Contents

Acronyms

AI Artificial Intelligence. 1

AMM Agent Markov Model. 29

ANOVA Analysis Of the VAriance. 107, 116

DAG Directed Acyclic Graph. 67, 68, 69, 71, 73, 82, 87, 93

DEL Dynamic Epistemic Logic. 48, 53, 58

GUI Graphical User Interface. 126, 127, 130, 132

HAN Human-Aware Navigation. 20, 21

HATP Hierarchical Agent-based Task Planner. 26, 27, 28, 40

HCI Human-Computer Interaction. 8, 9

HHI Human-Human Interaction. 8, 9

HRC Human-Robot Collaboration. 1, 2, 3, 7, 10, 14, 18, 19, 25, 28, 45, 62, 63

HRI Human-Robot Interaction. 7, 8, 10, 14, 25, 26, 125, 126

HTN Hierarchical Task Network. 15, 16, 26, 31, 34, 37, 38, 42, 52, 67, 71

ID Identification. 62, 66, 77, 78

IMHuS Intelligent Multi Human Simuator. 3

InHuS Intelligent Human Simuator. 3, 126

MDP Markov Decision Process. 29, 30

POMDP Partially Observable Markov Decision Process. 29

ROS Robot Operating System. 86, 127, 129, 132, 135, 141, 143, 152

SA Situation Assessment. 52, 58, 94

SD standard deviation. 107, 109, 110, 111, 114, 115, 116

ToM Theory of Mind. 46, 47

Introduction

Contents

Human-Robot Collaboration (HRC) is a growing field in robotics and Artificial Intelligence (AI) research that aims to enable safe and effective teamwork between humans and robots. This field concerns fully autonomous robots. Hence, it excludes exoskeletons or teleoperated robots such as surgery manipulators or remotely operated (aerial) vehicles.

The work presented is assumed to be in the Joint Action context, which is described in [Sebanz 2006b] as "any form of social interaction whereby two or more individuals coordinate their actions in space and time to bring about a change in the environment". Various relationships can exist between humans and robots, e.g., collaborators, companions, guides, tutors, or social interaction partners. In all these cases, we think the robot should help and facilitate the human in terms of physical effort, mental workload, and even emotional state.

Industrial robots are already popular in factories because they are fast, accurate, reliable, and never get tired, which makes them ideal for repetitive factory tasks. However, such robots are usually in dedicated areas where humans cannot enter for safety reasons. Hence, it is still an open challenge to endow robots with enough reliable reasoning capabilities and compliant motion control to allow efficient and trusted direct collaboration between humans and robots.

Moreover, HRC can also occur in various contexts that must be considered, ranging from co-worker robots in factories to householder robots for our everyday lives and including service robots in public places like restaurants and shops.

Focused on the decisional aspect of HRC, this work aims to design autonomous robots able to make explainable, acceptable, and efficient decisions to collaborate with humans.

Human-Robot Collaboration (HRC) is a multidisciplinary field that opens several technical challenges to address. The ideal collaborative robot should be the aggregation of solutions for each of the challenges listed below:

- **Navigation**: The robot should be able to acceptably and efficiently move in a human-populated environment. This implies being mechanically designed for it, anticipating and planning correct trajectories, and being able to adapt and follow these trajectories in real time. The robot should not move threateningly and should account for humans.

- **Manipulation**: The robot should be able to manipulate objects to interact with its environment. Hence, the robot should have an actuator like an arm

and a gripper and should be able to exhibit motions that are efficient and safe to nearby humans.

- **Communication**: To achieve congruent interaction and collaboration, collaborative agents must communicate. This implies that the robot should be able to communicate information to the human and understand the one received from the latter. These communications can be verbal or non-verbal.

- **Perception**: It is mandatory for the robot to have a reliable perception scheme to know the position of near objects, obstacles, and humans. First, relevant sensors must be used and placed on the robot or in the environment itself. After reasoning on the sensory data, relevant facts about the robot's environment, such as objects' positions, spatial relations, reachable objects, human knowledge, intentions, or the state of the current goal, must be extracted.

- **Decision-making**: This aspect is the focus of my work and implies that the robot should be able to make relevant decisions to be collaborative, that is, planning its actions to solve a collaborative task. It also implies being able to supervise the task execution and make online decisions to adapt to uncertainties in real time, which includes human actions and commands.

