return initial observation
9: end function
10: function STEP(action)
11:   Move RL agent and evaluate resulting tile:
       • Hazard → end episode with penalty
       • Chip → collect and reward
       • Socket → unlock if all chips collected and reward
       • Exit → end episode with the biggest reward
12:   Apply small step penalty
13:   Move the other agent with sampled action
14:   return observation, reward, termination flags
15: end function
16: function __MOVE(position, action)
17:   Compute new coordinates from direction
18:   if within bounds and not wall then
19:     return new position
20:   else
21:     return original position
22:   end if
23: end function
24: function __GET__OBS
25:   Overlay agent positions on grid
26:   return flattened grid as observation
27: end function

```

After setting up the Gymnasium environment, the agent was trained using the Stable Baselines3 (SB3) library, which offers a collection of widely used reinforcement learning algorithms—including PPO, DQN, A2C, SAC, and TD3—implemented in PyTorch. [57] For this experiment, the agent was trained using the Proximal Policy Optimization (PPO) algorithm due to its performance across a wide range of [RL] tasks. Detailed parameter configuration is shown in the table [B.1].

The experiment began by training an agent for 2 million timesteps, which served as a baseline model for subsequent reward function tuning. Building upon this base agent, additional training

Table B.1: PPO Training Hyperparameters

Parameter	Value
Policy	MlpPolicy
Learning Rate	3×10^{-4}
Discount Factor (γ)	0.99
GAE Lambda (λ)	0.95
Steps per Update (n_steps)	2048
Batch Size	64
Epochs per Update	10
Clipping Range	0.2
Random Seed	Fixed (None)

sessions were conducted for 1 million timesteps each, specifically targeting performance on Level 1. Despite a total of 3 million timesteps of training, the agent ultimately failed to learn how to complete the level. Figure B.1 illustrates the training trends for both the base agent and the subsequent runs.

