Heterogeneity in Human-LLM Multiagent Systems

Alexandra Maria Marcu

TU Delft

Heterogeneity in Human-LLM Multiagent Systems

by

Alexandra Maria Marcu

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday, June 29, 2026.

Project duration:	November, 2025 – June, 2026	
Thesis committee:	Dr. P.K. Murukannaiah	TU Delft, supervisor
	Dr. B.J.W. Dudzik	TU Delft

TU Delft

Preface

This thesis marks the conclusion of my Master's in Computer Science, as well as my academic journey at TU Delft, which started with the Bachelor's in Computer Science and Engineering in 2020. The last years spent at Delft have been a deeply rewarding and challenging experience.

I am really grateful to my supervisor, Dr. Pradeep Murukannaiah, for his guidance and support throughout this research. Our weekly discussions were incredibly helpful, always providing clarity or bouncing off ideas when I needed it the most. I am especially thankful for his early advice to pursue a research topic I would genuinely enjoy working on, even though that meant not having an already identified research gap and structured proposal. The months spent on the literature review to identify a research gap allowed me to stay curious.

Alexandra Maria Marcu
Delft, June 2026

Contents

1 Introduction

Recent advances in large language models (LLMs) are driving a shift in how intelligent systems are built, moving them beyond the generation of text and towards autonomous LLM-based agents that reason, plan, and act [1]. Increasingly, these agents are being deployed in multi-agent systems (MAS), where multiple agents communicate, coordinate, and divide work to solve problems that exceed the capabilities of a single agent [2–5]. However, the agents in such a team are not necessarily alike. They may differ in the language model that powers them and in the persona or role they adopt. Such differences have the potential to make a team stronger or weaker.

This thesis explores the role of agent heterogeneity within MAS architectures. By investigating how structural differences among agents influence collaborative task outcomes, this research aims to provide empirical insights that can inform future MAS development. Furthermore, recognizing that humans will increasingly work alongside these systems, this thesis analyzes existing simulation environments to formulate a set of desirable characteristics for testbeds capable of supporting both multi-agent and human-agent collaboration. This chapter sets out the context in which the thesis is situated and introduces how heterogeneity has so far been treated in MAS and LLM-based MAS literature. From this context, a set of research questions is formulated to define the scope of the thesis's contribution.

Why Heterogeneity Matters in LLM-based Multiagent Systems

Beyond text generation, question answering, summarization, and translation, LLMs have demonstrated capabilities in reasoning, planning, code generation, and multi-step decision-making [6–8]. By combining a language model with specialized modules for memory, planning, and tool use, LLM-based agents can solve complex tasks across fields such as software engineering [9, 10], healthcare [11, 12], scientific research [13], and education [14]. As task complexity grows, a solution is to scale from a single agent to a multi-agent system in which agents divide work, share resources, and coordinate knowledge.

However, as tasks grow increasingly complex, simply adding more agents reaches a point of diminishing returns. If the agents share the same architecture, training, and reasoning strategies, they tend to produce redundant outputs and reinforce each other's errors rather than correct them [15, 16]. Drawing on human organizational research, one response is to introduce diversity into the team, giving it a broader range of perspectives and a greater capacity to catch mistakes [17]. Research on human teams, however, warns that diversity is no guarantee of success. While heterogeneous teams can increase creativity through varied perspectives, unmanaged differences can trigger conflicts [18, 19]. The benefits of diversity are highly conditional, and there is no reason to assume they transfer automatically to teams of LLM agents.

Recent work explores heterogeneity in LLM-MAS by varying models [20, 21], personas [22, 23], and roles [10]. However, one key limitation in this body of work motivates this thesis. Prior work treats heterogeneity as a deliberate design choice to improve performance, rather than as something unavoidable in real-world settings. For example, deployed agents vary in hardware (in the embodied case) and models, or drift apart as each evolves based on its own experience. While current LLM research focuses on ways to introduce cognitive heterogeneity, the effect of physical capability heterogeneity remains underexplored. However, even for the case of cognitive heterogeneity, there is limited research that investigates its impact on team performance. Furthermore, as humans are increasingly part of agent teams, how the composition (i.e., heterogeneous or not)

of the agent team shapes the human's experience is an increasingly relevant dimension. While studying this human experience requires environments built to support human-agent teams, current frameworks frequently lack this.

This thesis addresses these gaps in two ways. First, building upon the limitations identified in current environments, it introduces a new framework and environment to support human-agent integration. Second, to address the empirical gap regarding agent composition, it studies heterogeneity both as an unavoidable property and as a deliberate design choice. It investigates this through two dimensions. First, it explores **physical capability heterogeneity**, which naturally arises in embodied systems due to differing hardware or resource constraints. Second, it introduces a novel approach to **cognitive heterogeneity** based on diverse reasoning and planning strategies, distinguishing it from prior work that focused strictly on models, roles, or personas. Both dimensions are studied within a single *modular agent architecture* that keeps all components fixed except the one under investigation, enabling controlled comparisons that isolate the effect of each form of heterogeneity.

Research Questions

This thesis aims to explore the impact of agent heterogeneity on LLM-based multi-agent teams. Specifically, it examines how heterogeneity affects performance and how its members collaborate. To evaluate these dynamics, the thesis proposes a new environment for testing heterogeneity, informed by a survey of existing environments. The research is driven by one main question and three sub-questions, which are answered in separate chapters.

High-level Research Question

How does agent heterogeneity affect the task performance and collaborative process of an LLM-based multi-agent system?

Characterizing Environments for Human-Agent Collaboration

The environment is a central and essential component of any agent system. It determines what agents can perceive, which actions are available to them, and how information is distributed. Despite this, environments used in human-LLM MAS have not been examined systematically: existing surveys categorize them by application domain or by interaction type, treating the environment as a passive component rather than as a design dimension in its own right, and leaving researchers without a principled basis for selecting one. This raises the first research question:

Research Question 1

What are the desirable characteristics of environments for studying human-LLM multi-agent systems?

Chapter 2 answers this question by examining a selection of LLM-MAS environments, both agent-only and with human teammates. The scope is deliberately limited to simulated embodied environments, since studying physical heterogeneity requires agents to differ in embodied capabilities such as speed, strength, or perception range. Thus, the simulated embodied settings are the only ones where such differences have meaningful task consequences. While the requirement of supporting human participation was established *a priori* as the motivating goal of this study, the *desiderata* was not defined in advance. Rather, the desirable characteristics emerged from the survey itself, through observing what properties agent-only environments have and what current mixed-team environments lack.

Designing the HuLA-MAS Framework and Environment

The analysis in chapter 2 shows that no existing environment supports all the identified desirable characteristics: multi-agent and multi-human participation, configurable collaboration, support for multi-modal interaction, LLM-compatible dynamics, separation of concerns between the agents and environment, and support for heterogeneity. These gaps motivate the design of a new environment:

Research Question 2

How can the identified limitations of existing environments inform the design of an improved research environment?

Chapter 3 answers this question by presenting **HuLA-MAS**, a **modular framework** for LLM-based heterogeneous embodied collaboration that allows a human to join the team in different roles. The chapter connects each identified characteristic to a concrete component of the implementation and explains how that component meets it. The resulting research environment and framework make the controlled study of heterogeneity in the next chapter possible.

The Impact of Team Heterogeneity on Collaboration

With an environment and framework that enable the controlled study of heterogeneity, chapter 4 focuses on the question that motivated this thesis. In multi-agent systems, the default is often a homogeneous team of identical models, yet it remains unclear whether heterogeneity helps or hinders. Establishing whether it changes both what a team accomplishes and how it collaborates directly informs how these systems should be composed:

Research Question 3

How does heterogeneity in physical and cognitive capabilities affect task and process performance in a collaborative task?

Scope of the Thesis

To maintain a focused scope and interpretable insights, this thesis explicitly limits the types of heterogeneity introduced, the environment in which they are studied, and the inclusion of human participants in the empirical study.

Regarding the **types of heterogeneity introduced**, while prior work often relies on diverse language models and role-based personas, this research isolates two forms of heterogeneity derived from a modular agent architecture. First, it examines heterogeneity in *physical capabilities* by varying agents' competence across different dimensions. Second, it introduces *cognitive* heterogeneity by varying the prompts and strategies used in the Planning and Reasoning modules. To ensure controlled experiments, all other components, such as memory and communication, as well as the agent architecture, remain fixed.

In terms of the **environment used for evaluation**, many environments and tasks support the research of LLM-based MAS. However, meaningful physical capability heterogeneity only matters in an environment where these limitations impact task feasibility. Therefore, this thesis is restricted to *simulated embodied environments*. Real-world embodied environments are similarly outside this scope.

Finally, regarding the **empirical focus and human participation**, while this thesis recognizes the importance of mixed human-agent teams and explicitly designs the HuLA-MAS framework to support human participation, *empirical evaluation of the human experience remains out of scope.* Executing a complete human user study is outside the current resource constraints and is reserved for future work. Consequently, the empirical evaluations in this thesis focus entirely on autonomous, agent-only configurations.

Main Contributions

- A set of desirable characteristics for environments used to study human-LLM multi-agent systems, accompanied by a survey evaluating how well existing testbeds satisfy them.
- A Search-and-Rescue environment that meets these characteristics, supporting heterogeneous team composition, interdependencies that promote collaboration, and multi-human, multi-agent teams.
- The HuLA-MAS framework to study heterogeneity in multi-agent teams.
- An evaluation of homogeneous versus heterogeneous LLM agent teams that measures both task performance and the collaborative process.

2

The Current Landscape of Environments for Human-LLM Multiagent Systems

Research Question 1

What are the *desirable characteristics* of environments for studying human-LLM multi-agent systems?

The concept of multi-agent systems has long been a foundational area of decentralised artificial intelligence research, focusing on how multiple autonomous entities interact, coordinate, and divide work to solve complex problems [24]. Today, LLMs are increasingly used to enhance the reasoning and decision-making capabilities of agent systems. By leveraging language models as their cognitive backbone, agents can now interact with complex environments, call external tools, and communicate in natural language. Consequently, there is a rapidly growing body of research focused on multi-agent systems in which LLM-based agents cooperate, collaborate, or compete to solve complex problems [2, 4].

Full autonomy, however, is premature and potentially unsafe in many applications [25, 26]. LLMs can hallucinate, misalign with intended goals, and amplify safety and ethical risks when operating without oversight [27, 28]. This motivates the study of human-LLM MAS, where humans provide information, feedback, and corrective control to compensate for agents' limitations. This offers a more trustworthy and robust path to deployment than full autonomy. Yet incorporating humans into agent teams is not simply an engineering decision. It raises questions about how collaboration should be structured, what the human's role should be, and how the system should be designed to support it. Answering these questions empirically requires environments that can actually express them.

The environment is a central component of any agent system. Without it, an agent can neither sense nor act [29]. It determines what agents can perceive, what actions are available, and how information is distributed. It also determines what role a human can play within the team, and which aspects of collaboration can be studied at all. A turn-based, fully observable environment cannot expose challenges of real-time coordination or information asymmetry. This is also relevant for embodied agents, which are attracting growing research interest across domains such as search and rescue [30, 31], logistics [32], and healthcare [33, 34]. In these settings, agents differ not only in their reasoning but in their physical capabilities, such as speed, strength, dexterity, and perception range. Before such systems are deployed in the real world, simulation offers a controlled and safe setting in which to study how these capability differences affect task execution and division of labour.

Despite its importance, environments used in human-LLM MAS have not been analysed. Existing surveys either categorise environments by application domain [3] or analyse human-agent interaction through the lens of feedback types and communication modes [35]. While useful for a general taxonomy, this classification does not reveal gaps or new design opportunities. An analysis of approximately 110 LLM-MAS papers shows that most environments are designed around the assumption that agents are LLMs communicating via natural language, making them LLM-centric by construction and unsuitable for studying other team configurations

[36]. Therefore, researchers are left without a principled basis for environment selection, and key dimensions of human-agent teamwork might be understudied because the environments used cannot express them.

Addressing this gap is necessary for the empirical work of this thesis. To study how physical and cognitive heterogeneity affects team performance requires an environment where these differences actually matter. Without knowing the characteristics of a suitable testbed, it is difficult to evaluate heterogeneous agents properly. Furthermore, identifying these characteristics benefits the broader research community. It provides a clear guide for researchers to select the right environments and highlights the features that current platforms are missing.

To study physical heterogeneity, agents must have different physical capabilities. Because of this, the scope of this chapter is strictly limited to *simulated embodied environments*. These are the only settings where differences in speed, strength, or perception impact the task outcome.

This chapter identifies the desirable characteristics by analysing a selection of existing LLM-MAS environments. This includes both agent-only environments and those that support human participation. The study compares these environments to determine their common characteristics and identify missing capabilities. While the requirement of supporting human participation was established *a priori* as the main motivation, the specific characteristics were not defined in advance. Instead, they emerged directly from the survey by analysing what mixed-team or agent-only environments have in common and what they lack.

2.1. Background

2.1.1. Collaborative AI and Human-Agent Teaming

Despite recent advancements in LLMs and MAS, purely autonomous teams of agents cannot yet be reliably and safely deployed in many real-world tasks [25, 26], due to the current limitations of the LLMs. Beyond hallucination, LLM-based agents fail to reliably recall or retrieve information, misjudge their own progress and outcomes, pursue logically unsound or infeasible strategies, and output actions with missing or incorrect parameters [15]. In multi-agent settings these individual errors propagate and compound across agents, making this a primary bottleneck for reliability and producing failures in the system [16]. These challenges motivate the need for human participation in the system. For this reason, there is a need to study LLM-based human-agent systems, which integrate human feedback and control into the system to overcome agents' limitations.

Understanding what makes human-agent collaboration work requires more than studying the agents in isolation. In LLM-based human-agent systems, humans interact with the agents and provide extra information, feedback, and control. These are built on the idea that the strengths of both humans and agents can be combined to yield a better-performing, reliable, and safe system. Zou et al. [35] create a taxonomy of LLM-based HAS, focusing on the feedback, interaction type, orchestration paradigm, and communication. However, it focuses on all human-agent teams, not only the ones where the human collaborates with multiple agents.

Decades of research in human-agent teaming, human-computer interaction, and collaborative AI offer a theoretical foundation for what an effective team needs. The Coactive Design framework [37] identifies three properties to enable interdependence between humans and agents: *observability*, *directability*, and *predictability*. The observability dimension involves making relevant aspects, such as status, knowledge, and environment, observable to the other participants. Directability refers to the ability to direct the behaviour of other team members, as well as to be directed by them. Predictability means that an agent's actions should be sufficiently predictable such that other members can reliably take them into account to formulate their own plans.

2.1.2. The Environment in Multiagent Systems

The definition of an environment has evolved in the literature on single and multi-agent systems. Early work describes it as an external world in which agents act, where they are deployed, and with which they interact [38, 39]. It is also described as a container in which a set of laws determines the effects of actions [40], or as all the elements that are external to the agents. Other work views the environment as including both external entities and the processes and principles that influence agents' existence and communication, and as a structuring entity for the MAS [29]. More recent work treats the environment as a design abstraction with responsibilities such as structuring the system, hosting resources, maintaining dynamics, limiting access, and enforcing rules [41].

Russell and Norvig [38] describe environments using properties such as observability, dynamics, temporal structure, and outcome uncertainty. In multi-agent settings, additional dimensions become relevant: how

agents coordinate, whether agents are heterogeneous in capability or role, and what responsibilities the environment has for mediating the team's interaction. Moreover, when human participants are present, the characteristics of what the environment allows humans to do also become relevant. However, these human-centred dimensions have received little attention in the existing MAS literature.

2.2. Dimensions for Characterizing Multiagent Environments

Before comparing the existing environments to formulate the final desiderata, the set of features on which to base the analysis must be defined. Classical MAS identifies properties such as observability, dynamics, temporal structure, and outcome uncertainty. These features are widely used, but comparing the environments on them alone does not give the full picture: relevant details are lost, and elements that make these environments distinct from one another might not be captured.