Contributions and manuscript organization

My work addressed the **decision-making** aspect of HRC and led to three main contributions. I began by participating in the development of a novel human-aware task planner called HATP/EHDA [Buisan 2022]. This approach considers two distinct agent models: an uncontrollable one to estimate the human's behavior and a controllable one to plan the robot's actions accordingly. The agent models include distinct beliefs, agendas, and action models. As a result, the planner can anticipate and even elicit human decisions and actions, but it never compels them. For these reasons, we believe this approach is promising to address HRC, and I built two of my main contributions upon this approach.

My first contribution is to propose some models and algorithms of the Theory of Mind and to integrate them into the deliberation process of HATP/EHDA. Despite considering distinct beliefs, they were only updated according to the description of the action effect provided in the estimated human action model. This means, for instance, that modeling the human observing and learning a new fact by entering a room had to be manually scripted in the 'move' action description. With this contribution, we propose that an agent can learn by observing their surroundings or an action execution. Additionally, we propose a way to detect false human beliefs during the planning process, which may be detrimental to the task. Eventually, these relevant belief divergences are fixed by planning minimal verbal robot communication or delaying robot action that humans will initially not observe.

My second contribution brings planning and execution closer by addressing the turn-taking assumption of HATP/EHDA and exploring parallel executions. We formulated a step-based model of compliant and concurrent joint action. This model describes how the two agents should coordinate as well as four possible online human decisions about the execution: (1) the human decides to be passive and let the robot act alone, (2) the human acts alone and the robot is passive, (3) the human starts acting then the robot adapts and acts in parallel, and finally (4) the human deliberately let the robot decide and start acting before complying with it. This model guides the exploration of our proposed new human-aware task planning approach. After exploring all relevant courses of action, the robot's behavioral policy is extracted using a plan evaluation based on estimations of human preferences. Eventually, by following the produced policy, the robot can comply concurrently with any human online decisions and aims to satisfy human preferences, which can be updated online.

My last contribution is distinguishable from the previous two because it addresses navigation decision-making instead of task planning. Indeed, I designed and implemented two systems simulating interactive social navigating agents endowed with decision-making processes. Additionally, these systems include evaluation processes to record data, compute metrics, and plot navigation performances. These systems generate challenging situations that permit challenging, debugging, and evaluating robot navigation schemes in intricate human-populated scenarios. The first system, Intelligent Human Simuator (InHuS), simulates a single human agent endowed with complex reasoning processes. The second system, Intelligent Multi Human Simuator (IMHuS), simulates several agents with social group behaviors but with less individual reasoning capabilities.

These contributions are detailed in the rest of this manuscript. The latter is structured as follows.

Chapter 1 provides more details about the context of my PhD by discussing related fields and works of HRC. This chapter eventually highlights the gap in the literature that motivates my PhD work.

The rest of the manuscript is divided into two parts. Part I gathers all my work concerning task planning for HRC. It represents a major portion of my PhD work and includes the chapters 2, 3, 4, 5, and 6. Part II concerns decision-making in navigation and how to simulate interactive social navigating agents. This part includes the two last chapters: 7 and 8.

Part I begins with Chapter 2, which presents the HATP/EHDA task planner. This planner has been the keystone of most of my work. Hence, the reader should understand the motivation and methods of this task planner.

Chapter 3 describes my first main contribution, proposing models and algorithms to incorporate Theory of Mind concepts in HRC task-planning. An empirical evaluation is provided and discussed, demonstrating how this contribution solves a broader class of problems than HATP/EHDA without systematic communication.

Chapter 4 presents my second main contribution, proposing a new human-aware task planning approach based on a step-based compliant and concurrent joint ac-

tion model. The approach's description is supported by empirical results proving its effectiveness in terms of the latitude of choice given to the human and the satisfaction of their internal preferences. We further validated this by developing an interactive simulator used for a user study, described in the following chapters.

Chapter 5 is a technical description of an interactive simulator I developed to execute the robot policy generated by the approach described in the previous chapter. This simulator proposes an execution scheme based on the model of execution to run and supervise the robot policy in a 3D simulator. It also allows a human operator to perform actions through intuitive mouse control. Hence, this simulator offers a way to collaborate with a robot following the approach we designed, and it will be used in the next chapter to evaluate the approach.