As shown in the figure, the agent’s mean reward consistently remained on the negative side, indicating unsuccessful training. Notably, a significant behavioral difference can be observed between the runs shown in Figure B.1b and Figure B.1c. In Figure B.1b, the agent exhibits a higher mean reward and longer episode length, suggesting that it learned to navigate around hazardous tiles, though without completing the level. In contrast, Figure B.1c shows both lower mean reward and shorter episode lengths. This decline resulted from modifying the reward function to assign a larger negative penalty per step, aimed at discouraging inefficient behaviors such as aimless wandering or repeatedly visiting the same locations. However, this adjustment backfired—the agent began stepping into hazardous tiles more frequently in an apparent attempt to avoid the accumulating negative reward.

Finally, Figure B.1d depicts run 6(green) where the maximum number of steps per episode was increased from 100 to 200. This change was intended to give the agent more time to learn the sequential logic of the level—collecting all chips and then reaching the exit. However, the results were nearly identical to those in Figure B.1c, run 5(magenta), indicating that extending the episode length alone was insufficient for improving learning outcomes.

Figure B.2 shows the resulting agent’s trajectories in visual way. We can observe for run 3 and run 5, agent’s last position is right to the pink hazardous tile, indicating the backfire triggered in the training session.

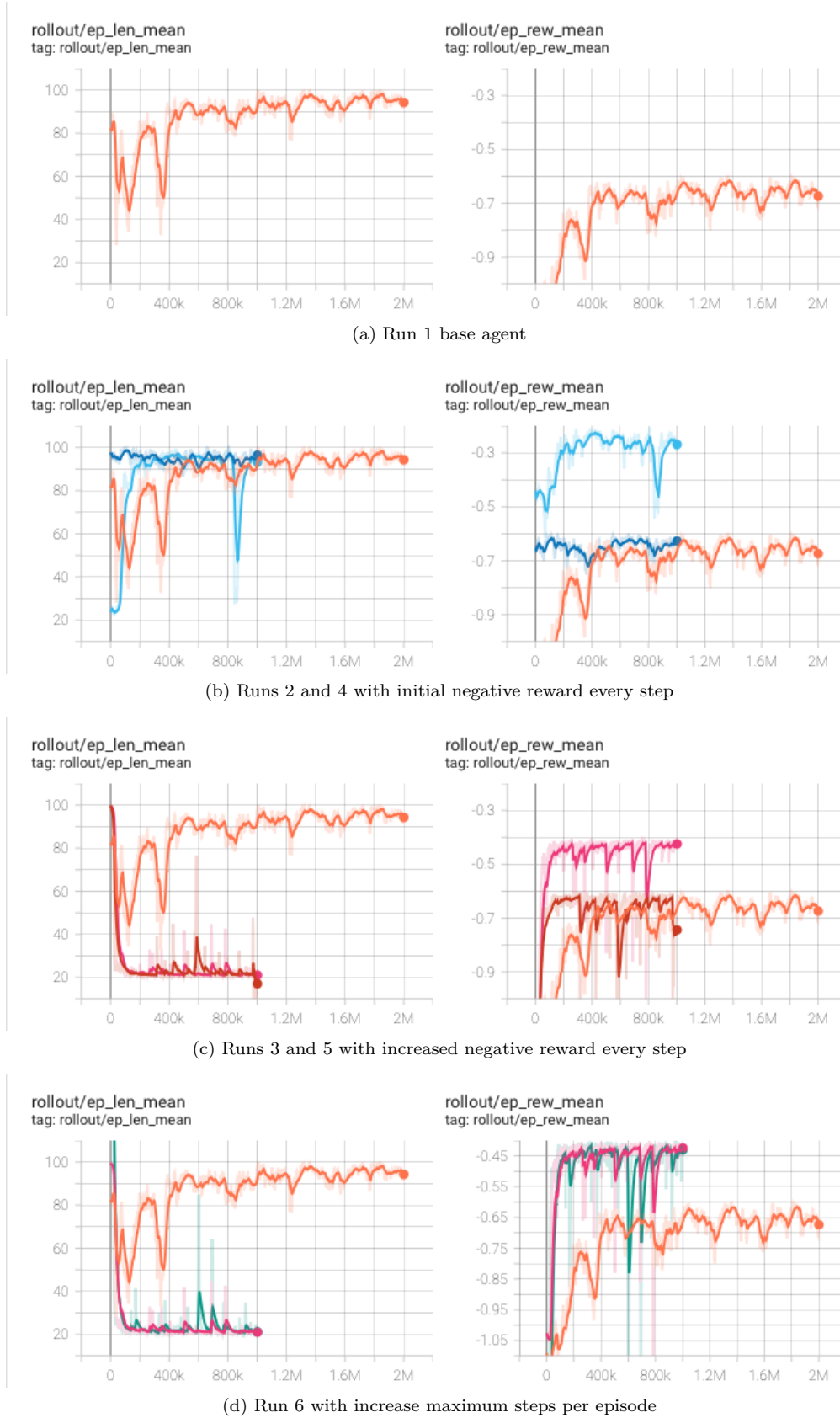
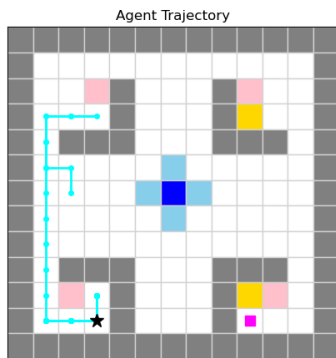
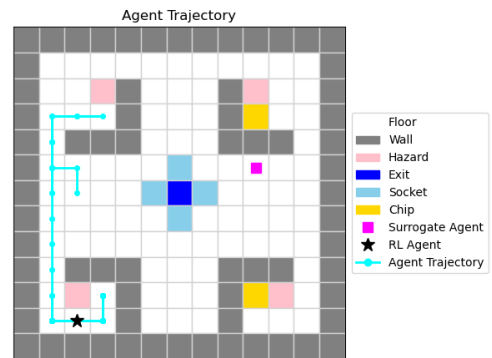


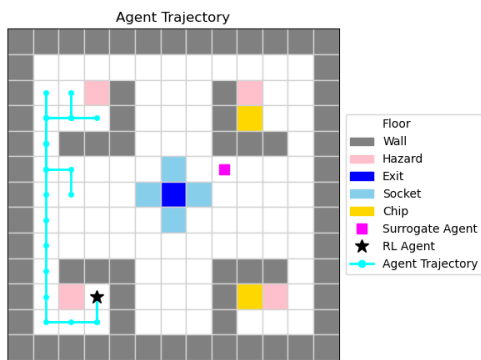
Figure B.1: Training progress across different experimental conditions.