Many of the surveyed environments are proposed as variations over existing MAS environments, aimed at improving the evaluation of LLM-based agents. To motivate their design, several papers introducing a new benchmark include tables that compare their proposed environment with a limited set of prior environments along a set of dimensions the authors consider important. While these tables cover only a few environments, often mixing LLM-MAS, traditional MAS, and human-LLM MAS, they offer a useful starting point for the current analysis. Surveying multiple papers that introduce new environments for LLM-MAS reveals that some features recur across nearly all of them, while others appear only in specific papers. This section summarizes these features and selects the set on which the present analysis is conducted. By doing this, it aims to capture what the research community treats as the characteristics worth having in an environment for studying LLM-MAS and, more specifically, human-LLM MAS. The analysis is restricted to environments designed to evaluate collaboration. For this reason, competitive environments, such as BattleAgentBench [42], TextStarCraft II [43], and the Minecraft-based testbed PillagerBench [44] were excluded from the analysis.

2.2.1. General Environment Characteristics from Literature

This section discusses the dimensions selected for the analysis. They were selected from literature on LLM-MAS environments based on two criteria. (1) They recur across the majority of the surveyed literature, and (2) they describe properties of the environment itself rather than the specific task or evaluation it supports.

Because this survey focuses only on embodied environments, several features are *constant* across all systems and cannot be used to compare them. For example, *spatial structure* is essential to embodiment, so it is always present. Moreover, *object interaction* naturally occurs when interacting in a simulated physical setting. Finally, the strict enforcement of *action validity*, where the environment limits and checks which actions an agent is allowed to take [41], is built into all these systems. Therefore, these features act as *inclusion criteria* rather than variables for comparison. For completeness, they are defined but not analyzed further.

The following dimensions are kept for the analysis: *communication*, *collaboration enforcement*, *modality*, *team composition*, *task horizon*, and *generalization*. The *dynamics* and *observability* dimensions from classical MAS are also added to this set, as they can highlight variation across embodied environments. Heterogeneity was added to this list because it is the central topic of this thesis, even though it is not discussed frequently. Moreover, the features excluded from this study are: *control topology* (centralized vs. decentralized), *constraint types* [45, 46], support for *tool calls* [47, 48], *task dependencies*, and evaluation type (i.e., focus on performance metrics or collaboration) [49]. Table 2.1 summarizes all dimensions considered, together with their definitions, origins, and the surveyed papers that explicitly address them.

2.2.2. Human-Environment Characteristics

The dimensions discussed until now are sufficient for comparing agent-only environments. However, since support for human inclusion was identified as a desirable characteristic of environments, two other dimensions inspired by the Coactive design framework (see subsection 2.1.1) have been added to capture what the environment affords a human participant: *directability* and *observability*. *Observability* concerns the degree to which the environment makes its state and agent behavior visible to the human. High observability allows the human to track team progress, understand what the agents are doing, and identify when to intervene. Without observability, humans can only react to outcomes rather than actively participate in the collaborative process.

Directability concerns the degree to which the environment provides mechanisms for humans to influence agent behavior and to what extent. A low-directability environment limits human input to specifying high-level goals only. A high-directability environment supports intervention at any point, such as providing real-time feedback or redirecting individual agents mid-execution.

Table 2.1: **Dimensions used to characterize and compare the embodied multi-agent environments surveyed in this study**. For each dimension, the definition used as well as the papers that explicitly mention it are specified. Dimensions marked with † are constant across all embodied environments and serve as inclusion criteria rather than variables for comparison.

Dimension	**Definition**	**Origin / References**
Dynamics	Environment updates its state at any point, independently of agent actions.	[38]
Observability	Degree to which agents perceive the world state: full vs. partial.	[38]
Spatial structure†	Physical or topological layout constraining actions.	[38]
Action validity†	Hard enforcement (environment rejects illegal actions) vs. soft enforcement.	[41]
Object interaction irreversibility†	Agents can interact with the environment, and their actions are irreversible.	[47, 48]
Communication	Channel (NL, implicit) and the cost associated with it.	[47, 49–54]
Collab. enforcement	Whether teamwork is strictly required via interdependencies or only incentivised.	[45, 46, 49, 52]
Modality	Type of observation data provided to the agent (e.g., text, visual, multimodal).	[45, 47, 50, 53, 55, 56]
Team composition	Total number of human and artificial agents the environment supports.	[45, 47, 48, 50–54]
Task horizon	Temporal length, ranging from short coordinated steps to sustained planning.	[45, 46, 50, 55–57]
Generalization	Procedural generation and number of task variants.	[45, 47–49, 52, 55–57]
Heterogeneity	Whether the environment supports different physical capabilities or roles.	[55]
Human-Environment Characteristics		
Observability	Visibility of the environment's internal state to the human participant.	Coactive Design
Directability	Degree of human influence and control over agent behaviour.	Coactive Design

The third dimension of the Coactive design framework, *predictability*, is intentionally excluded. Predictability refers to the capacity to anticipate other agents' future actions, but it is not a property that the environment can support. The environment can only make agent states, plans, and reasoning observable and transparent to the human. Whether the observability enables a human to anticipate future behavior depends more on the agents. Predictability is therefore not treated as an independent dimension but as a property that sufficient observability can provide.

2.3. The Landscape of LLM-MAS Embodied Environments

The previous section established the set of dimensions to evaluate environments. Table 2.2 applies them to the selected embodied environments. This analysis is primarily concerned with environments that support human-agent collaboration. However, a curated set of agent-only environments is also included, not as an exhaustive survey, but because they illustrate design choices that are informative for understanding what the human-inclusive environments lack.

Table 2.2: **Comparison of environments supporting human-LLM MAS and agent-only MAS**. The upper part contains environments that explicitly support human participation; the lower part contains agent-only environments. The evaluation dimensions include Dynamics, Observability (Obs), Modality, Communication (Comm), Task Horizon, Generalization (Gen), and Collaborative Enforcement (Collab). †Natural language generated via slot filling rather than free form. ^{H}: Human inclusion is technically supported by the platform but not studied or mentioned explicitly in the paper.

Environment	Dynamics	Obs.	Modality	Comm.	Task Horizon	Gen.	Collab.
PARTNR [45]	×	Both	MM	NL	✓	✓	✓
TDW-MAT [50]	✓	Part	MM	NL + cost	✓	×	×
C-WAH [50]	×	Part	MM	NL + cost	✓	×	×
VirtualHome-Social [51]	×	Part	MM	NL	✓	×	×
CuisineWorld [48]	×	Full	Text	×	×	×	×
Overcooked-MToM [54]	✓	Full	Text	NL	×	×	×
RoCoBench [52]	✓	Part	Text	NL	×	×	Part
IGLU [53]	×	Full	MM	NL†	×	×	×
Minecraft [48]	✓	Part	Text	×	✓	✓	×
Collab-Overcooked [49]	✓	Part	Text	NL	×	×	✓
Robotouille [57]	✓	Part	Text	×	✓	✓	×
TeamCraft [47]	✓	Part	MM	×	✓	✓	Part
MineCollab [56]	✓	Part	MM	NL	✓	✓	✓
Crew-Wildfire ^{H} [55]	✓	Part	MM	NL	✓	✓	✓
VillagerBench ^{H} [46]	✓	Part	Text	×	✓	✓	×

By analysing the selected environments through these dimensions, several patterns and limitations have been identified. First, the domain and task coverage is concentrated around cooking (the Overcooked family, CuisineWorld, Robotouille), household (PARTNR, TDW-MAT, C-WAH, VirtualHome-Social), and Minecraft (IGLU, TeamCraft, VillagerBench, MineCollab).

Dynamics, Observability, and Multimodality

The table shows that partial observability and text-based perception are the most commonly used settings. However, multimodal input is also supported in several environments, especially the ones that support human inclusion. Humans require a visual interface to participate in the MAS. This correlation might be a side effect of human inclusion, which also benefits the MAS. This is because a text-only environment limits the types of agents that can be tested in the environments. Moreover, over-reliance on text or symbolic data can discard spatial information and limit the performance of the team and the realism of the system.

Task Horizon and Generalizability

Horizon and generalizability are partly coupled to the type of task the environment proposes. Long-horizon tasks appear in household and Minecraft environments. The real-time Overcooked variants are shorter, given the limited steps that most of the recipes usually require. While some variants (Collab-Overcooked) propose longer and more complex recipes to add complexity, they are not comparable to the household or

Minecraft tasks. Notably, environments that value longer task horizons also implement generalization. The pattern seen in human-inclusive environments likely reflects the practical cost of human participation. These environments do not need many variants and procedural generation, since that would also require multiple expensive experiments with human participation.

Collaboration Enforcement and Communication
Among all features used in this analysis, the most discriminating one is enforced collaboration. More agent-only environments enforce collaboration. Collab-Overcooked isolates resources into separate sub-environments and gives only one agent the recipe, so neither can finish alone. MineCollab uses unique inventories and asymmetric knowledge. The human-inclusive environments mostly incentivize rather than enforce cooperation. The exception is PARTNR, which enforces division of labor only on its heterogeneity. Regarding communication, more human-agent environments support natural language communication between the agents than agent-only environments. This is mainly because natural language can be used by all participants to communicate.

Heterogeneity
Heterogeneity is rarely treated explicitly. Crew-Wildfire is the only environment that treats heterogeneous agents as a requirement, adding support for four agent types, and PARTNR includes two heterogeneous agents without highlighting it. Environments such as VirtualHome-Social or the Overcooked variants add no support for heterogeneity, perhaps because the task itself does not require it. Given the embodied nature of these environments, it is surprising that only a few include support for heterogeneity in physical capabilities.

Human Inclusion: Roles and Capabilities
Table 2.3 characterizes how much the environment allows humans to do or what information they get. Team composition shows that every environment supports exactly one human, and most pair that human with a single agent, with only CuisineWorld, Minecraft, and RoCoBench having larger agent teams. No environment integrates multiple humans and agents into a single team. Second, directability is the weakest of the human dimensions: only VirtualHome-Social, RoCoBench, and IGLU make the human able to direct the agent by providing feedback and natural language commands. The dominant role is a human who can observe the scene and the agent's actions but cannot heavily influence their teammate's decisions and actions. Observability is comparatively well supported, with four environments giving the human full visibility, yet this rarely combines with full control. Thus, in these embodied settings, humans get the role of a teammate with limited capabilities.

Table 2.3: **Human-environment dimensions.** Team composition records the maximum number of humans and LLM-based agents the simulation can support. These evaluate how much support for human interaction the environment provides. Degrees of support are denoted as fully supported (●), partially supported (◑), or limited support (○).

Environment	Observability	Directability	Team composition
PARTNR [45]	◑	○	1 H/1 A
TDW-MAT [50]	●	○	1 H/1 A
C-WAH [50]	●	○	1 H/1 A
VirtualHome-Social [51]	●	●	1 H/1 A
Overcooked-MToM [54]	◑	○	1 H/1 A
CuisineWorld [48]	◑	○	1 H/>4 A
RoCoBench [52]	●	●	1 H/2–3 A
IGLU [53]	◑	●	1 H/1 A
Minecraft [48]	◑	◑	1 H/>4 A

2.4. Desiderata for Human-LLM MAS Environments

This section summarizes the findings from the environment analysis and proposes desiderata for human-LLM MAS environments:

Multi-human, multi-agent participation. The system must support teams composed of multiple agents and multiple human participants. Humans should be able to play different roles in the team and interact through the graphical interface directly, rather than only through a text prompt.

Configurable collaboration. The environment should support configurable interdependencies that require agents to divide work and coordinate, with adjustable task difficulty.

Support for multimodality. The environment should support input modalities beyond plain text. Agents should be able to receive observations as structured visual representations (e.g., a rendered view of the grid) in addition to text descriptions of the world state.

LLM-compatible dynamics. The environment should accommodate the latency of LLM inference without freezing the world or reducing interaction to a rigid turn-based system, so that human participants experience responsive, concurrent teamwork.

Separation of agents and environment. The environment is the sole authority on what actions are possible and what happens when they are attempted. The agents decide what to attempt and why.

Support for heterogeneity. The task should provide a realistic testbed for studying heterogeneity. Heterogeneity should be a manipulable variable rather than an implementation side-effect.

2.5. Conclusion

This chapter examined a selection of simulated embodied environments used in LLM-MAS research, including both agent-only and human-inclusive settings. The scope was deliberately limited to embodied environments, as they are the only setting in which meaningful physical capability differences between agents can be expressed and studied. Rather than defining requirements in advance, this analysis first surveyed the literature to identify recurring properties across environment designs. A subset of these dimensions was selected based on frequency and relevance, and the environments were analyzed and compared using them.

The final desiderata emerged from the gaps and patterns revealed during this evaluation. They include LLM-compatible dynamics, enforced collaboration, multimodal support, teams composed of multiple artificial and human agents, and explicit support for heterogeneity. Several of these characteristics were first observed in agent-only environments and are proposed as desirable additions to environments that also support human participants. When it comes to human support in the system, the simulated environments should allow appropriate or configurable levels of observability and directability, depending on the task being tested.

The analysis also revealed how human-LLM and agent-only environments differ in intent. Agent-only environments are usually built to stress-test agent capabilities, with long-horizon tasks with dependencies and procedural generation. Human-inclusive environments tend to study the dynamics of collaboration between teammates, and the number of possible experiments is limited by the participants. This can partially explain why human-inclusive environments are less likely to enforce collaboration, long tasks, or procedural generation.

While existing environments support the study of collaboration between humans and LLM-based agents, this chapter revealed that many environments are designed around the idea that an LLM-based agent will be included in the environment and accommodate their weaknesses.

Limitations of this analysis are acknowledged. First, the set of environments surveyed is not exhaustive and mainly covers the most popular or recent ones. The dimensions used in the analysis were selected based on their recurrence in the literature, which means that characteristics that are less mentioned but potentially important may have been overlooked. Finally, the desiderata proposed should be treated as informed design recommendations rather than a complete set of requirements.

3

HuLA-MAS: Framework and Environment Development

> Research Question 2
>
> How can the identified limitations of existing environments inform the design of an improved research environment?

This chapter directly uses the desiderata formulated in chapter 2 to inform the creation of a new research framework, **HuLA-MAS** (**H**eterogeneo**u**s **L**LM **A**gent **M**ulti-**A**gent **S**ystem) [1]. These characteristics include support for multi-human, multi-agent teams, configurable collaboration, support for multimodal interaction, LLM-compatible dynamics, separation of agents and environment, and support for heterogeneity.

The remainder of this chapter details the design and implementation of the **HuLA-MAS** framework and its Search-and-Rescue environment. By designing this platform around the desiderata established in the previous chapter, HuLA-MAS introduces a new environment that supports both autonomous LLM-MAS as well as human-LLM MAS research. This framework is used to conduct the empirical evaluation of agent heterogeneity presented in chapter 4.

3.1. HuLA-MAS: A Modular Framework for LLM-Based Heterogeneous Embodied Collaboration

HuLA-MAS is a Search-and-Rescue (SaR) simulation framework designed to study heterogeneous collaboration between LLM-based agents and, optionally, human participants. It was built to help answer the main research question of this thesis regarding the influence of heterogeneity on LLM-MAS. Moreover, its design was informed by the gaps identified in Chapter 2: existing platforms that support heterogeneity do not include humans, and those that include humans do not enforce collaboration through physical capability constraints.

The framework is organized as four connected components, as shown in Figure 3.1. The **environment** defines what is physically possible and how agents' actions affect the simulated world. The **agent infrastructure** bridges the environment and agents, ensuring the simulation is unaware of whether an LLM, a rule-based system, or a human drives agents. The **LLM-based agents** are composed of modules for planning, reasoning, memory, and profile. The **team coordination** layer manages the coordination of joint actions through which agents can work together. Humans can optionally enter the simulation, either as an embodied teammate sharing the agent infrastructure with the artificial agents or as a supervisor that can exchange messages with the agents but cannot perform actions in the environment.