Chapter 6 presents a user study validating the approach proposed in Chapter 4 using the simulator described in Chapter 5. For this purpose, several scenarios have been designed using a BlocksWorld task, and human participants were asked to collaborate with the simulated robot to evaluate its behavior. We compared our approach with a baseline behavior where the robot always imposes its decisions on the human.

Part II begins with Chapter 7. This chapter describes the InHuS system, addressing decision-making in navigation and challenging robot navigation schemes. This chapter also compares two robot navigation systems using InHuS, proving that our approach effectively challenges robot schemes and allows measuring and comparing human-aware navigation properties.

Chapter 8 presents IMHuS, which complements the previous system to choreograph several agents with group movements and social behaviors. This system has been qualitatively evaluated in an elevator scenario.

Eventually, general conclusions regarding my overall PhD work are shared. Additional materials are provided in the appendix.

List of Publications

As main author

- Anthony Favier, Phani-Teja Singamaneni, Rachid Alami. Simulating Intelligent Human Agents for Intricate Social Robot Navigation. Social Robot Navigation workshop - Robotics: Science and Systems (RSS'21), Jul 2021, Washington, United States.

- Anthony Favier, Phani-Teja Singamaneni, Rachid Alami. An Intelligent Human Avatar to Debug and Challenge Human-aware Robot Navigation Systems. Late Breaking Report - 2022 ACM/IEEE International Conference on Human-Robot Interaction (HRI'22), Mar 2022, Sapporo, Japan.

- Anthony Favier, Shashank Shekhar, Rachid Alami. Robust Planning for Human-Robot Joint Tasks with Explicit Reasoning on Human Mental State.

AI-HRI Symposium at AAAI Fall Symposium Series (FSS'22), Nov 2022, Arlington, United States.

- Anthony Favier, Shashank Shekhar, Rachid Alami. Anticipating False Beliefs and Planning Pertinent Reactions in Human-Aware Task Planning with Models of Theory of Mind. PlanRob Workshop - International Conference on Automated Planning and Scheduling (ICAPS'23), Jul 2023, Prague, Czech Republic.

- Anthony Favier, Shashank Shekhar, Rachid Alami. Models and Algorithms for Human-Aware Task Planning with Integrated Theory of Mind. IEEE International Conference on Robot and Human Interactive Communication (RO-MAN'23), Aug 2023, Busan, South Korea.

- Anthony Favier, Phani Teja Singamaneni, Rachid Alami. Challenging Human-Aware Robot Navigation with an Intelligent Human Simulation System. Social Simulation Conference (SSC'23), Sep 2023, Glasgow, France.

- Anthony Favier, Rachid Alami. Planning Concurrent Actions and Decisions in Human-Robot Joint Action Context. Symbiotic Society with Avatars workshop - ACM/IEEE International Conference on Human-Robot Interaction (HRI'24), Mar 2024, Boulder, United States.

As co-author

- Guilhem Buisan, Anthony Favier, Amandine Mayima, Rachid Alami. HATP/EHDA: A Robot Task Planner Anticipating and Eliciting Human Decisions and Actions. IEEE International Conference On Robotics and Automation (ICRA 2022), May 2022, Philadelphia, United States. ⟨10.1109/ICRA46639.2022.9812227⟩.

- Phani-Teja Singamaneni, Anthony Favier, Rachid Alami. Towards Benchmarking Human-Aware Social Robot Navigation: A New Perspective and Metrics. IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), 2023, Aug 2023, Busan, South Korea.

- Phani-Teja Singamaneni, Anthony Favier, Rachid Alami. Human-Aware Navigation Planner for Diverse Human-Robot Contexts. 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sep 2021, Prague (online), Czech Republic.

- Phani-Teja Singamaneni, Anthony Favier, Rachid Alami. Invisible Humans in Human-aware Robot Navigation. IEEE International Conference on Robotics and Automation (ICRA 2022), May 2022, Philadelphia, United States.

- Phani-Teja Singamaneni, Anthony Favier, Rachid Alami. Watch out! There may be a Human. Addressing Invisible Humans in Social Navigation. 2022

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022), Oct 2022, Kyoto, Japan.