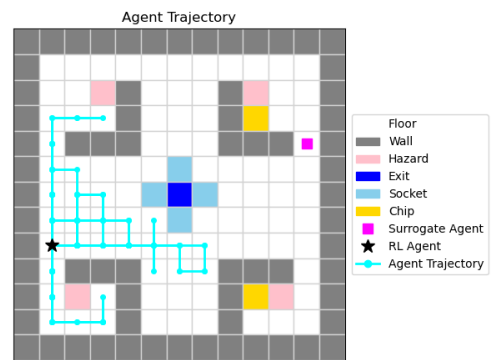
(a) Run 1 base agent trajectory



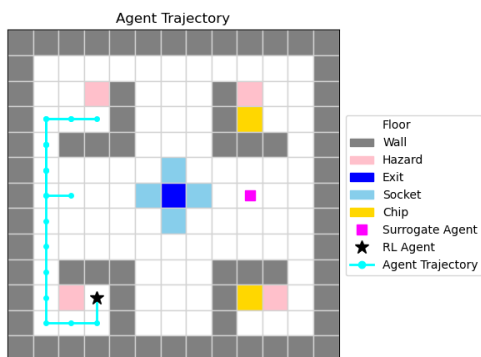
(b) Run 2 agent trajectory



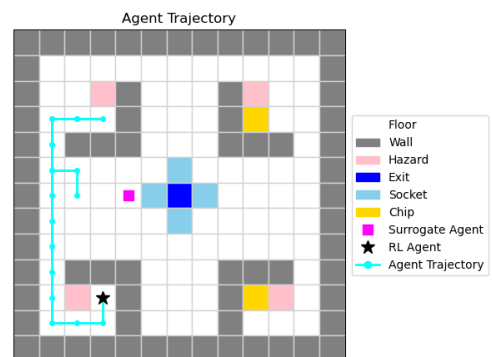
(c) Run 3 agent trajectory



(d) Run 4 agent trajectory



(e) Run 5 agent trajectory



(f) Run 6 agent trajectory

Figure B.2: Evaluation trajectories of trained agents across six different runs

C

User Experiment Raw Data

Co-op Tile World User Test Data Log

Participant	Level	Time Limit	Level Start Time	Level End Time	Play Duration	Level Cleared	Agent Alignment	P1 Ind Row	P2 Ind Row	Assignment Log	P1 Inventory	P2 Inventory	P1 Step Directions	P2 Step Directions	P1 Final Reward	P2 Final Reward
A	1	1	20 0:04	0:45	16:36	Yes	0	20	20	20 {FLOOR, 3, 3: 1, CHIP_YELLOW/CHIP_CHIP_GREEN/CHIP_1 {LEFT, 'UP, 'UP, 'UP {UP, 'DOWN, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	37.5
A	2	2	60 0:56	2:52	55.7	Yes	0	80	80	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
B	1	1	20 0:04	0:35	12:64	No	2	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
B	2	2	60 0:48	4:20	59:69	No	2	40	40	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
C	1	1	20 1:56	2:35	16:56	No	2	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
C	2	2	60 5:33	6:47	45:99	No	2	40	40	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
D	1	1	20 0:04	0:40	16:98	Yes	2	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
D	2	2	60 0:04	1:23	53:95	Yes	2	40	40	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
E	1	1	20 0:05	0:51	15:36	Yes	1	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
E	2	2	60 0:04	1:45	48:38	Yes	1	40	40	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
F	1	1	20 0:04	0:50	19:59	Yes	2	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
F	2	2	60 0:04	-64:04	No	-64:04	No	2	30	50 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	50 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
G	1	1	20 0:04	0:35	14:27	Yes	0	20	20	20 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	50 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
G	2	2	60 0:46	5:02	60:01	No	0	60	60	50 {FLOOR, 1, 1: 2, KEY_RED/KEY_GREEN/KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'LEFT, 'UP, '1	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
H	1	1	20 0:04	0:30	12:63	Yes	0	20	20	20 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	50 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
H	2	2	60 0:04	2:04	38.8	Yes	0	40	40	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
I	1	1	20 0:04	0:51	14:14	Yes	0	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
J	1	2	60 1:01	4:24	51:51	Yes	0	40	40	20 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
J	2	1	20 1:26	2:06	15:51	No	1	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
J	2	2	60 4:51	6:17	60:01	No	1	70	70	40 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
K	1	1	20 0:04	0:50	14:07	Yes	2	20	20	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
K	2	2	60 3:01	4:37	60	No	2	40	40	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
L	1	1	20 0:04	0:37	12:88	Yes	0	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_GREEN/CHIP_1 {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
L	2	2	60 0:04	2:56	51:84	Yes	0	40	40	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
M	1	1	20 1:26	1:57	14:59	Yes	1	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
M	2	2	60 0:04	2:03	60:02	No	1	40	40	50 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_GREEN/CHIP_1 {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
N	1	1	20 0:04	0:44	12:69	Yes	1	20	20	20 {FLOOR, 3, 3: 1, CHIP_YELLOW/CHIP_CHIP_GREEN/CHIP_1 {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
N	2	2	60 5:46	6:39	41:82	Yes	1	40	40	70 {FLOOR, 3, 0: 2, KEY_RED/BOOT_FRI KEY_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
O	1	1	20 0:04	0:47	12:78	Yes	1	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
O	2	2	60 0:04	4:21	42:11	Yes	1	40	40	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
P	1	1	20 1:22	1:45	12:62	Yes	2	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
P	2	2	60 6:58	8:47	60:02	No	2	40	40	60 {FLOOR, 3, 10: 1, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