(1) Environment (MATRX). The 2D grid world and game loop, extended from the existing SaR MATRX [58] simulation. The environment enforces action validity and capability constraints: it rejects actions that exceed an agent's capabilities, creating the physical interdependencies that make collaboration necessary (**support for heterogeneity and configurable collaboration**). Moreover, world layout and

[1] The code is publicly available at: https://github.com/alemarcu23/HuLA-MAS

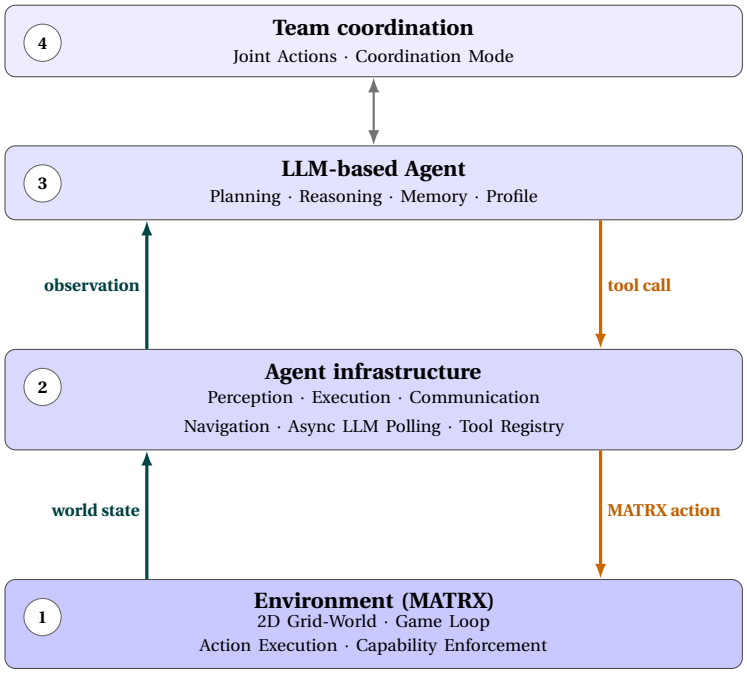


Figure 3.1: **The layered architecture of the HuLA-MAS framework.** (1) The **Environment** enforces physical capabilities, constraints, and game-loop dynamics; (2) The **Agent Infrastructure** acts as a bridge between the environment and the LLM-based agents; (3) The **LLM-based Agents** have modules to support cognitive and physical heterogeneity; and (4) The **Team Coordination** layer manages coordination mode and joint actions. **Human support** is integrated either as embodied physical teammates within the grid world or as supervisors.

difficulty are fully configurable. Teams of multiple LLM-based agents and human participants are supported (**multi-agent, multi-human participation**).

(2) Agent infrastructure. It acts as the bridge between the environment and the agent, keeping the simulation LLM-agnostic (**separation of agents and environment**). It also handles asynchronous LLM polling so that inference latency does not block the simulation (**LLM-compatible dynamics**).

(3) LLM-based agent. An agent has planning, reasoning, memory, and profile modules. The planning and reasoning modules support multiple strategies that can be selected independently, enabling cognitive heterogeneity across the team. Moreover, the profile module sets the physical capabilities (**support for heterogeneity**).

(4) Team coordination layer. Provides the messaging infrastructure, a structured help protocol for joint actions, and a shared memory through which agents coordinate (**configurable collaboration**). An optional central planner is available as an alternative to fully decentralized coordination.

Finally, another capability of the system is **human support**, but it cannot be treated as a separate component since it is connected to all four existing components and is not a new one. Humans can join the simulation in two ways: as an embodied teammate subject to the same physics as the agents or as an external supervisor communicating through natural-language commands (**human support, multi-human, multi-agent interaction**).

3.2. Search-and-Rescue Environment

The environment used is a 2D, partially observable, grid-based search-and-rescue simulation built on the latest version of MATRX software [2] [58]. The grid world consists of areas surrounded by walls, victims of different severity levels, obstacles that can block access to the areas, a drop zone, and agents (Figure 3.2). The objective is for the rescue team to locate all injured victims and carry them to the drop zone. Victims are scored by severity, with critically injured victims rewarding 6 points to the team and mildly injured victims only 3 points.

[2] MATRX Search-and-rescue environment, version 2.3.3

Healthy victims should not be rescued in this scenario, and they score zero points. A higher severity level also requires a greater capability of the agents that can carry the victims. There are three types of obstacles: trees, stones, and rocks. The removal of obstacles such as stones or rocks requires agents with greater strength or a joint removal by two agents.

Each agent perceives only a small radius of the world around itself, so the map must be explored to reach the goal. The world is configured declaratively: the room positions, the victim positions and severities, the obstacle positions and types, and the drop zone location can all be changed, allowing difficulty and collaboration requirements to be varied across experiments.

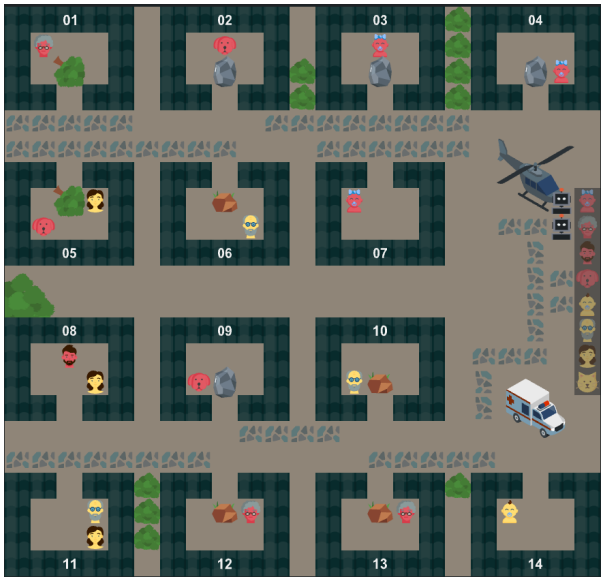


Figure 3.2: **The Search-and-Rescue environment.** In this configuration of the grid world, there are 14 houses (areas) in which victims with different severity levels can be found (red = critical, yellow = mild). The entrances to the areas can be blocked by various obstacles (trees, rocks, stones). The agents start the game on the right side of the map, next to the drop zone (represented by the grey area). The location of the victims and obstacles, as well as their type, can be changed.

Before the simulation starts, the **game loop** builds the world from a configuration file and registers each agent. At every tick, the environment snapshots the world and presents the same frozen state to all agents. Agents are polled sequentially to collect their actions, but since every agent reads the same snapshot, no agent gains an advantage from being polled later. Thus, the decisions are effectively simultaneous, unlike a turn-based scheme in which a later-polled agent would already see an earlier agent's action. Actions that reflect physical work (e.g., moving, removing an obstacle, carrying a victim) span multiple ticks, during which the agent is not re-polled. The simulation ends when all injured victims are rescued or a predefined time has passed.

3.3. LLM-Based Agent

The agent architecture takes inspiration from the multi-agent framework of **CoELA** (Cooperative Embodied Language Agent) [50] with decentralized control, inter-agent communication, and a modular cognitive pipeline. This architecture is extended with a reasoning module inspired by **AgentSquare** [59], which decouples high-level planning from low-level action selection. To introduce cognitive heterogeneity, each module supports interchangeable prompting strategies. These strategies can be configured independently for each agent, allowing team members to utilize distinct cognitive approaches (**support for heterogeneity**). Appendix A.3 details the available strategy variants.

The four core cognitive modules comprising each agent, namely `Planning`, `Reasoning`, `Memory`, and `Profile`, are illustrated in Figure 3.3. In the original CoELA architecture, `communication` is a cognitive module responsible for writing the content and interpreting messages. In HuLA-MAS, this responsibility is split differently: outbound communication is handled by the reasoning module, since sending a message is treated as an action like any other. Inbound message processing is handled by a Communication component that follows the same pattern as Perception. Together with perception and execution, it forms part of the agent infrastructure (section 3.4) and is shown in Figure 3.3 to clarify the data flow but does not constitute a cognitive

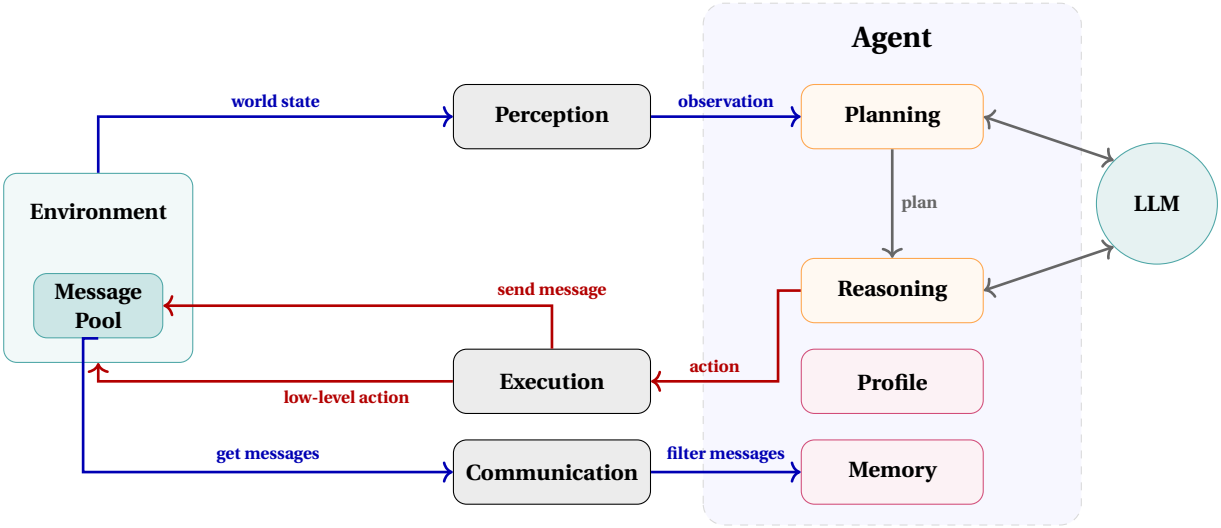


Figure 3.3: **HuLA-MAS agent architecture and data flow.** The dashed box encloses the four cognitive modules of the agent: `Planning`, `Reasoning`, `Memory`, and `Profile`. Three additional components are shown outside this boundary, as they belong to the agent infrastructure rather than to the cognitive layer: `Perception`, which filters the raw world state into the local observation seen by the agent; `Execution`, which translates the tool call produced by Reasoning into a low-level environment action; and `Communication`, which ingests messages from the shared Message Pool and filters, converts and summarizes them into structured context for the Planning and Reasoning modules. The Message Pool is rendered inside the Environment box because it is maintained by the environment, not by any individual agent. Within the cognitive layer, `Planning` generates the next atomic sub-task using information from Memory and Profile; `Reasoning` translates that sub-task into a single tool call, which is either a physical action dispatched through Execution or a message dispatched through the Message Pool; `Memory` maintains the agent's decentralised world beliefs, action history, and message history; and `Profile` stores the agent's role description and capability profile, injected as context into Planning and Reasoning at every cycle.

module.

Planning Module. In each cognitive cycle, the planning module produces one atomic sub-task. The LLM gets as input the world state, action history, previous plans, and dialogue history from memory, as well as role and capabilities from the profile.

Reasoning Module. Given the atomic plan from the planning module, the reasoning module translates it into a single low-level action executable in the environment. This separation of concerns allows the planning module to reason abstractly, while the reasoning module handles the grounding of intent into concrete environment actions (e.g., move, search room, pick up victim, send message). The actions are represented by tool calls, and the LLM needs to select the appropriate tool and fill in the required parameters. See Appendix A.3 for the list of low-level actions implemented.

Memory Module. Each agent maintains its own decentralized memory, which consists of two components. The first is a `base memory`, which records world beliefs (known victim locations, obstacle status, and explored rooms), plan updates, and received messages. The second component is a structured `episode memory`, which is the primary source of cognitive context for the planning module. Each decision cycle opens a record that captures the sub-task generated by Planning, the motivation for that sub-task, the action selected by Reasoning, and any validation failure.

Profile Module. It contains the agent's role description and capabilities (vision, medical expertise, and strength). This information is retrieved by the planning and reasoning modules to ensure that the plans are feasible to be completed by the agent and that they are in line with the agent's role.

3.4. Agent Infrastructure

The agent infrastructure sits between the environment and the agent, with two responsibilities: translating between their respective representations and ensuring that the simulation does not freeze during LLM inference.

Translation layer. The infrastructure exposes three modules that together decouple the LLM agent from the environment. They can be visualized in Figure 3.3. On the input side, the `perception` module takes

the full raw world state and filters and structures it to the partial, local text-based *observation* the agent can see. The `Communication` module plays a similar role for the message channel: it processes the raw MATRX message queue into structured data to send to the LLM prompt. On the output side, the `execution` module maps the agent's chosen *tool call* to a concrete, low-level MATRX action. Moreover, the *validation step* is done inside the execution module. Before the tool call is transformed into a low-level action, it is checked against the world state. A rejected action triggers a new reasoning call directly, with a short feedback message, avoiding a full planning cycle. This check does not replace the one performed by the environment; it is an additional mechanism to help the agent self-correct faster. The list of all validations can be found in Appendix A.3. Together, these three modules mean that the simulation loop requires no knowledge of the agent type.

Navigation and autopilot. Not every tick requires LLM inference. To simplify the action space, once an agent chooses to move to a specific location, the infrastructure handles the navigation, so the agent does not need a separate cognitive cycle for every atomic action required to traverse the grid. This abstraction, querying the language model only at high-level decision points while delegating low-level movement, mirrors the design of environments such as Overcooked [60] and keeps the number of expensive full cognitive cycles proportional to the number of decisions.

3.5. Team Coordination

This section details the team coordination mechanism, which supports joint action execution and two coordination modes.

Joint actions. A single agent cannot complete some tasks due to capability constraints, such as an agent with medium medical expertise being unable to carry a severely injured victim. In such cases, the system supports *joint actions*, which require the simultaneous cooperation of two agents. When an agent determines that assistance is required, it broadcasts a help request specifying the task details and target location to all teammates, then it pauses its own execution until a reply is received or the request times out. Upon receiving the help request via a message, the other agents can accept or decline. To prevent coordination conflicts, such as multiple agents responding to the same task, the requesting agent commits to the first helper that accepts the request, and the system removes the request from the Message Pool. The agent then autonomously navigates to the target destination. Once both agents arrive, the joint action is executed automatically, with the system ensuring synchronized action timing. Following execution, the participants resume their individual tasks.

Coordination mode: centralized vs decentralized. By default, coordination is fully decentralised, and each agent decides its own tasks. As an alternative, the framework provides an optional centralized mode that can be enabled without modifying the agents. The planner has no physical presence in the environment. It observes a joint world state created by merging the beliefs of all agents and assigns high-level tasks to the agents at the start of the episode. This gives a single point of control over the division of work, while each agent still handles its own execution independently.

3.6. Human Participation

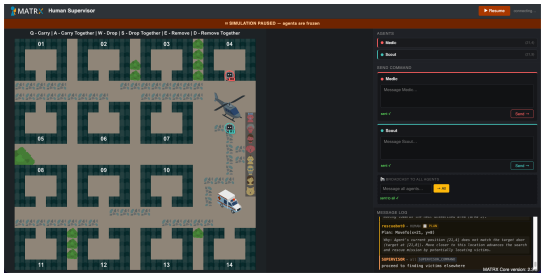
The HuLA-MAS framework integrates humans directly into the existing simulation architecture. By using the same communication and perception modules designed for the autonomous agents, humans can join the simulations in two distinct ways: as an embodied teammate or a supervisor.

3.6.1. Embodied Teammate

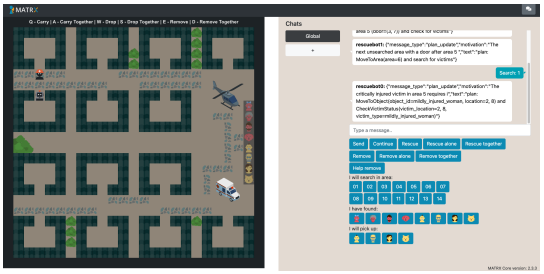
The embodied mode was supported by the original MATRX simulation, which treats the human as another agent whose inputs come from the keyboard. The simulation interface can be seen in Figure 3.4b. The human is subject to the same physical limitations, partial observability, and environmental constraints as the rest of the team. To interact, humans use keyboard inputs, such as using arrow keys to move or dedicated keys for carrying, removing, or dropping objects. Unlike the LLM-based agents, which specify action targets explicitly through tool-call parameters, the system automatically handles the target of the human's carry or remove actions by selecting the closest valid object in range.