- Olivier Hauterville, Camino Fernández, Phani-Teja Singamaneni, Anthony Favier, Vicente Matellán, et al.. IMHuS: Intelligent Multi-Human Simulator. IROS2022 Workshop: Artificial Intelligence for Social Robots Interacting with Humans in the Real World, Oct 2022, Kyoto, Japan.

- Olivier Hauterville, Camino Fernández, Phani-Teja Singamaneni, Anthony Favier, Vicente Matellán, et al.. Interactive Social Agents Simulation Tool for Designing Choreographies for Human-Robot-Interaction Research. ROBOT2022: Fifth Iberian Robotics Conference, Nov 2022, Zaragoza, Spain.

Chapter 1

Human-Robot Collaboration Context

Contents

This chapter provides more details about the context of my PhD by discussing related fields and works of HRC. This chapter eventually highlights the gap in the literature that motivates my PhD work.

1.1 Multidisciplinarity of Human-Robot Interaction

In [Bartneck 2020], the Human-Robot Interaction (HRI) is considered unique because of the interaction of humans with social robots, which is at the core of this multidisciplinary research field. These interactions usually include physically embodied robots, and their embodiment makes them inherently different from other

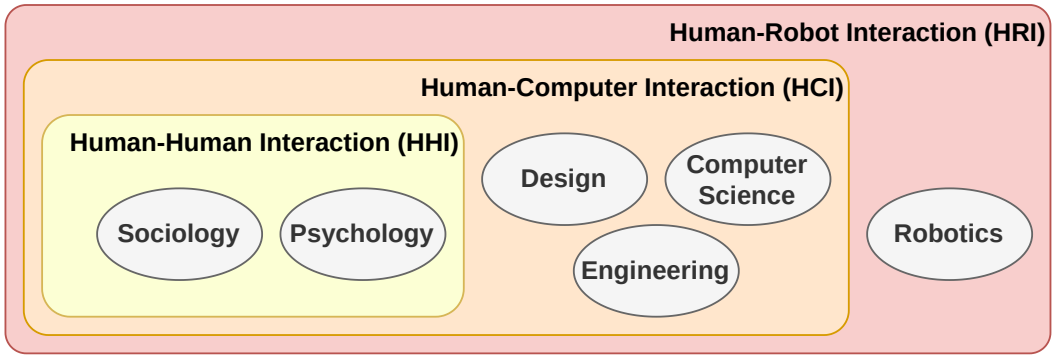


Figure 1.1: Multidisciplinarity of the Human-Robot Interaction field.

computing technologies. Moreover, social robots are perceived as social actors bearing cultural meaning, strongly impacting contemporary and future societies. Saying that a robot is embodied does not mean it is simply a computer on legs or wheels. Instead, we must understand how to design that embodiment, both in terms of software and hardware, as it is commonplace in robotics, and in terms of its effects on people and the kinds of interactions they can have with such a robot.

Overall, Human-Robot Interaction (HRI) focuses on developing robots that can interact with people in various everyday environments. This opens up technical challenges resulting from the dynamics and complexities of humans and the social environment. This also opens up design challenges—related to robotic appearance, behavior, and sensing capabilities—to inspire and guide interaction. From a psychological perspective, HRI offers the unique opportunity to study human affect, cognition, and behavior when confronted with social agents other than humans. In this context, social robots can be research tools to study psychological mechanisms and theories.

As a result, by taking inspiration from Human-Human Interaction (HHI) and Human-Computer Interaction (HCI), HRI is an endeavor that brings together ideas from a wide range of disciplines such as Engineering, Computer Science, Robotics, Psychology, Sociology, and Design. In the following, we discuss some related aspects and works of the mentioned disciplines by categorizing them between HHI, HCI, and HRI as depicted in figure 1.1.

1.1.1 Human-Human Interaction

Many works dealing with interacting with humans take inspiration from Human-Human Interaction (HHI), including research in sociology and psychology. HHI refers to the communication and collaboration between two or more individuals, where humans engage in various forms of social, cognitive, and emotional exchanges. Such interaction can occur through verbal and non-verbal communication, such as speech, gestures, facial expressions, and body language.