Q	1	1	20 0:04	0:57	12:59	Yes	1	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
Q	2	2	60 0:04	1:51	53:23	Yes	1	40	40	70 {FLOOR, 3, 7: 1, KEY_RED/BOOT_FRI KEY_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	
R	1	1	20 0:04	0:54	13:47	Yes	0	20	20	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	30 {FLOOR, 12, 2: 1, KEY_RED/BOOT_FRI KEY_BLUE/BOOT_W {LEFT, 'DOWN, 'DO {LEFT, 'DOWN, 'DO	20 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150
R	2	2	60 0:04	1:40	28:55	Yes	0	30	30	80 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	20 {FLOOR, 9, 3: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {RIGHT, 'UP, 'UP, '1	70 {FLOOR, 7, 9: 2, KEY_RED/BOOT_FRI KEY_GREEN/KEY_BI {LEFT, 'DOWN, 'DO {DOWN, 'RIGHT, 'D	70 {FLOOR, 9, 10: 2, CHIP_YELLOW/CHIP_CHIP_PINK/CHIP_GRI {LEFT, 'UP, 'UP, 'UP {UP, 'RIGHT, 'RIGHT, 'D	37.5	150	

Figure C.1: Raw experimental data collected from the Co-op Tile World user test

Bibliography

- [1] K. Arulkumaran, A. Cully, and J. Togelius, “AlphaStar: An evolutionary computation perspective,” in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO ’19, New York, NY, USA: Association for Computing Machinery, Jul. 2019, pp. 314–315, ISBN: 978-1-4503-6748-6. DOI: [10.1145/3319619.3321894](https://doi.org/10.1145/3319619.3321894). [Online]. Available: <https://dl.acm.org/doi/10.1145/3319619.3321894> (visited on 05/02/2025).
- [2] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, “The Arcade Learning Environment: An Evaluation Platform for General Agents,” *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, Jun. 2013, arXiv:1207.4708 [cs], ISSN: 1076-9757. DOI: [10.1613/jair.3912](https://doi.org/10.1613/jair.3912). [Online]. Available: <http://arxiv.org/abs/1207.4708> (visited on 05/04/2025).
- [3] M. Campbell, A. J. Hoane, and F.-h. Hsu, “Deep Blue,” *Artificial Intelligence*, vol. 134, no. 1, pp. 57–83, Jan. 2002, ISSN: 0004-3702. DOI: [10.1016/S0004-3702\(01\)00129-1](https://doi.org/10.1016/S0004-3702(01)00129-1). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0004370201001291> (visited on 05/02/2025).
- [4] A. Dafoe, Y. Bachrach, G. Hadfield, E. Horvitz, K. Larson, and T. Graepel, “Cooperative AI: Machines must learn to find common ground,” en, *Nature*, vol. 593, no. 7857, pp. 33–36, May 2021, ISSN: 0028-0836, 1476-4687. DOI: [10.1038/d41586-021-01170-0](https://doi.org/10.1038/d41586-021-01170-0). [Online]. Available: <https://www.nature.com/articles/d41586-021-01170-0> (visited on 02/06/2025).
- [5] J. K. Gupta, M. Egorov, and M. Kochenderfer, “Cooperative Multi-agent Control Using Deep Reinforcement Learning,” en, in *Autonomous Agents and Multiagent Systems*, G. Sukthankar and J. A. Rodriguez-Aguilar, Eds., Cham: Springer International Publishing, 2017, pp. 66–83, ISBN: 978-3-319-71682-4. DOI: [10.1007/978-3-319-71682-4_5](https://doi.org/10.1007/978-3-319-71682-4_5).
- [6] J. C. R. Licklider, “Man-Computer Symbiosis,” *IRE Transactions on Human Factors in Electronics*, vol. HFE-1, no. 1, pp. 4–11, Mar. 1960, ISSN: 2168-2836. DOI: [10.1109/THFE2.1960.4503259](https://doi.org/10.1109/THFE2.1960.4503259). [Online]. Available: <https://ieeexplore.ieee.org/document/4503259/authors> (visited on 05/02/2025).

-
- [7] B. Raiter, M. Shanbhag, and M. Schmidt, *Tile World 2 - An open-source reimplementaion of the game "Chip's Challenge"*, Aug. 2024. [Online]. Available: <https://tw2.bitbusters.club> (visited on 05/04/2025).
- [8] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of Go with deep neural networks and tree search," en, *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016, Publisher: Nature Publishing Group, ISSN: 1476-4687. DOI: [10.1038/nature16961](https://doi.org/10.1038/nature16961). [Online]. Available: <https://www.nature.com/articles/nature16961> (visited on 05/02/2025).
- [9] J. K. Terry, B. Black, N. Grammel, *et al.*, *PettingZoo: Gym for Multi-Agent Reinforcement Learning*, arXiv:2009.14471 [cs], Oct. 2021. DOI: [10.48550/arXiv.2009.14471](https://doi.org/10.48550/arXiv.2009.14471). [Online]. Available: <http://arxiv.org/abs/2009.14471> (visited on 02/11/2025).