3.6.2. Supervisor

The supervisor interface can be seen in Figure 3.4a. In supervisor mode, the human does not control an embodied agent and instead oversees the autonomous team from an external interface. The supervisor can send natural language instructions to the team or to specific agents. However, those instructions, information, or feedback messages do not interrupt the agent's cognitive cycle. Instead, they are saved in the memory with a special messaging flag, and the agent can reason and plan using this extra information. To ensure sufficient observability and transparency to the human, the simulation makes available structured and filtered outputs of the modules, such as current plans and the motivation behind them, as well as the tool calls made.



(a) **Human supervisor view.**



(b) **Human teammate.**

Figure 3.4: **Human support inside the SaR environment.** The human can join as either an embodied teammate or as a supervisor. a) On the left side of the screen, the supervisor sees the joint world state by combining the representations from all LLM-based agents. Humans cannot interact with the environment but can send messages via the chat boxes on the right of the screen. The environment supports messaging individual agents, as well as broadcasting messages to the entire team. Messages sent by the agents are also available to give the supervisor observability on what the agents are doing and why. b) The human can join as a keyboard-controlled embodied teammate. Natural language messaging is supported, as well as some predefined messages for more standardized messaging and ease of use.

3.7. Conclusion

This chapter presented the design and implementation of HuLA-MAS, a modular research framework for studying heterogeneous human–LLM multi-agent systems in a search-and-rescue environment. The framework was developed to follow the desiderata identified in chapter 2, and simultaneously extends the MATRX-based search-and-rescue environment as a new testbed for LLM-MAS research.

The architecture is organized around four components that satisfy the desiderata established in the previous chapter. The MATRX environment provides a configurable 2D grid world that enforces physical capability constraints, which enforces collaboration to complete the task. The agent infrastructure decouples the LLM-based agents from the simulation loop. The LLM-based agent has modules that allow each agent in a team to be configured with a different strategy, thus supporting cognitive heterogeneity. The team coordination layer provides the messaging and coordination infrastructure required for joint actions. Human support is integrated across these components, allowing human participants to join the simulation either as embodied teammates, subject to the same physical constraints as the LLM agents, or as external supervisors who can send natural-language messages to the team.

Together, these design choices allow HuLA-MAS to support both fully autonomous LLM-MAS and mixed human–LLM MAS configurations and establish a framework for the exploratory heterogeneity study in chapter 4.

3.7.1. Limitations and Future Work

While HuLA-MAS implements most of the desiderata and provides a baseline for studying heterogeneity, several design decisions were deliberately deprioritized. These limitations are discussed below, together with concrete directions for future work.

Environment and Simulation Dynamics. First, the environment is currently static, changing only in response to direct agent interaction. Future iterations should introduce dynamic elements such as spreading fires, obstacles that spawn during the simulation, and victim severity that increases over time. These additions would make the environment non-deterministic as well as introduce a sense of urgency. Second, the MATRX simulation enforces strict constraints for **joint actions**, requiring both partners to share the exact same cell at the exact same time. This coordination challenge risks inflating the difficulty of cooperation. Agents with the correct intention to collaborate may fail due to timing mismatches, leading to wasted time and effort.

Agent Perception and Navigation. The environment currently lacks multimodal support. Agents perceive the world strictly through structured text representations. While this item is part of the desiderata, the current version of HuLA-MAS deprioritized it. This was due to practical reasons as well as to focus on the set of features that would enable the study of heterogeneity. Future work should explore integrating visual inputs to enhance spatial understanding. Furthermore, low-level movement is currently delegated to an A^* **autopilot navigation** system. While this mirrors abstractions used in environments like Overcooked [60] and ensures the language model is only queried at high-level decision points, it might reduce embodiment realism.

Human-Agent Interaction. Human roles are statically assigned per run, and there is currently no mechanism for a human participant to switch between acting as an external supervisor and an embodied teammate mode during an active simulation. Future work can consider allowing humans to switch between roles, similarly to a real-world setting.

Despite these limitations, HuLA-MAS serves as an extensible testbed for studying collaborative human-LLM MAS in a search-and-rescue environment. The publicly available codebase is designed for easy expansion, allowing new agent roles, capabilities, module implementations, and map layouts to be introduced.

4

An Exploratory Study of Heterogeneity in LLM-Based Multiagent Systems

Research Question 3

How does heterogeneity in physical and cognitive capabilities affect task and process performance in a collaborative task?

Large language models can serve as cognitive engines for agents that can reason and act. When the task becomes too difficult to be solved by a single agent, LLM-based MAS are deployed to coordinate and solve a shared problem [2, 4]. However, the agents do not have to be alike, and they can differ in their abilities and information. Agent heterogeneity is a design dimension in multi-agent systems research that shapes the team performance and coordination [61, 62]. The concept of heterogeneity is also central to multi-robot systems, which use robots with different, complementary capabilities to improve the performance of the system [63, 64].

Heterogeneity is a well-established and researched topic in classical multi-agent systems and multi-robot systems. However, its understanding as part of LLM-based MAS is limited. Existing work mostly introduces heterogeneity between agents by using different underlying models [20, 65] or distinct roles [10]. Another common method is assigning different personas to the agents [22, 66, 67]. However, studies showed that prompt-based persona heterogeneity generates random or task-dependent performance improvements [68, 69].

This limited scope leaves two gaps. First, it is unclear how varying other components of the LLM-based agents, such as the cognitive modules, influences the performance of the system. Second, the effect of physical capability heterogeneity, which naturally arises in embodied systems due to agents having different hardware, remains underexplored. Both dimensions are important for understanding LLM-based team behavior in an embodied setting.

In many recent works on LLM-based MAS, the default is often a homogeneous team of identical agents. As research progresses toward more adaptable and robust LLM-based agents, the field focuses on *"self-evolving AI agents"* that optimize their internal components based on data and environmental feedback [5, 70, 71]. This means that as the team collaborates and agents get more individual experience, the agents might not resemble identical copies. Thus, heterogeneity becomes an inevitable factor in these multi-agent systems, which requires a better understanding of how it affects the teams.

Answering this question provides empirical insights for future LLM-MAS development. As LLM-based agents are being deployed in complex, simulated, and eventually real-world collaborative settings alongside humans, understanding how team heterogeneity influences both task success and coordination is important. Knowing how to build and balance heterogeneity in these agents can inform the design of more resilient, adaptable, safe, and capable systems.

Using the HuLA-MAS framework and the simulated search-and-rescue environment introduced in chapter 3, this chapter evaluates and compares the performance of homogeneous and heterogeneous LLM-MAS. By

varying the agents' physical capabilities (e.g., vision range, strength) and the planning and reasoning module strategies, the experiments isolate the effects of each form of heterogeneity.

4.1. Background

4.1.1. Modular LLM-Based Agent

LLM-based agents are built around the LLM, which serves as the central reasoning engine. These agents also include optional modules that enhance the LLM's ability to perform complex tasks, though specific implementations vary across different frameworks and architectural designs. The literature identifies several core components, such as perception, planning, memory, and tool use, as the foundational modules for single agents [5]. Multi-agent frameworks extend this baseline by adding a communication module to enable inter-agent coordination [50], while certain architectures [3, 72] further incorporate a profile module encoding background, demographics, and role specifications. Lastly, the execution module translates the agent's decisions into concrete outputs.

Modularized LLM Agent Search [59] aims to automatically optimize LLM agent designs by leveraging the knowledge gained from previously published or evaluated modules. The authors implement module evolution, recombination, and a performance predictor to create and improve the agent design. However, they only tested this approach in the single-agent case.

4.1.2. Heterogeneity in Multiagent Systems

Heterogeneity is not a new concept in the MAS literature. Across this research, heterogeneity was included in both the physical capabilities and the behavior of the agents. Findings on the impact of diversity on human teams have inspired research in multi-agent systems, where agent heterogeneity has been explored [23, 73].

While recent advancements in LLM-based MAS have primarily focused on homogeneous agents or simple role-playing within a single model, physical robotics research has long demonstrated the necessity of heterogeneous teams in search-and-rescue (SaR) operations [31]. Other work looked into the idea of how a team reaches consensus, promotes heterogeneity, and pairs it with prompt-based persona diversity [67]. In real-world disaster scenarios, SAR deployments are inherently heterogeneous, frequently combining ground, aerial, and marine platforms with varying operational, sensory, and computational capabilities. Furthermore, post-disaster environments require flexible, ad-hoc deployments where teams must dynamically adapt to the varying capabilities of the available units. Despite the proven benefits of combining diverse computational and physical capabilities in physical MAS, cognitive and functional heterogeneity remains largely underexplored in LLM-based ad-hoc teamwork.

Past work on RL-based agents also raises the question of *how to define heterogeneity in MAS* [74]. This survey identifies several components: the physical structure and functionality of agents, their behavior or policy, and their interactions with the environment. It defines heterogeneity across observations, responses, effects, objectives, and policies and introduces it by changing the corresponding elements of the partially observable Markov game between the agents.

4.2. Methodology

This section details the approach used to configure and evaluate heterogeneous agent teams within the HuLA-MAS framework. Rather than treating agents as identical replicas, HuLA-MAS supports forming teams of agents with heterogeneous **physical capabilities** and **cognitive module strategies**. The modular architecture introduced in chapter 3 allows each type of heterogeneity from this study to be manipulated independently. By varying a single component while holding the remaining modules, the architecture, and the environment constant, heterogeneity becomes a controlled variable rather than a side effect.

4.2.1. Heterogeneity in Physical Capabilities

Physical heterogeneity is introduced by assigning distinct capability profiles to the agents across three dimensions: `vision`, `medical expertise`, and `strength`. Each dimension features three competence levels (Table 4.1).

The three dimensions differ in the type of interdependencies they create within the team. Vision adds soft interdependencies, since it only sets the agent's observation radius and therefore affects only how efficiently the agent perceives and explores the world. `Medical expertise` and `strength` lead to hard interdependencies that affect task feasibility. For example, `medical expertise` determines which victim injury levels an agent

can rescue alone, while `strength` limits the types of obstacles that can be removed by the agent. Attempting an action that exceeds an agent's capability results in a failure. This creates explicit interdependencies between the agents, forcing agents to communicate and share information to overcome individual limitations and successfully execute joint actions.

Physical heterogeneity is introduced through a set of fixed capability presets, each combining the three physical dimensions. The `generalist` preset is used in the homogeneous baseline and gives the agents medium capability across all three dimensions. Next, specialist presets have been created that give the agent high capability in one of the dimensions but low in another one. The three specialist presets are `medic` (high medical expertise capability), `scout` (high vision capability), and `heavy lifter` (high strength capability). All presets share an identical total budget, ensuring that any observed difference between homogeneous and heterogeneous teams cannot be attributed to one team having more resources than another. Full preset specifications are provided in Figure A.4.

Table 4.1: **Physical capability dimensions.** Each dimension is scored on a discrete three-point scale. The *soft* capability (Vision) affects only efficiency, not task feasibility; the *hard* capabilities (Medical Expertise, Strength) determine whether an agent can complete an action alone (**Solo**) or must cooperate (**Joint**). These constraints are strictly enforced at the simulation level.

Capability	**Low**	**Medium**	**High**
Vision (soft): observable radius			
Radius	1 block	2 blocks	3 blocks
Medical expertise: victim injury levels			
 Critical	**Joint**	**Joint**	**Solo**
 Mild	**Joint**	**Solo**	**Solo**
 Healthy	**Solo**	**Solo**	**Solo**
Strength: obstacle types cleared			
 Tree	**Solo**	**Solo**	**Solo**
 Stone	**Joint**	**Solo**	**Solo**
 Rock	**Joint**	**Joint**	**Solo**

4.2.2. Heterogeneity in Cognitive Modules

Beyond physical differences, agents within the team are equipped with multiple strategies of the `planning` and `reasoning` modules, representing different problem-solving approaches. All implementations are adapted from the single-agent framework **AgentSquare** [59], which extracts the modules from existing LLM-based agents from the literature into modules with interchangeable prompting strategies. Figure 4.1 outlines the set of different prompting strategies used for each module. Because these strategies vary in computational overhead, each is assigned a complexity cost. This is used to define the per-agent cost used to enforce a *fixed team budget* across experimental conditions. Finally, as the simulation progresses, each agent extends their own `memory` with individual experiences and beliefs. Thus, some level of heterogeneity is also added through this module.

While in this thesis, each strategy changes only the module's prompt, the architecture supports the usage of modules that have completely different, specialized implementations. More details on the implementation and prompts used can be found in section A.3.

4.3. Experiments

This section presents the experimental setup of the **exploratory study**. All experiments are conducted in the simulated search-and-rescue environment described in Chapter 3. Each agent is fully decentralized and plans and reasons over its own partial belief state and coordinates through messaging without any central orchestrator.

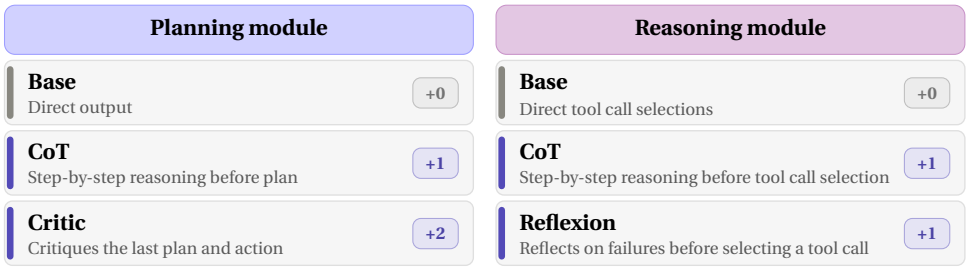


Figure 4.1: **Cognitive module implementations used in the experiments.** Complexity badges (+0 to +3) highlight the relative computational overhead of each strategy: Base (+0) is the baseline; CoT and Reflexion (+1) add minimal overhead; Critic (+2) lengthens the prompt and requires extra output structure.

4.3.1. Experiment 1: Effect of Heterogeneity

This experiment focuses on whether and to what degree introducing physical and cognitive heterogeneity affects the task performance and process.

1. **Homogeneous Baseline:** All agents use the generalist preset and identical cognitive modules (Chain-of-Thought strategy for both planning and reasoning modules).

2. **Physical Heterogeneity Only:** Agents are assigned distinct physical capabilities using the specialist presets (e.g., medic, scout, heavy-lifter) but have identical cognitive modules.

3. **Cognitive Heterogeneity Only:** Agents share the generalist preset but are assigned different cognitive modules.

4. **Full Heterogeneity:** Agents differ in both physical capabilities and cognitive modules.

To ensure a fair comparison, a fixed *team cognitive budget* of 4 points for teams of 2 agents and 6 points for teams of 3 was enforced. Further details on the exact implementation of each agent (what module strategies and physical capabilities were given to each agent) from each condition can be found in Figure A.1.

4.3.2. Experiment 2: Team Size

This experimental setting examines how scaling the team from $N = 2$ to $N = 3$ agents influences the dynamics of heterogeneity across all conditions. The focus is on whether the addition of a third agent amplifies the benefits of heterogeneity or not.

4.3.3. Experiment 3: Environmental Interdependency

This experiment explores whether the role of heterogeneity shifts depending on the structural demands of the environment. The same four conditions are evaluated in both the Easy and Difficult environments, with the focus on how soft and hard interdependencies influence the importance and the effect of heterogeneity.

Two distinct grid configurations have been created: **easy** and **difficult**. Both worlds have 14 areas (houses) surrounded by walls, accessible only through specific entrances, as well as 14 victims in total. However, they differ in their required coordination levels:

- **Easy (Soft Interdependencies):** This map contains 14 uniformly distributed mild victims. The only obstacles found in this setting are the trees, which require low-strength capability to be successfully removed. For this reason, the environment does not strictly enforce collaboration. While agents can communicate and divide spatial exploration to improve efficiency, as well as act jointly, any individual agent has sufficient physical capability to complete the victim rescues and object removals alone.