Since communication has a crucial role in collaboration, communication theories have been widely studied for a long time [Cherry 1957, Smith 1998]. Professor Al-

bert Meharbian indicates in [Mehrabian 1967] that, when communicating feelings, the words pronounced, the voice tone, and the body language correspond respectively to 7%, 38%, and 55% of the effective communication. These results suggest that communication, especially about feelings, is more complex than "simply" finding the right words. The way these words are pronounced and shared strongly affects the message sent. The Grice's four maxims of conversation [Grice 1975], or Gricean maxims, can also be mentioned and describe the relevance of four distinct aspects of communication: quantity, quality, relation, and manner. These four maxims describe specific rational principles observed by people who follow the cooperative principle in pursuit of effective communication.

HHI also led to studies on collaboration, teamwork, and so-called Joint Action theories [Cohen 1970, Levesque 1990, Cohen 1991]. As put in [Sebanz 2006a], these works aim to describe and understand social interaction whereby two or more individuals coordinate their actions in space and time to bring about a change in the environment, usually trying to reach a common goal.

We must first understand how humans interact with each other before making robots able to interact correctly with humans. However, perfectly mimicking humans is questionable since robots fundamentally differ from humans. Humans have created robots to help and assist them. Thus, HHI should inspire robot design, but additional research is mandatory to determine how to create appropriate interactive and collaborative robots.

1.1.2 Human-Computer Interaction

A first step of artificial interaction and collaboration is the field of Human-Computer Interaction (HCI). HCI is the field of study that focuses on optimizing how users and computers interact by designing interactive computer interfaces that satisfy users' needs. It is a multidisciplinary subject covering computer science, behavioral sciences, cognitive science, ergonomics, psychology, and design principles. Today, HCI focuses on designing, implementing, and evaluating interactive interfaces that enhance user experience using computing devices. This includes user interface design, user-centered design, and user experience design.

This field is made up of four key components: The User, along with their needs, goals, interaction patterns, cognitive capabilities, emotions, and experiences; The Goal-Oriented Task which is the objective or goal the user has in mind; The Interface is about the overall user interaction experience through senses such as touch, click, gesture, voice, display size, and colors; The Context must be taken into account because it influences the interaction.

To produce easy interaction with robots, the study of HCI is relevant and helps to design intuitive, user-friendly interactive robots.

1.1.3 Human-Robot Interaction

Human-Robot Interaction (HRI) is a field of study exploring the design, development, and evaluation of robots interacting with humans in various settings. HRI aims to create robots that can effectively and seamlessly collaborate with humans in domestic environments, workplaces, or other contexts. HRI can be categorized in several domains, not necessarily exclusive. Here are some examples:

Social robotics focuses on social interactions with humans and, thus, explores how robots can understand and respond to human emotions, social cues, and communication styles. A significant amount of work is dedicated to HRI in Healthcare to assist patients, especially the elderly and children with conditions. Those works are also usually linked to emotion-aware robotics focused on recognizing and responding to human emotions using affective computing techniques. A common application is storytelling for children to convey ideas, feelings, or culture.

Human-Centered Robotics emphasizes the importance of considering human needs and preferences. This subfield often involves user studies to ensure and identify if and how robots are user-friendly and can seamlessly integrate into human environments.

Robot Ethics is another central subfield focused on considerations such as privacy, safety, responsibility/accountability, and the impact of robots on society.

Explainable AI and transparency are a growing interest in making decision-making processes more understandable to humans, and thus, help robots be legible, predictable, and acceptable.

Computational HRI, as described in [Thomaz 2016], is the subset of HRI concerned explicitly with the algorithms, techniques, models, and frameworks necessary to build robotic systems that engage in social interactions with humans. This thesis is part of this category because it is focused on developing task planning algorithms and models relevant to a collaborative robot.

Human-Robot Collaboration or Collaborative Robotics focuses on developing robots that work alongside humans in shared workspaces, usually as a team. HRC is the main topic of my thesis and is a vast subject worth delving into. Hence, the following section is dedicated to providing more details about HRC.

1.2 Human-Robot Collaboration

Human-Robot Collaboration (HRC) refers to the synergy and cooperation between humans and robots in shared environments to achieve common goals. In HRC, humans and robots work together, often leveraging their complementary strengths to enhance overall performance and efficiency. According to human desires, the robot can also act in a way that eases and facilitates the human part of the task. This collaborative approach involves close interaction, communication, and coordination between human and robotic agents.