- [10] C. L. Baker, R. Saxe, and J. B. Tenenbaum, "Action understanding as inverse planning," en, *Cognition*, vol. 113, no. 3, pp. 329–349, Dec. 2009, ISSN: 00100277. DOI: [10.1016/j.cognition.2009.07.005](https://doi.org/10.1016/j.cognition.2009.07.005). [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0010027709001607> (visited on 04/10/2025).
- [11] M. E. Bratman, D. J. Israel, and M. E. Pollack, "Plans and resource-bounded practical reasoning," en, *Computational Intelligence*, vol. 4, no. 3, pp. 349–355, 1988, _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8640.1988.tb00284.x>, ISSN: 1467-8640. DOI: [10.1111/j.1467-8640.1988.tb00284.x](https://doi.org/10.1111/j.1467-8640.1988.tb00284.x). [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8640.1988.tb00284.x> (visited on 03/29/2025).
- [12] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, no. 1, pp. 139–159, Jan. 1991, ISSN: 0004-3702. DOI: [10.1016/0004-3702\(91\)90053-M](https://doi.org/10.1016/0004-3702(91)90053-M). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/000437029190053M> (visited on 02/26/2025).
- [13] G. Chalkiadakis, E. Elkind, and M. Wooldridge, "Cooperative Game Theory: Basic Concepts and Computational Challenges," *IEEE Intelligent Systems*, vol. 27, no. 3, pp. 86–90, May 2012, Conference Name: IEEE Intelligent Systems, ISSN: 1941-1294. DOI: [10.1109/MIS.2012.47](https://doi.org/10.1109/MIS.2012.47). [Online]. Available: <https://ieeexplore.ieee.org/document/6237463/?arnumber=6237463> (visited on 03/20/2025).
- [14] M. G. Collins, I. Juvina, and K. A. Gluck, "Cognitive Model of Trust Dynamics Predicts Human Behavior within and between Two Games of Strategic Interaction with Computerized Confederate Agents," English, *Frontiers in Psychology*, vol. 7, Feb. 2016, Publisher: Frontiers, ISSN: 1664-1078. DOI: [10.3389/fpsyg.2016.00049](https://doi.org/10.3389/fpsyg.2016.00049). [Online]. Available: <https://www.frontiersin.org/article/10.3389/fpsyg.2016.00049> (visited on 03/20/2025).
-

- frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2016.00049/full (visited on 03/30/2025).
- [15] A. C. Costa, K. Bijlsma-Frankema, and B. Jong, *The Role of Social Capital on Trust Development and Dynamics: Implications for Cooperation, Monitoring and Team Performance*, en, SSRN Scholarly Paper, Rochester, NY, 2009. [Online]. Available: <https://papers.ssrn.com/abstract=2050913> (visited on 05/06/2025).
- [16] J. W. Crandall, M. Oudah, Tennom, *et al.*, “Cooperating with machines,” en, *Nature Communications*, vol. 9, no. 1, p. 233, Jan. 2018, ISSN: 2041-1723. DOI: [10.1038/s41467-017-02597-8](https://doi.org/10.1038/s41467-017-02597-8). [Online]. Available: <https://www.nature.com/articles/s41467-017-02597-8> (visited on 02/06/2025).
- [17] A. Dafoe, E. Hughes, Y. Bachrach, *et al.*, *Open Problems in Cooperative AI*, en, arXiv:2012.08630 [cs], Dec. 2020. DOI: [10.48550/arXiv.2012.08630](https://doi.org/10.48550/arXiv.2012.08630). [Online]. Available: <http://arxiv.org/abs/2012.08630> (visited on 02/03/2025).
- [18] A. M. Evans and J. I. Krueger, “The Psychology (and Economics) of Trust,” en, *Social and Personality Psychology Compass*, vol. 3, no. 6, pp. 1003–1017, 2009, _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1751-9004.2009.00232.x>, ISSN: 1751-9004. DOI: [10.1111/j.1751-9004.2009.00232.x](https://doi.org/10.1111/j.1751-9004.2009.00232.x). [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1751-9004.2009.00232.x> (visited on 03/30/2025).
- [19] R. Gockley, A. Bruce, J. Forlizzi, *et al.*, “Designing robots for long-term social interaction,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ISSN: 2153-0866, Aug. 2005, pp. 1338–1343. DOI: [10.1109/IR05.2005.1545303](https://doi.org/10.1109/IR05.2005.1545303). [Online]. Available: <https://ieeexplore.ieee.org/document/1545303> (visited on 03/30/2025).