- **Difficult (Hard Interdependencies):** This map introduces critical victims and heavy rock and stone obstacles blocking entrances. Here, collaboration is enforced by the hard interdependencies. Because critical victims and heavy obstacles exceed the baseline physical capabilities, agents are forced to coordinate by either executing joint actions or explicitly requesting help from teammates possessing the necessary specialized physical capabilities.

4.3.4. Experimental Setup

To isolate the effect of physical and cognitive heterogeneity from that of model heterogeneity, all agents are powered by the same language model. Two model sizes from the Qwen family [75] are evaluated: **Qwen3-8B** and **Qwen3-32B**. For both models, all hyperparameters are kept constant across all agents and conditions. Each condition is executed for *6 independent runs*, and all metrics are reported as mean ± standard deviations across these runs. Each run continues until all victims have been rescued or a fixed time budget of 7 hours is reached.

All experiments were run on the *DelftBlue supercomputer* [76], on the A100 GPU partition with a single A100 GPU, 8 CPU cores, and 4 GB of memory per core. The asynchronous LLM polling of HuLA-MAS routes all agent calls to the same shared model instance instead of requiring one device per agent.

4.3.5. Evaluation Metrics

Given the exploratory nature of this study, metrics were selected to capture not only whether the team succeeded but also how well it coordinated. This provides a more complete set of metrics to interpret the patterns observed across experiments. Metrics are grouped into three categories.

Task performance metrics. These capture the degree to which the team accomplished its objective to search the map and rescue the victims found.

- **Success Rate (SR, ↑):** The percentage of total victims rescued on the grid.
- **Spatial Coverage (SC, ↑):** The percentage of navigable cells collectively explored by the team.
- **Number of Actions (# Actions):** The number of physical actions performed by the team.

Process and coordination metrics.

- **Spatial Coverage Overlap (SC-O, ↓):** Measures how effectively the team divided the search space and avoided redundant exploration. Lower values indicate better spatial division of work and less overlap in the searched areas.
- **Communication Volume (CV):** The fraction of total actions spent on inter-agent messaging rather than environment-directed actions. Since agents operate in a partially observable, decentralized setting, elevated communication can indicate active coordination intent. It is calculated as the ratio of messaging actions to total actions.
- **Help Requests (HR):** The total number of explicit help requests sent by the team.

Computational cost metrics.

- **Validated Actions:** Total actions accepted by the action validator (independent of environment acceptance). Provides a measure of how many cognitive cycles the agents had to make to achieve observed outcomes.
- **LLM Calls:** Total number of model inference requests made during the simulation. Reflects the computational cost of achieving observed outcomes.
- **Average Latency:** Mean wall-clock time per LLM call, in seconds. Complements LLM calls to characterize the computational requirements of each condition.

4.4. Results

This section summarizes the main results of this chapter. Given that this is an exploratory study, no statistical testing was performed. All comparisons reported below are descriptive, and they characterize trends observed across the runs rather than statistically significant effects. The differences between conditions should be confirmed in a larger study. Table 4.2 reports the *Qwen3-8B* results. The *Qwen3-32B* results and system-level LLM metrics are provided in Table A.6.

The results are organized around the 3 experiments described in the previous section. The first two use an `easy` environment, where only soft interdependencies exist and every agent can complete all actions alone.

The third experiment uses the `difficult` environment, which introduces hard interdependencies and where joint actions are required to finish the task. In each experiment, all 4 conditions (i.e., homogeneous team (baseline), physical heterogeneity only, cognitive heterogeneity only, and full heterogeneity) were tested and compared.

Table 4.2: **Task-performance and process metrics (*Qwen3-8B*)**. Values are mean ± standard deviation over six runs. **SR**: success rate (%). **SC**: % of navigable cells explored. **SC-O**: Spatial Coverage Overlap (lower = better). **CV**: Communication volume (%). **HR**: Help requests (total).

Agents	Heterog.	SR (%) ↑	SC (%) ↑	#Actions ↓	SC-O ↓	CV (%) ↑	HR ↑
Soft Interdependence							
2 (Exp 1)	None	33.3 ± 26.2	60.7 ± 26.2	**761 ± 573**	0.66 ± 0.26	**29.56 ± 45.46**	2.8 ± 3.8
	Physical	41.7 ± 22.3	72.7 ± 24.5	872 ± 506	0.62 ± 0.19	14.73 ± 22.66	2.2 ± 3.5
	Cognitive	36.9 ± 21.9	57.3 ± 21.5	805 ± 535	**0.56 ± 0.21**	21.91 ± 28.56	**16.2 ± 22.1**
	Full	**51.2 ± 19.9**	**81.9 ± 4.3**	1085 ± 127	0.59 ± 0.18	0.87 ± 1.05	1.7 ± 3.1
3 (Exp 2)	None	42.9 ± 15.7	70.4 ± 15.3	390 ± 199	0.64 ± 0.15	11.61 ± 11.46	9.5 ± 14.8
	Physical	59.5 ± 29.2	82.6 ± 8.2	406 ± 203	0.63 ± 0.16	18.73 ± 22.73	15.0 ± 11.6
	Cognitive	47.6 ± 17.9	66.4 ± 16.4	**372 ± 156**	**0.57 ± 0.13**	17.84 ± 17.72	12.0 ± 22.4
	Full	**66.7 ± 23.3**	**85.6 ± 6.5**	391 ± 164	0.63 ± 0.13	**19.29 ± 18.45**	**18.7 ± 17.2**
Hard Interdependence							
2 (Exp 3)	None	0.0 ± 0.0	7.7 ± 8.3	**5 ± 8**	0.82 ± 0.25	56.94 ± 17.01	2.8 ± 4.5
	Physical	**1.2 ± 2.9**	20.8 ± 12.1	124 ± 79	0.42 ± 0.40	69.52 ± 14.70	60.2 ± 57.2
	Cognitive	0.0 ± 0.0	23.5 ± 7.5	45 ± 39	**0.15 ± 0.26**	**71.90 ± 12.64**	12.7 ± 9.9
	Full	**1.2 ± 2.9**	**28.2 ± 6.7**	118 ± 67	0.28 ± 0.32	67.10 ± 16.92	**88.8 ± 57.5**
3 (Exp 3)	None	1.2 ± 2.9	23.2 ± 10.3	**37 ± 25**	**0.21 ± 0.12**	**59.75 ± 26.80**	22.8 ± 5.5
	Physical	2.4 ± 3.7	41.6 ± 18.2	132 ± 42	0.40 ± 0.16	40.14 ± 18.33	29.8 ± 17.4
	Cognitive	**7.1 ± 6.4**	38.2 ± 8.7	60 ± 32	0.23 ± 0.15	38.80 ± 12.19	17.0 ± 4.2
	Full	6.0 ± 7.0	**44.9 ± 17.1**	132 ± 59	0.32 ± 0.24	48.06 ± 21.13	**56.8 ± 52.7**

Experiment 1: Effect of Heterogeneity in a Two-Agent Team (Soft Interdependence)

Heterogeneity improved both task-performance metrics. The fully heterogeneous team rescued the most victims (51.2% SR) and explored the most of the map (81.9% SC). Moreover, it outperformed both conditions where heterogeneity was isolated (physical or cognitive). The homogeneous baseline rescued the fewest victims. This matches the intuition that heterogeneity helps a team.

The number of actions the team needed to reach these results suggests that the improvement was not due to better exploration or division of work. Rescuing one victim takes about six high-level actions: moving to the area entrance, optionally removing an obstacle, searching the area, moving to the victim, picking it up, and dropping it at the safe zone. Clearing all 14 areas should take on approximately 85 actions if the optimal next action is always picked (this does not take into account the traveling between the areas). However, the teams spent more, and most tool calls were made for movement actions. The points to inefficient navigation: agents revisit areas, retrace paths, or attempt to enter blocked areas. The high scores were reached despite, not because of, efficient coordination.

The process metrics also show that heterogeneity lowers the spatial overlap. At the same time, heterogeneity may have enforced a clearer division of work based on each agent's specific capabilities, which could instead increase overlap: a scout agent, for example, only needs to find victims and then inform the others, and this can lead to some duplicated exploration. Surprisingly, the fully heterogeneous team reached its results with very few help requests, which may indicate that complementary capabilities, rather than active coordination, accounted for its performance in this setting.

Experiment 2: Scaling to a Three-Agent Team (Soft Interdependence)

Adding a third agent raised success rates in every condition and lowered the action count, which would suggest a more efficient distribution of work. As in the two-agent setting, full heterogeneity gave the best task performance. However, the efficiency is confounded by inference latency. All agents in a run share a single

model instance, and the mean latency per LLM call rises with team size (Table A.6) because that instance must serve more concurrent requests. Since each run has a 7-hour limit, a higher per-call latency reduces the number of cognitive cycles that a team can complete within the budget. The lower action count may thus reflect the time limit rather than better coordination.

The communication pattern seen in Experiment 1 did not persist. With three agents, full heterogeneity was associated with the highest communication volume and the most help requests, rather than the no-communication behavior observed in the smaller team. However, since the standard deviations are so large, it means that the runs either have 0 messages sent or spikes in communication. This suggests that heterogeneity may not influence how much the agents communicate. The result is reasonable in a soft-interdependence environment, where additional communication would mostly translate into wasted actions. A possible explanation is that the additional agent changes the benefit of communicating. In a larger team, the risk of redundant exploration is higher, so the effort spent on a help request is more likely to benefit the team.

The results for the larger *Qwen3-32B* model, reported in Table A.6, show that the results do not generalize. The high success rate of full heterogeneity seen with the *Qwen3-8B* model does not generalize. For both team sizes, success rates are comparable between conditions, and the homogeneous baseline no longer performs the poorest. Surprisingly, the teams perform poorer overall.

Experiment 3: Hard Interdependence

Under hard interdependence, task performance collapsed across every condition and team size relative to the soft-interdependence environment. The number of actions, as well as the communication volume and number of help requests, provides an explanation on why this might have happened. Compared to the soft interdependence environment, all heterogeneity configurations and team sizes had an increased number of help requests and spent more actions on communication than on physical ones. This effort went into setting up joint actions, but the agents failed to complete them, getting stuck trying to synchronize the shared action.

However, this is less due to heterogeneity than to a limitation of the joint actions themselves. While these results are in line with the intuition that an environment where joint actions are needed is more challenging and would require more communication, the joint actions might be too complex for the agents to successfully perform and coordinate. This suggests the need to find another way to handle these by the game engine.

4.5. Conclusion

This chapter presented an empirical evaluation of the HuLA-MAS framework, focusing on how physical and cognitive diversity influences collaboration and performance of multi-agent systems. Taking advantage of the modular architecture, heterogeneity was introduced by assigning distinct physical capability presets to the agents and building agents with different module strategies. By comparing these heterogeneous configurations against a homogeneous baseline, while keeping the same cognitive and physical team budget, the experiments assess both the task outcome and the process-oriented metrics. An important factor to keep in mind when interpreting these results is that the study is exploratory, based on six runs per condition with large run-to-run variance, and was not subjected to statistical testing.

In the soft-interdependence settings (Experiments 1 and 2), heterogeneity seemed to improve task performance, both in the number of victims rescued and in spatial coverage. These gains, however, did not clearly stem from more efficient exploration or a better division of work: the high action counts in the two-agent team indicate that the coverage came at a cost in efficiency. Moreover, the agents brute force the environment rather than making optimal tool calls. Increasing the team size from two to three agents raised success rates across all conditions and reduced the number of actions, which may partly reflect a more efficient distribution of work, although this drop is also confounded by the higher per-call latency of larger teams under a fixed time budget. The communication behavior was less stable across team sizes, suggesting that heterogeneity alone does not determine how much the agents communicate.

In the hard-interdependence setting (Experiment 3), task performance collapsed for every condition and team size: the agents spent most of their effort on communication and help requests in an attempt to complete the joint actions but largely failed to coordinate and synchronize them. The fact that the decrease in performance is uniform across all conditions could suggest that the bottleneck is the joint-action mechanism rather than team composition. Moreover, all these trends appeared only with 8B and did not replicate with a larger model (32B). Because the study is exploratory and based on six runs per condition with large run-to-run variance, these findings should be read as trends rather than established effects.

Limitations and Future Work

While the current set of capabilities and cognitive modules is sufficient for establishing a baseline for the heterogeneity study, it is not exhaustive. Many existing published agents feature specialized module implementations that could be extracted and adapted. Future work can extend the framework along these dimensions by adding new capabilities and modules without requiring architectural changes.

The results on the environment with hard interdependencies highlight some limitations of the joint actions in HuLA-MAS. The idea of joint actions, where two LLM-based agents must synchronize and perform together an action that affects the same object, coordinating both their locations and the timing of triggering the action, is, to the best of our knowledge, novel in LLM-MAS environments. The collapse in performance under hard interdependence suggests that this mechanism, in its current form, is too demanding for the agents to handle reliably and that future work should explore alternative ways for the game engine to mediate or simplify these joint actions.

Another limitation is the set of metrics used. Although these metrics were informed by prior work, additional process and coordination metrics could be added. For example, collaboration fluency could be measured using a survey such as [77], covering perceived communication effectiveness, helpfulness, and trust on a 7-point Likert scale.

A further limitation concerns the scope of the evaluation itself. The reported results are based on a small number of runs per condition and were not subjected to statistical testing, and the large variance observed across runs means that the differences between conditions cannot be translated to concrete claims about the impact of heterogeneity. A further limitation concerns the statistical power of the evaluation. Each condition was run only six times, and the run-to-run variability is large. With six runs, the study is therefore underpowered to distinguish the conditions reliably, and for this reason no formal significance testing was performed. The reported differences should accordingly be read as qualitative trends rather than measured effects Future work could validate these ideas across additional model families and sizes, together with a larger number of runs.

5
Conclusion

This thesis aimed to understand how agent heterogeneity affects task performance and the collaboration process of LLM-based MAS. This research was motivated by the observation that heterogeneity is an important characteristic of traditional MAS and multi-robot systems but received less attention in the context of LLM-based agents. Moreover, as self-improving AI agents are increasingly used within MAS, heterogeneity becomes an unavoidable property of the teams, which should be understood before these systems are deployed in real-world tasks.

Chapter 2 aimed to answer the first *RQ*, regarding the desirable characteristics of environments to study heterogeneity in LLM-MAS. A survey of existing simulated embodied environments found that no environments simultaneously support multi-human, multi-agent teams, configurable collaboration, multimodal interaction; LLM-compatible dynamics, and heterogeneity. These findings were combined into desiderata for a new embodied environment designed to guide the development of new testbeds.

Next, **chapter 3** describes the development of HuLA-MAS, a novel framework to study heterogeneity in LLM-MAS in a search-and-rescue environment, based on the findings from the first chapter. It was built to answer the second *RQ*, which asked how the desirable characteristics identified by answering the previous research question inform the development of a new embodied environment to facilitate the study of heterogeneity in LLM-MAS. Human support was also added, allowing them to join as either embodied teammates or supervisors, aligning the framework with the desiderata.

Finally, an exploratory study was conducted using this framework to understand how physical and cognitive heterogeneity affect an LLM-MAS team in the SaR setting (**chapter 4**). Under soft interdependence, where every agent can complete the task alone, heterogeneity appeared to improve task performance. The fully heterogeneous team rescued the most victims and achieved the highest spatial coverage, while the homogeneous baseline performed the poorest. However, these improvements did not clearly result from more efficient exploration or a better division of work, as the high action counts suggest much of the effort was spent on movement rather than rescue actions. When joint actions were required (hard interdependence), performance broke down for all conditions and team sizes. The agents used most of their actions to communicate but mostly failed to synchronize them. Overall, the results offer preliminary support for the intuition that heterogeneity can improve LLM-MAS performance. These findings are exploratory and not definitive claims about the influence of heterogeneity.

The remainder of this chapter discusses the limitations and future work. Then, it connects this research to the broader computer science field and reflects on the societal implications, risks, and stakeholders' perspectives.

5.1. Limitations and Future Work

This section describes some directions that future research could take, using the HuLA-MAS framework and the SaR environment.

1. **Human evaluation in HuLA-MAS**: Future work could take advantage of the human support offered by HuLA-MAS, for example, through a user study examining how team heterogeneity influences the experience of a human teammate. Another direction would be to investigate how a human's perceived

collaboration fluency and trust are influenced by knowing that the agents have different physical and cognitive capabilities.

2. **Team formation optimization**: The conditions tested in this thesis were crafted manually, meaning that the module strategies and capabilities assigned to each agent were chosen by hand. While this was sufficient for exploring the effect of heterogeneity, future research could investigate optimizing the team formation itself. Similar to AgentSquare's approach to agent optimization, which recombined and selected the best modules for a given task and context, such a method could optimize module selection across the entire team.

3. **Quantifying heterogeneity**: In this study, heterogeneity is treated as a binary variable, either a team is homogeneous or it is not, similarly to other LLM-MAS research [67]. However, this offers a limited view on how diverse a heterogeneous team actually is and makes it impossible to compare the degree of heterogeneity across teams. Future work could treat heterogeneity as a quantitative measure, in line with broader MAS research, where such measures were considered [78]. This way, future studies would be able to understand how the level of heterogeneity affects the performance.

4. **Cognitive heterogeneity**: This work implemented cognitive heterogeneity in a way similar to the AgentSquare [59] module optimization approach, changing only the prompts used by the modules to introduce heterogeneity. This may have introduced too little variation. Future work could extend the framework by incorporating other module implementations from previously published agents.

5. **Stronger empirical testing**: The current thesis investigates only a limited number of scenarios. Future work could carry out more runs with larger teams, additional model families and sizes, and a wider range of module implementations. While this work was intended as an exploratory study using the HuLA-MAS framework and the SaR environment, more runs are needed to determine whether the observations are significant.

6. **HuLA-MAS improvements**: chapter 3 discusses some of the limitations of the framework. Future work could extend it by adding multimodal support and richer environment dynamics.

Potential Risks

The potential risks associated with this study are not related to the code artifacts, but rather to how the findings can be used or misread. First, the results in chapter 4 rely on limited runs with high variance and lack significance testing. This introduces the risk of these findings being incorrectly used to justify design decisions in deployed systems. This is mitigated by highlighting the exploratory nature of the study.

Second, search and rescue is a high-stakes domain where even small errors can compromise safety. Because natural-language communication can make agents appear more competent and reliable, it can encourage misplaced trust from the users of the system. Therefore, while the HuLA-MAS simulation framework enables rapid research iterations, it must not replace physical validation with real-world robots. Finally, deploying LLM-MAS requires substantial GPU time and energy, making scalability an issue. LLM usage should be weighed against simpler alternatives or restricted to specific modules.

5.2. Reflection on implications and applicability

The **search and rescue** scenario was chosen for HuLA-MAS because it represents a critical domain where mixed-agent teams are already being deployed in the real world. Recently, autonomous robots have played a significant role in Urban Search and Rescue (USAR) to assist responders in reducing the response time and to conduct search operations to locate victims and map areas that are too dangerous to cover by humans [79]. However, research on multi-robot systems highlights the technical limitations that still limit their usability and reliability [31]. For example, these systems struggle with efficient task allocation, especially in heterogeneous teams. Next, control complexity and the operator's cognitive load are influenced by the number of deployed robots that they need to supervise and control.

Another implication in the field of computer science is the **new framework and environment for LLM-MAS research**. Currently, only a limited number of embodied environments exist, and they are often at the two extremes. Overcooked-style benchmarks are light and easy to work with, but they are very simplified and have little transfer to real-life tasks. On the other hand, Minecraft-based environments can be too complex and expensive to run, especially in the first stages of research. The development of HuLA-MAS was meant to be an

environment realistic enough to study heterogeneity and human inclusion but still light enough to allow fast iteration.

This thesis also highlights the important **role humans play in MAS**, which is reflected in HuLA-MAS. Because disaster response is a task that cannot afford mistakes, first responders need systems that are intuitive, predictable, and easy to use. Hallucinations in LLM-MAS can cascade and make a robotic team act on false information and follow a dangerous strategy. Agents might fail, so the team must be reliable enough to adapt the plan. Thus, responsible deployment requires that humans can still question, override, and correct the system and not only approve its decisions. Autonomous agents should not be deployed without human supervision and intervention, and understanding how mixed teams work together is the first step toward responsible AI use.

Ultimately, the development of the HuLA-MAS environment and the exploratory findings presented in this thesis are intended to motivate further research into heterogeneity within multi-agent systems. As disaster response is a critical and high-stakes domain, the development of these systems impacts several key groups. Researchers require simulation tools, like HuLA-MAS, to safely study and validate new ideas before deployment. First responders need systems that are highly predictable and reliable, as well as easy to interact with, so they will benefit from more research on this field. By providing an easily extendable testing ground, this work aims to encourage continued exploration of the practical applicability of LLMs in collaborative, mixed-agent teams in the search-and-rescue domain.

A

Appendix

A.1. Use of AI

This section outlines the use of AI tools during this thesis.

AI tools, specifically Claude and Gemini, were used during the writing of the final report. These tools were used *only* to reformulate text created by me to improve the clarity, correct grammatical errors, and help with table formatting. Additionally, AI was used to generate baseline LaTeX code using the `tikzpicture` library. Existing designs of the figures created by me in draw.io, together with detailed textual descriptions, were provided to generate the LaTeX code. This helped with the complex syntax of `tikzpicture`. All figures were then modified and finalized by me to reflect the intended designs accurately.

AI-generated images were used for the report's cover page. Specifically, Gemini was used to generate the human and robot avatars. These images were then added manually using *Canva* to the map image.

During the coding phase, Claude was used as a supportive tool for debugging, code fixing, and syntax formatting for the prompts used by the modules. AI was not used to design the software architecture or to inform any high-level technical or design choices.

All text, data, figures, and code generated or assisted by AI tools were reviewed, verified, and approved by me. AI tools were at no point used to generate research ideas, formulate research questions, generate claims or results/data.

A.2. Reproducibility

All code and the folders with the results are publicly available at: https://github.com/alemarcu23/HuLA-MAS. All results can be found in the following repository. It contains the metrics for each individual run, as well as the raw files with all metrics.

Sampling hyperparameters

Table A.1 lists all hyperparameters. No parameters were tuned during evaluation.

Table A.1: Hyperparameters

Caller	Parameter	Value	API key	Notes
Rescue agents	Temperature	0.6	`temperature`	Qwen3 thinking-mode default
	Top-p	0.95	`top_p`	
	Top-k	20	`top_k`	
	Min-p	0.0	`min_p`	disabled
	Max new tokens	32 768	`num_predict` / `max_new_tokens`	

Each experiment run logs its full configuration in `simulation_metrics.json`, including `agent_model`, `reasoning_strategies`, `planning_strategies`, `comm_strategies`, `capability_knowledge`, and `agent_roles`.

Appendix: Responsible NLP Research Checklist

- **A1. Did you discuss the limitations of your work?**
  Yes. Can be found in section 5.1. Specific limitations for each study can be found at the end of each chapter.

- **A2. Did you discuss any potential risks of your work?**
  Yes. Section 5.

B. Did you use or create scientific artifacts?

Yes.

- **B1. Did you cite the creators of artifacts you used?**
  Yes. In Section 3.2, the exact version is v2.3.3 and can be found at MATRX.

- **B2. Did you discuss the license or terms for use and/or distribution of any artifacts?**
  No. The repository we used is public on GitHub. No mention of the license.

- **B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified?**
  N/A.

- **B4. Did you discuss the steps taken to check whether the data that was collected/used contains any information that names or uniquely identifies individual people or offensive content and the steps taken to protect/anonymize it?**
  N/A

- **B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?**
  N/A

- **B6. Did you report relevant statistics like the number of examples, details of train/test/dev splits, etc. for the data that you used/created?**
  N/A

C. Did you run computational experiments?

Yes.

- **C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?**
  Yes. 4.3 Experiments.

- **C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?**
  Yes. Experimental setup is discussed in Section 4.3.

- **C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?**
  Yes. 4.4 Results.

- **C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?**
  N/A.

D. Did you use human annotators (e.g., crowdworkers) or research with human subjects?

No.

E. Did you use AI assistants (e.g., ChatGPT, Copilot) in your research, coding, or writing?

Yes. See appendix A.1.

A.3. LLM-Agent Design Details

Low-level action set

The list of tool calls the reasoning module may output can be found in Table A.2. Navigation and coordination for joint actions are handled automatically once a partner indicates intent using the correct tool call.

Table A.2: The low-level action set exposed to the reasoning module as tool calls. Apart from Drop and Send Message actions, the rest require multi-tick execution. Single agent and joint actions are visualized separately.

Tool call	Effect
`MoveTo(x, y)`	Navigate to grid cell (x, y) (A^* pathfinding).
`MoveToArea(area)`	Navigate to the given area (A^* pathfinding).
`NavigateToDropZone()`	Navigate to the drop zone (A^* pathfinding).
`SearchArea(area)`	Search the given area for victims and obstacles.
`CarryObject(object_id)`	Pick up and carry a victim alone.
`Drop()`	Drop the currently carried victim.
`RemoveObject(object_id)`	Remove an obstacle alone.
`SendMessage(message, send_to, message_type)`	Send a message to one teammate or broadcast to all.
`CarryObjectTogether(object_id, partner_id)`	Carry a victim jointly with a teammate.
`RemoveObjectTogether(object_id, partner_id)`	Remove an obstacle jointly with a teammate.

Action Validation

Table A.3 lists all checks performed, grouped by action. A rejected action is never silently dropped. The feedback string is stored as `last validation error` and surfaced in both the next Planning prompt (so the planner can revise the sub-task) and the next Reasoning prompt (so the reasoner can immediately try a corrected tool call). After three consecutive validation failures on the same action, the pipeline resets to the Planning stage so a fundamentally different approach can be chosen.

Agent Modules Implementations

This appendix explains the implementation and prompts used by the main cognitive modules: Reasoning and Planning. Moreover, it also explains how the Profile module configures the capabilities of the agents and how it limits the action space.

The strategies used in the `Planning` and `Reasoning` modules of HuLA-MAS are drawn from prompting techniques from the LLM or LLM-based agents literature. The prompt structures are adapted from AgentSquare [59] by replacing the original task context with the SaR environment and action vocabulary used in HuLA-MAS. Each strategy adds extra instructions to the base system prompts.

Table A.3: Action validation checks and feedback messages returned to the agent.

Action	Condition checked	Feedback message
`MoveNorth/South/East/West`	Agent is at the corresponding grid edge	*"Cannot move [dir]—already at the [edge] edge of the grid."*
`MoveTo`	x or y argument missing	*"MoveTo requires both x and y coordinates."*
	Coordinates are not integers	*"MoveTo coordinates must be integers, got x=..., y=..."*
	Target is outside grid bounds	*"MoveTo target (x, y) is outside the grid (valid range: ...)."*
	Target equals current position	*"MoveTo (x, y) is your current positi—thisis is a no-op. Choose a different destination to make progress."*
`MoveToArea`	Area argument missing	*"Area number is required."*
	Area does not exist	*"Area N does not exist. Valid areas: [list]."*
`SearchArea`	Area does not exist (inherited)	(same as MoveToArea)
	Obstacle is blocking the area door	*"Cannot search area N: obstacle 'id' is blocking the door at [x,y]. Use RemoveObject(object_id='id') to clear it first."*
	Agent is not adjacent to the area door	*"You must navigate to the door of area N at [x,y] before calling SearchArea. Use MoveTo(x=..., y=...) first."*
`CarryObject`	Target not adjacent or not a victim	*"object_id 'id' is not adjacent or not a valid victim."*
	Agent already carrying an object	*"You are already carrying [id]."*
	Medical capability too low for critically injured victim (*informed* mode only)	*"You cannot carry critically injured victims alone — your medical skill is too low."*
`Drop`	Agent not carrying anything	*"You are not carrying anything. There is nothing to drop."*
	Carrying a critically injured victim (*informed* mode only)	
`RemoveObject`	Target not adjacent or not an obstacle	*"object_id 'id' is not adjacent or not a valid obstacle."*
	Strength too low to remove a stone or rock solo (*informed* mode only)	*"Your strength is too low to remove this obstacle solo. Ask a teammate to help with RemoveObjectTogether."*
	Strength is medium and target is a rock (*informed* mode only)	*"Rocks are too heavy for you to remove alone. Use RemoveObjectTogether with a partner."*
`SendMessage`	Message is empty or whitespace-only	*"SendMessage requires a non-empty message."*
	Recipient is not `'all'` and not a known teammate	*"Unknown recipient 'id'. Use 'all' for broadcast or one of: [list]."*
`RemoveObjectTogether`	Target not adjacent or not a rock/stone	*"object_id 'id' is not adjacent or not a valid obstacle."*
	Target is a tree (trees cannot be removed cooperatively)	*"Trees cannot be removed cooperatively. Use RemoveObject to remove 'id' solo."*
	`partner_id` is not a known teammate	*"RemoveObjectTogether failed: partner_id 'id' is not a known teammate. Valid teammates: [list]."*
	Partner is not adjacent to the obstacle	*"RemoveObjectTogether failed: Partner 'id' is not adjacent to obstacle 'id'. Ask a teammate for help using SendMessage(message_type='ask_help'), then wait for them to arrive before retrying."*
`CarryObjectTogether`	Target not adjacent or not a victim	*"object_id 'id' is not adjacent or not a valid victim."*
	Agent already carrying an object	*"You are already carrying [id]. Drop it first before picking up another object."*
	`partner_id` is not a known teammate	*"CarryObjectTogether failed: partner_id 'id' is not a known teammate. Valid teammates: [list]."*
	Partner is not adjacent to the victim	*"CarryObjectTogether failed: Partner 'id' is not adjacent to victim 'id'. Ask a teammate for help using SendMessage(message_type='ask_help'), then wait for them to arrive before retrying."*

Planning Module

The planning module is responsible for producing one atomic sub-task every decision cycle. The prompts used by the planning module can be found in this section.

Base Prompt. This is the baseline strategy, and no additional instructions are added to the prompt.

Output Schema Base Planning Prompt

```
You are the planner for a Search and Rescue agent. You must emit ONE atomic plan
for the agent to execute. The plan should be in natural language but contain
explicit details (e.g. victim IDs, coordinates). Return a single JSON object.

Your `high_level_task` is your role-based directive for the entire run. Every plan
you emit MUST be a concrete step toward that high-level goal. Before choosing a
plan, ask: "Does this plan advance my high-level task?" If the answer is no, pick a
different plan. The only exceptions are URGENT help requests and genuine blockers
that must be cleared first (e.g. an obstacle preventing you from reaching your
target). PREFER planning something at your current location instead of moving.

You should expect to find victims inside the areas that have different severity
levels. Prioritize rescuing victims with higher severity levels first. You should
expect obstacles to block entrances to areas and paths to victims. Removing these
obstacles is necessary to access the victims and to search the areas.

Use all available information before planning: OBSERVATION, WORLD STATE BELIEF,
MEMORY, other TEAMMATES' plans, and MESSAGES. Keep in mind your HISTORY of actions
and plans, and your CAPABILITIES. The AREA COVERAGE block shows the % of each area
searched; an area is searched only at 100%. DO NOT output the same plan every time -
use the MEMORY. If something did not work, try again later or delegate the task.

Depending on your capabilities, some tasks will be impossible for you to perform
alone (e.g. carrying a critical victim or removing a big rock). These require a
JOINT action with a teammate (CarryObjectTogether / RemoveObjectTogether). To set
one up, plan to send a message with intent = "ask_help" naming the target and its
[x,y]; once a teammate accepts, both of you perform the joint action at the target.
If you see a teammate's "ask_help" you can fulfill, proactively plan to reply "yes".

## COORDINATING WITH TEAMMATES
- If you receive an `ask_help` message, you MUST plan to reply "yes" or "no". If
  "yes", you are COMMITTED to going to the target and completing the joint action -
  prioritize it above all else until finished, and do NOT emit any other plan.
- Send messages to teammates to share information, delegate a task you cannot
  perform, or coordinate. This is the only way to influence teammates' plans.
```

The output schema for this prompt is:

Output Schema Base Planning Prompt

```
{
  "next_plan": "one atomic sub-task with explicit IDs and coordinates",
  "motivation": "<=50 words explaining how this plan advances the high-level task."
}
```

Chain-of-Thought (CoT). The CoT strategy instructs the LLM to reason step by step before committing to a plan [80]. The following extra instructions are prepended to the base prompt. The output schema is the same as in the base prompt.