- [20] M. A. Goodrich and A. C. Schultz, “Human-Robot Interaction: A Survey,” en, *Foundations and Trends® in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007, ISSN: 1551-3955, 1551-3963. DOI: [10.1561/11000000005](https://doi.org/10.1561/11000000005). [Online]. Available: <http://www.nowpublishers.com/article/Details/HCI-005> (visited on 01/15/2025).
- [21] D. J. Gunkel, “Ars Ex Machina : Rethinking Responsibility in the Age of Creative Machines,” in *Human-Machine Communication: Rethinking Communication, Technology, and Ourselves*, ser. Digital Formations, vol. 117, New York: Peter Lang Verlag, 2018, pp. 221–236, ISBN: 978-1-4331-4252-9. [Online]. Available: https://gunkelweb.com/articles/gunkel_computational-creativity_preprint.pdf (visited on 03/30/2025).

-
- [22] A. L. Guzman and S. C. Lewis, "Artificial intelligence and communication: A Human–Machine Communication research agenda," EN, *New Media & Society*, vol. 22, no. 1, pp. 70–86, Jan. 2020, Publisher: SAGE Publications, ISSN: 1461-4448. DOI: [10.1177/1461444819858691](https://doi.org/10.1177/1461444819858691). [Online]. Available: <https://doi.org/10.1177/1461444819858691> (visited on 03/29/2025).
- [23] P. A. Hancock, D. R. Billings, K. E. Schaefer, J. Y. C. Chen, E. J. de Visser, and R. Parasuraman, "A meta-analysis of factors affecting trust in human-robot interaction," eng, *Human Factors*, vol. 53, no. 5, pp. 517–527, Oct. 2011, ISSN: 0018-7208. DOI: [10.1177/0018720811417254](https://doi.org/10.1177/0018720811417254).
- [24] J. Henrich and M. Muthukrishna, "The Origins and Psychology of Human Cooperation," en, *Annual Review of Psychology*, vol. 72, no. Volume 72, 2021, pp. 207–240, Jan. 2021, Publisher: Annual Reviews, ISSN: 0066-4308, 1545-2085. DOI: [10.1146/annurev-psych-081920-042106](https://doi.org/10.1146/annurev-psych-081920-042106). [Online]. Available: <https://www.annualreviews.org/content/journals/10.1146/annurev-psych-081920-042106> (visited on 02/09/2025).
- [25] S. Herse, J. Vitale, B. Johnston, and M.-A. Williams, "Using Trust to Determine User Decision Making & Task Outcome During a Human-Agent Collaborative Task," in *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '21, New York, NY, USA: Association for Computing Machinery, Mar. 2021, pp. 73–82, ISBN: 978-1-4503-8289-2. DOI: [10.1145/3434073.3444673](https://doi.org/10.1145/3434073.3444673). [Online]. Available: <https://doi.org/10.1145/3434073.3444673> (visited on 03/30/2025).
- [26] G. A. Kelly, *The psychology of personal constructs. Vol. 1. A theory of personality. Vol. 2. Clinical diagnosis and psychotherapy* (The psychology of personal constructs. Vol. 1. A theory of personality. Vol. 2. Clinical diagnosis and psychotherapy). Oxford, England: W. W. Norton, 1955, Pages: xxviii, 1218.
- [27] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, *Building Machines That Learn and Think Like People*, en, arXiv:1604.00289 [cs], Nov. 2016. DOI: [10.48550/arXiv.1604.00289](https://doi.org/10.48550/arXiv.1604.00289). [Online]. Available: <http://arxiv.org/abs/1604.00289> (visited on 01/19/2025).
- [28] J. D. Lee and K. A. See, "Trust in Automation: Designing for Appropriate Reliance," EN, *Human Factors*, vol. 46, no. 1, pp. 50–80, Mar. 2004, Publisher: SAGE Publications Inc, ISSN: 0018-7208. DOI: [10.1518/hfes.46.1.50_30392](https://doi.org/10.1518/hfes.46.1.50_30392). [Online]. Available: <https://journals.sagepub.com/action/showAbstract> (visited on 03/30/2025).

-
- [29] S. Legg and M. Hutter, “Universal Intelligence: A Definition of Machine Intelligence,” en, *Minds and Machines*, vol. 17, no. 4, pp. 391–444, Dec. 2007, ISSN: 1572-8641. DOI: [10.1007/s11023-007-9079-x](https://doi.org/10.1007/s11023-007-9079-x). [Online]. Available: <https://doi.org/10.1007/s11023-007-9079-x> (visited on 02/09/2025).
- [30] N. Luhmann, *Trust and Power : two works*, eng. Chichester ; New York : Wiley, 1979, ISBN: 978-0-471-99758-0. [Online]. Available: <http://archive.org/details/trustpowertwowor0000luhm> (visited on 05/06/2025).