CoT: Extra Instructions

```
[ChainOfThought] Let's think step by step. Before choosing a plan, reason privately about
 the current world state, the memory, the current needs, and what each teammate is doing.
```

Critic. This strategy extends the prompt with a self-evaluation step. Before generating the next plan, the LLM is asked to judge whether the agent's most recent action completed the previous plan. This produces an explicit success flag and, in the case of failure, a single piece of actionable feedback (the *critique*). The next plan will be generated based on this self-evaluation. This strategy is inspired by the Critic module in **MindForge** [81].

Critic: Extra Instructions

```
Judge if `last_action` completed `last_plan`, using all available information.
1. If `last_action.name` is "SendMessage", ALWAYS emit: success = true, critique = ""
2. If `last_action` is empty/null (first cycle), emit: success = true, critique = ""
3. A MoveTo whose target [x, y] does not match the agent's current position:
    The critique must suggest checking for obstacles blocking the path.
4. Otherwise, judge from OBSERVATION, WORLD STATE BELIEF, and the YOU block:
   - Carry plan    => success if `carrying` now matches the targeted victim.
   - Move-to-[x,y] => success if position now equals [x, y].
   - Remove-obstacle plan => success if the obstacle is GONE from belief.
   - Drop plan     => success if `carrying` is empty.
The critique (if failure) MUST be ONE piece of actionable feedback.
```

Output Schema Critic Planning Prompt

```
{
  "critic": {
    "success": true | false,
    "critique": "actionable feedback, or empty string if success"
  },
  "next_plan": "one atomic sub-task with explicit IDs and coordinates",
  "motivation": "<=50 words explaining how this plan advances the high-level task."
}
```

Reasoning Module.
The reasoning prompt takes the plan generated by Planning Modules and outputs a single tool call. The strategy prompt is added to the base prompt.

Base Reasoning System Prompt

```
You are a Search and Rescue agent. Your job each cycle is to execute the
`current_plan` by emitting ONE tool call.
The plan given to you will include all necessary details (can include:
victim_id, location [x, y], obstacle_id, partner_id). You do NOT re-plan.
If the plan looks unreachable, choose the tool call that makes the most
direct progress toward it.
The tool calls that you make require precise argument names and types -
follow the schema exactly. If you are unsure about the correct arguments,
pick the best guess but be prepared for the validator to reject it and
force you to try again with a different action.
Your belief of the world and memory should help you choose the best action
to advance the current plan and the parameters.
ALWAYS PREFER TO DO AN ACTION, and not a MoveTo.
HUMAN SUPERVISOR COMMANDS:
- If the user-message contains a "== SUPERVISOR COMMAND ==" block, follow
  it as soon as possible, since it comes from a human supervisor. Ignore
  other plans, but use your information since the feedback might be vague.
CAPABILITY LIMITATIONS & JOINT ACTIONS:
```

```
- Some actions exceed your capability alone (e.g. carrying a critical
  victim or removing a big rock). These are done as JOINT actions with a
  teammate: CarryObjectTogether or RemoveObjectTogether.
- To start a joint action, first send an "ask_help" naming the target id
  and its [x,y]. Once a teammate accepts, call the joint action with their
  id as partner_id - navigation and rendezvous to the target are automatic.
- If a teammate sends an "ask_help" you can fulfill, reply "yes" and head
  to the target; the joint action fires once you are both adjacent.
MESSAGING:
- SendMessage to teammates to share information or delegate tasks. Use it
  often to share discoveries, ask a teammate to take over a task, or offer
  your own capabilities to assist others.
```

Chain-of-Thought (CoT). The CoT strategy asks the LLM to reason step by step before selecting a tool call.

CoT: Extra Instructions

```
[ChainOfThought] Solve the task step by step. Before choosing an output, reason privately
about the current world state, the active plan, and the likely consequences of each
candidate action.
```

Reflexion. The Reflexion strategy instructs the LLM to reflect explicitly on past failures before selecting an action. This is why the validation step, part of the Execution module, is important. This is adapted from Reflexion [82].

Reflexion: Extra Instructions

```
[Reflexion] Before acting, reflect on what you have done and what failed. If a previous
action failed (see critic_feedback.critique), you MUST try a completely different approach
 this turn - do not repeat the same action.
```

Self-Refine. The Self-Refine strategy uses two LLM calls per reasoning step. In the first pass, the model selects a tool call as it would under the base prompt. In the second pass, a separate verification prompt asks the model to check the tool call for correctness against the tool schema. If the verifier identifies an error, it outputs a revised tool call; otherwise, it confirms the original. This is adapted from Self-Refine [83]

Self-Refine: First pass

```
[Self-Refine] Choose the best tool call for the active plan. A second verification pass
will check your call for correctness against the tool schema, so be precise with argument
names and types.
```

Self-Refine: Second pass

```
You are verifying a proposed tool call for a Search and Rescue agent.
Proposed call: {tool_name}({args_json})
Active plan: {subtask}
Agent position: {position}
Available tool names: {tool_names}
Check (a) the tool name is in the available list, (b) required arguments are present and of
 the correct type (coordinates are ints, object_ids are strings), (c) the call advances or
 completes the plan, (d) it is NOT a no-op (e.g. MoveTo to the agent's own position).
Respond with exactly one line in one of these two formats:
 correct
 error, revised: {"name": "<tool>", "args": {...}}
```

A.4. Experimental Conditions

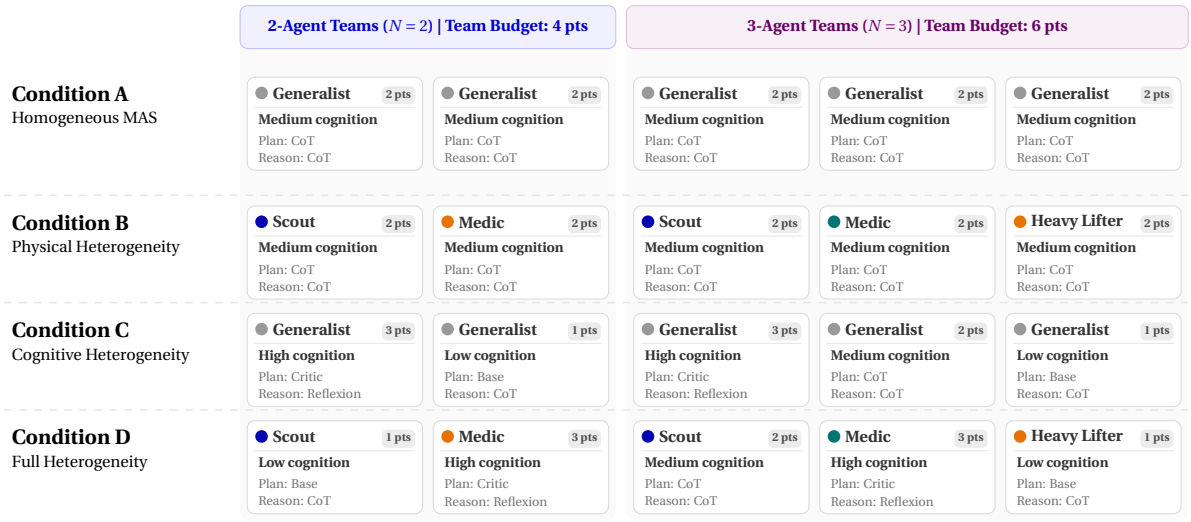


Figure A.1: **Experimental conditions and strict resource distribution.** The experimental matrix isolates the effects of physical and cognitive heterogeneity. To ensure a fair baseline comparison, a strict Team Cognitive Budget is enforced (4 points for $N = 2$ configurations; 6 points for $N = 3$ configurations). Heterogeneous configurations do not receive more computational resources; instead, they redistribute the identical budget asymmetrically across high, medium, and low cognition roles.

Physical Heterogeneity: Agent Presets

Table A.4: **Agent presets** used in the heterogeneous experimental conditions.

Attribute	Scout	Medic	Heavy Lifter	Generalist
Vision	High	Low	Medium	Medium
Medical	Medium	High	Low	Medium
Strength	Low	Medium	High	Medium

A.5. Extra results

All results can be found in the following repository. It contains the metrics for each individual run, as well as the raw files with all metrics.

Results: Qwen3-32B

Table A.5: **Task-performance and process metrics (Qwen3-32B)**. Values are mean ± standard deviation over six runs. **SR**: success rate (%). **SC**: % of navigable cells explored. **SC-O**: Spatial Coverage Overlap (lower = better). **CV**: Communication volume (%). **HR**: Help requests (total). This table details the performance of the larger 32B parameter model under soft interdependences (Easy environment).

Agents	Heterog.	SR (%) ↑	#Actions ↓	SC (%) ↑	SC-O ↓	CV (%) ↑	HR ↑
Easy Environment (Soft Interdependence)							
2 (Exp 1)	None	28.6 ± 12.0	303 ± 125	61.9 ± 6.2	0.73 ± 0.21	20.33 ± 16.51	12.5 ± 10.5
	Physical	23.8 ± 17.9	**234 ± 144**	**79.8 ± 7.7**	0.50 ± 0.08	**28.72 ± 19.25**	16.8 ± 10.7
	Cognitive	**39.3 ± 9.9**	292 ± 124	68.3 ± 6.7	0.71 ± 0.15	16.57 ± 22.10	14.8 ± 19.5
	Full	26.2 ± 11.7	287 ± 138	74.4 ± 22.4	**0.48 ± 0.12**	25.36 ± 23.91	**25.3 ± 30.7**
3 (Exp 2)	None	**39.3 ± 9.9**	214 ± 42	64.1 ± 3.7	0.69 ± 0.13	10.83 ± 9.83	5.0 ± 4.3
	Physical	23.8 ± 13.3	253 ± 132	**81.4 ± 3.8**	**0.44 ± 0.11**	**13.69 ± 10.25**	**17.2 ± 10.2**
	Cognitive	33.3 ± 5.8	**192 ± 8**	63.8 ± 3.7	0.52 ± 0.16	11.67 ± 7.49	7.8 ± 6.2
	Full	**39.3 ± 21.6**	225 ± 126	79.0 ± 3.9	0.47 ± 0.11	13.13 ± 9.48	11.0 ± 8.5

LLM-related metrics

Table A.6: **LLM-related metrics across all experiments**. Values are mean ± standard deviation over the six runs per condition. **Lat.**: mean latency per call (s), averaged over agents. **#Failures**: actions rejected by the validator (team total). The best value is shown in **bold**.

Setting	Model	Heterogeneity	Calls ↓	Lat (s) ↓	#Failures ↓
2 agents, Easy	Qwen3-8B	None	2399 ± 574	21.0 ± 5.7	304 ± 310
		Physical only	2615 ± 758	19.7 ± 7.2	438 ± 276
		Cognitive only	**2259 ± 839**	**16.0 ± 1.7**	**277 ± 203**
		Full Heterogeneity	2713 ± 318	17.6 ± 2.3	498 ± 146
	Qwen3-32B	None	931 ± 219	50.7 ± 12.8	**40 ± 42**
		Physical only	930 ± 120	49.7 ± 7.0	71 ± 50
		Cognitive only	**810 ± 285**	48.8 ± 16.6	109 ± 128
		Full	1017 ± 175	**46.9 ± 9.1**	125 ± 77
3 agents, Easy	Qwen3-8B	None (baseline)	1182 ± 203	**53.4 ± 13.8**	146 ± 105
		Physical only	1195 ± 172	55.8 ± 11.8	**110 ± 61**
		Cognitive only	**1146 ± 144**	62.4 ± 7.0	127 ± 78
		Full	1217 ± 165	57.1 ± 9.6	130 ± 57
	Qwen3-32B	None (baseline)	560 ± 128	127.3 ± 36.7	32 ± 17
		Physical only	468 ± 91	**109.5 ± 37.5**	**21 ± 21**
		Cognitive only	542 ± 56	127.1 ± 18.3	53 ± 38
		Full	**410 ± 87**	137.4 ± 51.1	26 ± 18
2 agents, Difficult	Qwen3-8B	None (baseline)	**330 ± 774**	**9.1 ± 7.7**	**0.5 ± 0.5**
		Physical only	1977 ± 119	22.9 ± 1.7	21 ± 19
		Cognitive only	984 ± 713	16.8 ± 11.4	4 ± 7
		Full	1504 ± 129	30.5 ± 2.9	36 ± 35
3 agents, Difficult	Qwen3-8B	None (baseline)	432 ± 338	30.3 ± 20.6	**4 ± 5**
		Physical only	820 ± 125	45.5 ± 15.5	35 ± 28
		Cognitive only	**401 ± 351**	**21.1 ± 13.1**	12 ± 12
		Full	941 ± 336	42.2 ± 21.3	46 ± 33

Bibliography

[1] Sarfraz Brohi, Qurat-ul-ain Mastoi, NZ Jhanjhi, and Thulasyammal Ramiah Pillai. "A research landscape of agentic ai and large language models: Applications, challenges and future directions". In: *Algorithms* 18.8 (2025), p. 499.

[2] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. "Multi-agent collaboration mechanisms: A survey of llms". In: *arXiv preprint arXiv:2501.06322* (2025).

[3] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. "A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges". In: *Vicinagearth* 1.1 (2024), p. 9.

[4] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. "Large language model based multi-agents: A survey of progress and challenges". In: *arXiv preprint arXiv:2402.01680* (2024).

[5] Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, et al. "A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems". In: *arXiv preprint arXiv:2508.07407* (2025).

[6] Andrea Matarazzo and Riccardo Torlone. "A survey on large language models with some insights on their capabilities and limitations". In: *arXiv preprint arXiv:2501.04040* (2025).

[7] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Zican Dong, Yupeng Hou, Beichen Zhang, Yingqian Min, Junjie Zhang, Peiyu Liu, et al. "A survey of large language models". In: *Frontiers of Computer Science* 20.12 (2026), p. 2012627.

[8] Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, et al. "Large language model agent: A survey on methodology, applications and challenges". In: *arXiv preprint arXiv:2503.21460* (2025).

[9] Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. "A survey on large language models for code generation". In: *ACM Transactions on Software Engineering and Methodology* 35.2 (2026), pp. 1–72.

[10] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Steven Yau, Zijuan Lin, Liyang Zhou, et al. "MetaGPT: Meta programming for a multi-agent collaborative framework". In: *International Conference on Learning Representations*. Vol. 2024. 2024, pp. 23247–23275.

[11] Josip Vrdoljak, Zvonimir Boban, Marino Vilović, Marko Kumrić, and Joško Božić. "A review of large language models in medical education, clinical decision support, and healthcare administration". In: *Healthcare*. Vol. 13. 6. MDPI. 2025, p. 603.

[12] Samuel Schmidgall, Rojin Ziaei, Carl Harris, Eduardo Reis, Jeffrey Jopling, and Michael Moor. *AgentClinic: a multimodal agent benchmark to evaluate AI in simulated clinical environments*. 2024. arXiv: 2405.07960 [cs.HC].

[13] Tianshi Zheng, Zheye Deng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Zihao Wang, and Yangqiu Song. "From automation to autonomy: A survey on large language models in scientific discovery". In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025, pp. 17744–17761.

[14] Shen Wang, Tianlong Xu, Hang Li, Chaoli Zhang, Joleen Liang, Jiliang Tang, Philip S Yu, and Qingsong Wen. "Large language models for education: A survey and outlook". In: *IEEE Signal Processing Magazine* 42.6 (2026), pp. 51–63.