- [31] B. F. Malle and J. Knobe, “The Distinction between Desire and Intention: A Folk-Conceptual Analysis,” English, in *Intentions and Intentionality*, Cambridge, MA: MIT Press, Apr. 2001, pp. 45–68, ISBN: 978-0-262-13386-9. [Online]. Available: <https://direct.mit.edu/books/edited-volume/2643/Intentions-and-IntentionalityFoundations-of-Social> (visited on 11/05/2024).
- [32] A. M. Andrew, “Intelligent Systems: Architecture, Design, and Control,” English, *Kybernetes*, vol. 31, no. 5/6, pp. 926–928, 2002, Num Pages: 3 Place: London, United Kingdom Publisher: Emerald Group Publishing Limited, ISSN: 0368492X. DOI: [10.1108/k.2002.06731fae.006](https://www.proquest.com/docview/213907873/abstract/1116CF65C12B4563PQ/1). [Online]. Available: <https://www.proquest.com/docview/213907873/abstract/1116CF65C12B4563PQ/1> (visited on 02/09/2025).
- [33] J. Michael and E. Pacherie, “On Commitments and Other Uncertainty Reduction Tools in Joint Action,” en, *Journal of Social Ontology*, vol. 1, no. 1, pp. 89–120, 2015, Number: 1, ISSN: 2196-9663. [Online]. Available: <https://journalofsocialontology.org/index.php/jso/article/view/6897> (visited on 03/30/2025).
- [34] J. Michael and A. Salice, “The Sense of Commitment in Human–Robot Interaction,” *International Journal of Social Robotics*, vol. 9, no. 5, pp. 755–763, 2017, ISSN: 1875-4791. DOI: [10.1007/s12369-016-0376-5](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5698395/). [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5698395/> (visited on 03/30/2025).
- [35] K. Okamura and S. Yamada, “Adaptive trust calibration for human-AI collaboration,” eng, *PloS One*, vol. 15, no. 2, e0229132, 2020, ISSN: 1932-6203. DOI: [10.1371/journal.pone.0229132](https://doi.org/10.1371/journal.pone.0229132).
- [36] M. J. Osborne and A. Rubinstein, *A course in game theory*, en. Cambridge, Mass: MIT Press, 1994, ISBN: 0-262-65040-1.
- [37] D. Premack and G. Woodruff, “Does the chimpanzee have a theory of mind?” en, *Behavioral and Brain Sciences*, vol. 1, no. 4, pp. 515–526, Dec. 1978, ISSN: 1469-1825, 0140-525X. DOI:

- [10.1017/S0140525X00076512](https://doi.org/10.1017/S0140525X00076512). [Online]. Available: <https://www.cambridge.org/core/journals/behavioral-and-brain-sciences/article/does-the-chimpanzee-have-a-theory-of-mind/1E96B02CD9850016B7C93BC6D2FEF1D0> (visited on 03/29/2025).
- [38] N. Raihani, *The Social Instinct: How Cooperation Shaped the World*, English. New York: St. Martin's Press, Aug. 2021, ISBN: 978-1-250-26282-0.
- [39] J. L. Sachs, U. G. Mueller, T. P. Wilcox, and J. J. Bull, "The Evolution of Cooperation," *The Quarterly Review of Biology*, vol. 79, no. 2, pp. 135–160, Jun. 2004, Publisher: The University of Chicago Press, ISSN: 0033-5770. DOI: [10.1086/383541](https://doi.org/10.1086/383541). [Online]. Available: <https://www-journals-uchicago-edu.us1.proxy.openathens.net/doi/full/10.1086/383541> (visited on 02/09/2025).
- [40] M. Schwaninger, "Intelligent organizations: An integrative framework," en, *Systems Research and Behavioral Science*, vol. 18, no. 2, pp. 137–158, 2001, _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/sres.408>, ISSN: 1099-1743. DOI: [10.1002/sres.408](https://doi.org/10.1002/sres.408). [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/sres.408> (visited on 02/09/2025).
- [41] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots That Use Language," en, *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. Volume 3, 2020, pp. 25–55, May 2020, Publisher: Annual Reviews, ISSN: 2573-5144. DOI: [10.1146/annurev-control-101119-071628](https://doi.org/10.1146/annurev-control-101119-071628). [Online]. Available: <https://www.annualreviews.org/content/journals/10.1146/annurev-control-101119-071628> (visited on 03/30/2025).
- [42] Y. Wang, W. Kinsner, and D. Zhang, "Contemporary Cybernetics and Its Facets of Cognitive Informatics and Computational Intelligence," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 4, pp. 823–833, Aug. 2009, Conference Name: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), ISSN: 1941-0492. DOI: [10.1109/TSMCB.2009.2013721](https://doi.org/10.1109/TSMCB.2009.2013721). [Online]. Available: <https://ieeexplore.ieee.org/document/4810135> (visited on 02/26/2025).
- [43] William Spaniel, *Game Theory 101: The Complete Textbook*, eng. CreateSpace Independent Publishing Platform, Sep. 2011, ISBN: 978-1-4927-2815-3. [Online]. Available: <http://archive.org/details/gametheory101com0000will> (visited on 04/10/2025).
- [44] H. Dubberly, P. Pangaro, and U. Haque, "What is interaction? are there different types?" *interactions*, vol. 16, no. 1, pp. 69–75, Jan. 2009, ISSN: 1072-5520. DOI: [10.1145/1456202.1456220](https://doi.org/10.1145/1456202.1456220). [Online]. Available: <https://dl.acm.org/doi/10.1145/1456202.1456220> (visited on 03/30/2025).

-
- [45] *How to perform a Mixed ANOVA in SPSS Statistics*. [Online]. Available: <https://statistics.laerd.com/spss-tutorials/mixed-anova-using-spss-statistics.php> (visited on 04/08/2025).
- [46] P. Bhandari, *What is Effect Size and Why Does It Matter? (Examples)*, en-US, Dec. 2020. [Online]. Available: <https://www.scribbr.com/statistics/effect-size/> (visited on 05/05/2025).
- [47] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed. New York: Routledge, May 2013, ISBN: 978-0-203-77158-7. DOI: [10.4324/9780203771587](https://doi.org/10.4324/9780203771587).
- [48] E. Fehr and S. Gächter, “Altruistic punishment in humans,” en, *Nature*, vol. 415, no. 6868, pp. 137–140, Jan. 2002, Publisher: Nature Publishing Group, ISSN: 1476-4687. DOI: [10.1038/415137a](https://doi.org/10.1038/415137a). [Online]. Available: <https://www.nature.com/articles/415137a> (visited on 04/15/2025).
- [49] G. E. Gignac and E. T. Szodorai, “Effect size guidelines for individual differences researchers,” *Personality and Individual Differences*, vol. 102, pp. 74–78, Nov. 2016, ISSN: 0191-8869. DOI: [10.1016/j.paid.2016.06.069](https://doi.org/10.1016/j.paid.2016.06.069) [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0191886916308194> (visited on 05/05/2025).
- [50] D. Kallogjeri and J. F. Piccirillo, “A Simple Guide to Effect Size Measures,” eng, *JAMA otolaryngology– head & neck surgery*, vol. 149, no. 5, pp. 447–451, May 2023, ISSN: 2168-619X. DOI: [10.1001/jamaoto.2023.0159](https://doi.org/10.1001/jamaoto.2023.0159).
- [51] R. Milner, “A Modal Characterisation of Observable Machine-Behaviour,” in *Proceedings of the 6th Colloquium on Trees in Algebra and Programming*, ser. CAAP ’81, Berlin, Heidelberg: Springer-Verlag, Mar. 1981, pp. 25–34, ISBN: 978-3-540-10828-3. (visited on 04/21/2025).
- [52] I. Rahwan, M. Cebrian, N. Obradovich, *et al.*, “Machine behaviour,” en, *Nature*, vol. 568, no. 7753, pp. 477–486, Apr. 2019, Publisher: Nature Publishing Group, ISSN: 1476-4687. DOI: [10.1038/s41586-019-1138-y](https://doi.org/10.1038/s41586-019-1138-y). [Online]. Available: <https://www.nature.com/articles/s41586-019-1138-y> (visited on 04/21/2025).
- [53] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, *Playing Atari with Deep Reinforcement Learning*, arXiv:1312.5602 [cs], Dec. 2013. DOI: [10.48550/arXiv.1312.5602](https://doi.org/10.48550/arXiv.1312.5602). [Online]. Available: <http://arxiv.org/abs/1312.5602> (visited on 05/04/2025).
- [54] The Artificial Intelligence Channel, *A History of Reinforcement Learning - Prof. A.G. Barto*, Jul. 2018. [Online]. Available: <https://www.youtube.com/watch?v=ul6B2oFPNDM> (visited on 05/04/2025).

-
- [55] G. Brockman, V. Cheung, L. Pettersson, *et al.*, *OpenAI Gym*, arXiv:1606.01540 [cs], Jun. 2016. DOI: [10.48550/arXiv.1606.01540](https://doi.org/10.48550/arXiv.1606.01540). [Online]. Available: <http://arxiv.org/abs/1606.01540> (visited on 05/04/2025).
- [56] M. Towers, A. Kwiatkowski, J. K. Terry, *et al.*, *Gymnasium: A Standard Interface for Reinforcement Learning Environments*, original-date: 2022-09-08T01:58:05Z, May 2025. [Online]. Available: <https://github.com/Farama-Foundation/Gymnasium> (visited on 05/04/2025).
- [57] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations,” *J. Mach. Learn. Res.*, vol. 22, no. 1, 268:12348–268:12355, Jan. 2021, ISSN: 1532-4435.