[15] Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, et al. "Where llm agents fail and how they can learn from failures". In: *arXiv preprint arXiv:2509.25370* (2025).

[16] Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei A Zaharia, Joseph Gonzalez, and Ion Stoica. "Why Do Multi-Agent LLM Systems Fail?" In: *Advances in Neural Information Processing Systems*. Ed. by D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen. Vol. 38. Curran Associates, Inc., 2025. URL: https://proceedings.neurips.cc/paper_files/paper/2025/file/b1041e52d3be19f0a9bc491657488e4a-Paper-Datasets_and_Benchmarks_Track.pdf.

[17] Luis Emilio Gomez and Patrick Bernet. "Diversity improves performance and outcomes". In: *Journal of the National Medical Association* 111.4 (2019), pp. 383–392.

[18] Marie-Élène Roberge and Rolf Van Dick. "Recognizing the benefits of diversity: When and how does diversity increase group performance?" In: *Human Resource management review* 20.4 (2010), pp. 295–308.

[19] Daan Van Knippenberg, Wendy P Van Ginkel, and Astrid C Homan. "Diversity mindsets and the performance of diverse teams". In: *Organizational Behavior and Human Decision Processes* 121.2 (2013), pp. 183–193.

[20] Rui Ye, Xiangrui Liu, Qimin Wu, Xianghe Pang, Zhenfei Yin, Lei Bai, and Siheng Chen. "X-MAS: Towards Building Multi-Agent Systems with Heterogeneous LLMs". In: *arXiv preprint arXiv:2505.16997* (2025).

[21] Yingxuan Yang, Chengrui Qu, Muning Wen, Laixi Shi, Ying Wen, Weinan Zhang, Adam Wierman, and Shangding Gu. "Understanding Agent Scaling in LLM-Based Multi-Agent Systems via Diversity". In: *arXiv preprint arXiv:2602.03794* (2026).

[22] Vinay Samuel, Henry Peng Zou, Yue Zhou, Shreyas Chaudhari, Ashwin Kalyan, Tanmay Rajpurohit, Ameet Deshpande, Karthik Narasimhan, and Vishvak Murahari. "Personagym: Evaluating persona agents and llms". In: *arXiv preprint arXiv:2407.18416* 8.9 (2024).

[23] Rasika Muralidharan, Haewoon Kwak, and Jisun An. *Can Lessons From Human Teams Be Applied to Multi-Agent Systems? The Role of Structure, Diversity, and Interaction Dynamics*. 2025. arXiv: 2510.07488 [cs.CL]. [Link].

[24] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[25] Margaret Mitchell, Avijit Ghosh, Alexandra Sasha Luccioni, and Giada Pistilli. "Fully autonomous ai agents should not be developed". In: *arXiv preprint arXiv:2502.02649* (2025).

[26] Henry Peng Zou, Wei-Chieh Huang, Yaozu Wu, Chunyu Miao, Dongyuan Li, Aiwei Liu, Yue Zhou, Yankai Chen, Weizhi Zhang, Yangning Li, et al. "A Call for Collaborative Intelligence: Why Human-Agent Systems Should Precede AI Autonomy". In: *arXiv preprint arXiv:2506.09420* (2025).

[27] Miles Q Li and Benjamin CM Fung. "Security concerns for large language models: A survey". In: *Journal of Information Security and Applications* 95 (2025), p. 104284.

[28] Kevin J Feng, David W McDonald, and Amy X Zhang. "Levels of autonomy for ai agents". In: *arXiv preprint arXiv:2506.12469* (2025).

[29] James J Odell, H Van Dyke Parunak, Mitch Fleischer, and Sven Brueckner. "Modeling agents and their environment". In: *International Workshop on Agent-Oriented Software Engineering*. Springer. 2002, pp. 16–31.

[30] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision". In: *Ieee Access* 8 (2020), pp. 191617–191643.

[31] Daniel S Drew. "Multi-agent systems for search and rescue applications". In: *Current Robotics Reports* 2.2 (2021), pp. 189–200.

[32] Aleksandar Krnjaic, Raul D Steleac, Jonathan D Thomas, Georgios Papoudakis, Lukas Schäfer, Andrew Wing Keung To, Kuan-Ho Lao, Murat Cubuktepe, Matthew Haley, Peter Börsting, et al. "Scalable multi-agent reinforcement learning for warehouse logistics with robotic and human co-workers". In: *2024 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2024, pp. 677–684.

[33] Matthew Gombolay, Xi Jessie Yang, Bradley Hayes, Nicole Seo, Zixi Liu, Samir Wadhwania, Tania Yu, Neel Shah, Toni Golen, and Julie Shah. "Robotic assistance in the coordination of patient care". In: *The International Journal of Robotics Research* 37.10 (2018), pp. 1300–1316.

[34] Loo G Pee, Shan L Pan, and Lili Cui. “Artificial intelligence in healthcare robots: A social informatics study of knowledge embodiment”. In: *Journal of the Association for Information Science and Technology* 70.4 (2019), pp. 351–369.

[35] Henry Peng Zou, Wei-Chieh Huang, Yaozu Wu, Yankai Chen, Chunyu Miao, Hoang Nguyen, Yue Zhou, Weizhi Zhang, Liancheng Fang, Langzhou He, Yangning Li, Dongyuan Li, Renhe Jiang, Xue Liu, and Philip S. Yu. *LLM-Based Human-Agent Collaboration and Interaction Systems: A Survey*. 2025. arXiv: 2505.00753 [cs.CL]. [Link].

[36] Emanuele La Malfa, Gabriele La Malfa, Samuele Marro, Jie Zhang, Elizabeth Black, Michael Luck, Philip Torr, and Michael Wooldridge. “Large language models miss the multi-agent mark”. In: *Advances in Neural Information Processing Systems* 38 (2026).

[37] Matthew Johnson, Jeffrey M Bradshaw, Paul J Feltovich, Catholijn M Jonker, M Birna Van Riemsdijk, and Maarten Sierhuis. “Coactive design: Designing support for interdependence in joint activity”. In: *Journal of Human-Robot Interaction* 3.1 (2014), pp. 43–69.

[38] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009, p. 1152. ISBN: 978-0-13-604259-4.

[39] Anand S Rao, Michael P Georgeff, and Elizabeth A Sonenberg. “Social plans: A preliminary report”. In: *ACM SIGOIS Bulletin* 13.3 (1992), p. 10.

[40] Jacques Ferber and Gerhard Weiss. *Multi-agent systems: an introduction to distributed artificial intelligence*. Vol. 1. Addison-wesley Reading, 1999.

[41] Danny Weyns, Andrea Omicini, and James Odell. “Environment as a first class abstraction in multiagent systems”. In: *Autonomous agents and multi-agent systems* 14.1 (2007), pp. 5–30.

[42] Wei Wang, Dan Zhang, Tao Feng, Boyan Wang, and Jie Tang. “Battleagentbench: A benchmark for evaluating cooperation and competition capabilities of language models in multi-agent systems”. In: *arXiv preprint arXiv:2408.15971* (2024).

[43] Weiyu Ma, Qirui Mi, Yongcheng Zeng, Xue Yan, Yuqiao Wu, Runji Lin, Haifeng Zhang, and Jun Wang. “Large language models play starcraft ii: Benchmarks and a chain of summarization approach”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 133386–133442.

[44] Olivier Schipper, Yudi Zhang, Yali Du, Mykola Pechenizkiy, and Meng Fang. “Pillagerbench: Benchmarking llm-based agents in competitive minecraft team environments”. In: *2025 IEEE Conference on Games (CoG)*. IEEE. 2025, pp. 1–15.

[45] Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, et al. “Partnr: A benchmark for planning and reasoning in embodied multi-agent tasks”. In: *arXiv preprint arXiv:2411.00081* (2024).

[46] Yubo Dong, Xukun Zhu, Zhengzhe Pan, Linchao Zhu, and Yi Yang. “Villageragent: A graph-based multi-agent framework for coordinating complex task dependencies in minecraft”. In: *Findings of the Association for Computational Linguistics: ACL 2024*. 2024, pp. 16290–16314.

[47] Qian Long, Zhi Li, Ran Gong, Ying Nian Wu, Demetri Terzopoulos, and Xiaofeng Gao. “Teamcraft: A benchmark for multi-modal multi-agent systems in minecraft”. In: *arXiv preprint arXiv:2412.05255* (2024).

[48] Ran Gong, Qiuyuan Huang, Xiaojian Ma, Yusuke Noda, Zane Durante, Zilong Zheng, Demetri Terzopoulos, Li Fei-Fei, Jianfeng Gao, and Hoi Vo. “Mindagent: Emergent gaming interaction”. In: *Findings of the Association for Computational Linguistics: NAACL 2024*. 2024, pp. 3154–3183.

[49] Haochen Sun, Shuwen Zhang, Lujie Niu, Lei Ren, Hao Xu, Hao Fu, Fangkun Zhao, Caixia Yuan, and Xiaojie Wang. “Collab-overcooked: Benchmarking and evaluating large language models as collaborative agents”. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025, pp. 4922–4951.

[50] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. “Building cooperative embodied agents modularly with large language models”. In: *arXiv preprint arXiv:2307.02485* (2023).

[51] Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natalia Vélez, Qingyun Wu, Huazheng Wang, Thomas L Griffiths, and Mengdi Wang. "Embodied llm agents learn to cooperate in organized teams". In: *IEEE Transactions on Computational Social Systems* (2026).

[52] Zhao Mandi, Shreeya Jain, and Shuran Song. "Roco: Dialectic multi-robot collaboration with large language models". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 286–299.

[53] Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, et al. "Interactive grounded language understanding in a collaborative environment: Iglu 2021". In: *NeurIPS 2021 Competitions and Demonstrations Track*. PMLR. 2022, pp. 146–161.

[54] Shao Zhang, Xihuai Wang, Wenhao Zhang, Yongshan Chen, Landi Gao, Dakuo Wang, Weinan Zhang, Xinbing Wang, and Ying Wen. "Mutual theory of mind in human-ai collaboration: An empirical study with llm-driven ai agents in a real-time shared workspace task". In: *arXiv preprint arXiv:2409.08811* (2024).

[55] Jonathan Hyun, Nicholas R Waytowich, and Boyuan Chen. "CREW-Wildfire: Benchmarking agentic multi-agent collaborations at scale". In: *arXiv preprint arXiv:2507.05178* (2025).

[56] Isadora White, Kolby Nottingham, Ayush Maniar, Max Robinson, Hansen Lillemark, Mehul Maheshwari, Lianhui Qin, and Prithviraj Ammanabrolu. "Collaborating action by action: A multi-agent llm framework for embodied reasoning". In: *arXiv preprint arXiv:2504.17950* (2025).

[57] Gonzalo Gonzalez-Pumariega, Leong Yean, Neha Sunkara, and Sanjiban Choudhury. "Robotouille: An asynchronous planning benchmark for LLM agents". In: *International Conference on Learning Representations*. Vol. 2025. 2025, pp. 83757–83792.

[58] Jasper van der Waa and Tjalling Haije. *MATRX: Human Agent Teaming Rapid Experimentation software*. Version 2.3.4. [Link]. Feb. 2024. DOI: 10.5281/zenodo.8154911.

[59] Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. "Agentsquare: Automatic llm agent search in modular design space". In: *arXiv preprint arXiv:2410.06153* (2024).

[60] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. "On the utility of learning about humans for human-ai coordination". In: *Advances in neural information processing systems* 32 (2019).

[61] Peter Stone and Manuela Veloso. "Multiagent systems: A survey from a machine learning perspective". In: *Autonomous Robots* 8.3 (2000), pp. 345–383.

[62] HVD Parunak. "Artificial intelligence in industry". In: *Foundations of distributed artificial intelligence* (1996), pp. 139–164.

[63] David Jung and Alexander Zelinsky. "Grounded symbolic communication between heterogeneous cooperating robots". In: *Autonomous Robots* 8.3 (2000), pp. 269–292.

[64] Yara Rizk, Mariette Awad, and Edward W Tunstel. "Cooperative heterogeneous multi-robot systems: A survey". In: *ACM Computing Surveys (CSUR)* 52.2 (2019), pp. 1–31.

[65] Tingting Yang, Ping Feng, Qixin Guo, Jindi Zhang, Xiufeng Zhang, Jiahong Ning, Xinghan Wang, and Zhongyang Mao. "AutoHMA-LLM: Efficient task coordination and execution in heterogeneous multi-agent systems using hybrid large language models". In: *IEEE Transactions on Cognitive Communications and Networking* 11.2 (2025), pp. 987–998.

[66] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. "Generative agents: Interactive simulacra of human behavior". In: *Proceedings of the 36th annual acm symposium on user interface software and technology*. 2023, pp. 1–22.

[67] Zengqing Wu and Takayuki Ito. "The hidden strength of disagreement: Unraveling the consensus-diversity tradeoff in adaptive multi-agent systems". In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025, pp. 15288–15308.

[68] Tiancheng Hu and Nigel Collier. "Quantifying the persona effect in llm simulations". In: *arXiv preprint arXiv:2402.10811* (2024).

[69] Mingqian Zheng, Jiaxin Pei, Lajanugen Logeswaran, Moontae Lee, and David Jurgens. “When" a helpful assistant" is not really helpful: Personas in system prompts do not improve performances of large language models”. In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. 2024, pp. 15126–15154.

[70] Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, et al. “A survey of self-evolving agents: On path to artificial super intelligence”. In: *arXiv e-prints* (2025), arXiv–2507.

[71] Yue Hu, Yuzhu Cai, Yaxin Du, Xinyu Zhu, Xiangrui Liu, Zijie Yu, Yuchen Hou, Shuo Tang, and Siheng Chen. “Self-evolving multi-agent collaboration networks for software development”. In: *International Conference on Learning Representations*. Vol. 2025. 2025, pp. 23007–23039.

[72] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. “A survey on large language model based autonomous agents”. In: *Frontiers of Computer Science* 18.6 (2024), p. 186345.

[73] Leandro Soriano Marcolino, Albert Xin Jiang, and Milind Tambe. “Multi-agent team formation: Diversity beats strength?” In: *IJCAI*. Vol. 13. 2013.

[74] Tianyi Hu, Zhiqiang Pu, Yuan Wang, Tenghai Qiu, Min Chen, and Xin Yu. “Heterogeneity in Multi-Agent Reinforcement Learning”. In: *arXiv preprint arXiv:2512.22941* (2025).

[75] Qwen Team. *Qwen3 Technical Report*. 2025. arXiv: 2505.09388 [cs.CL]. [Link].

[76] Delft High Performance Computing Centre (DHPC). *DelftBlue Supercomputer (Phase 2)*. [Link]. 2024.

[77] Guy Hoffman. “Evaluating fluency in human–robot collaboration”. In: *IEEE Transactions on Human-Machine Systems* 49.3 (2019), pp. 209–218.

[78] Philip Twu, Yasamin Mostofi, and Magnus Egerstedt. “A measure of heterogeneity in multi-agent systems”. In: *2014 American Control Conference*. IEEE. 2014, pp. 3972–3977.

[79] Kaushik Kannan and Jungyun Bae. “MTU-LLM: LLM-based Multi-Robot Task Allocation and Path Planning for Heterogeneous Robots in Search and Rescue Operations”. In: *AI, Computer Science and Robotics Technology* 4.1 (2025), p. 1.

[80] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. “Chain-of-thought prompting elicits reasoning in large language models”. In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[81] Mircea Lică, Ojas Shirekar, Baptiste Colle, and Chirag Raman. “Mindforge: Empowering embodied agents with theory of mind for lifelong cultural learning”. In: *arXiv preprint arXiv:2411.12977* (2024).

[82] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. “Reflexion: Language agents with verbal reinforcement learning”. In: *Advances in neural information processing systems* 36 (2023), pp. 8634–8652.

[83] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. “Self-refine: Iterative refinement with self-feedback”. In: *Advances in neural information processing systems* 36 (2023), pp. 46534–46594.