1.2.1 Inspirations & Theories informing HRC

This interdisciplinary field takes inspiration from various theories and fields, as introduced earlier. Nevertheless, three main inspirations can be highlighted:

Belief Desire Intention Model:

The belief-desire-intention (BDI) model was originally developed by Michael Bratman [Bratman 1987]. This model is used in intelligent agents research to describe and model intelligent agents. Straightforwardly, the BDI model is characterized by the implementation of the three notions appearing in its name, i.e., an agent's beliefs (knowledge of the world from the perspective of the agent), desires (objective or goal to accomplish), and intentions (the planned course of actions to achieve the agent's desire).

Shared Cooperative Activity:

Shared cooperative Activity defines prerequisites for an activity to be considered shared and cooperative. The main ones are mutual responsiveness, commitment to the joint activity, and commitment to mutual support. A good example to clarify these prerequisites is a scenario where agents move a table together. Mutual responsiveness ensures that the agents' movements are synchronized. The commitment to the joint activity reassures each agent that the others will not drop their side and quit the joint activity. Finally, the commitment to mutual support deals with possible breakdowns due to one agent's inability to perform part of the plan.

Joint Intention & Action Theory:

Joint Intention Theory proposes that for joint action to emerge, team members must communicate to maintain a set of shared beliefs and to coordinate their actions toward the shared plan [Cohen 1991]. In collaborative work, agents should be able to count on the commitment of other members. Therefore, each agent should inform the others when they conclude that a goal is achievable, impossible, or irrelevant [Hoffman 2004].

1.2.2 Key Aspects

In order to better picture the implications of the above theories, some key aspects of a seamless collaboration are listed and commented on below. This list is not exhaustive, but it highlights some skills that humans naturally exhibit and that a robot must be endowed with to collaborate with them.

Specialization of Roles: There are several human-robot relationships, including supervisor-subordinate, partner-partner, teacher-learner, and leader-follower. These roles can be predefined and fixed during the whole collaboration. The role distribution can also be flexible using weighting functions that allow a continuous change between the roles to adapt to every context and situation.

Establishing shared goal(s): Through direct discussion or inference, agents must determine and agree on the shared goals they are trying to achieve. However, a shared goal is not always necessary and can be established in the middle of a task execution either by the human or the robot.

Allocation of subtasks: After deciding how to achieve their goals, agents must determine what actions and subtasks will be done by each agent and how to coordinate each other. This can be done explicitly before starting the task or reactively done on the fly.

Progression tracking: Agents must be able to track progress toward their goals. That is, they must be able to determine what has been achieved, by whom, and what remains to be done.

Communication: Any collaboration requires communication, verbal or not. Most of the mentioned aspects can or must involve communication. However, it is essential to identify what and how to communicate during the collaboration. Communicating too much can be annoying, while not enough can induce confusion and harm collaboration.

Adaption and learning: On a short-term scale, agents must adapt to each other and the environment. In the longer term, agents must also learn from other partners and the acquired experience.

Ergonomics: It should be intuitive to collaborate and communicate with the robot. This aspect must be considered when designing the robot's hardware and software. Ergonomics is a central aspect of Human-Computer Interaction. Thus, many works from this field can be used in our context or serve as inspiration.

Explainability: This aspect is important for seamless collaboration as the human should be able to understand what the robot is doing and why. This topic is getting more and more attention and is often referred to as Explainable AI. This is especially relevant to counter the *black box* tendency of machine learning where it's impossible to explain a specific decision. Being explainable often enhances predictability, which is also essential for a collaborative robot.

1.2.3 Architectures & Complete Systems

It is important to remember that since a collaborative robot is issued from an interdisciplinary field, its different functionalities and capabilities are usually separated into several dedicated components. These components interact and communicate with each other, forming a complete architecture. Such architectures cover all aspects relevant to exhibiting the robot's behavior, from sensory perception to physical motions, including reasoning processes. Despite developing distinct robotic components, the robot must be considered a whole. Each component cannot be studied entirely independently of other aspects of the complete system. As a result, optimizing how these components interact by design has been the focus of several works proposing robotic architectures.

As Matthias Scheutz, Jack Harris, and Paul Schermerhorn put in [Scheutz 2013], architectures for intelligent robots have improved steadily over the years. Early

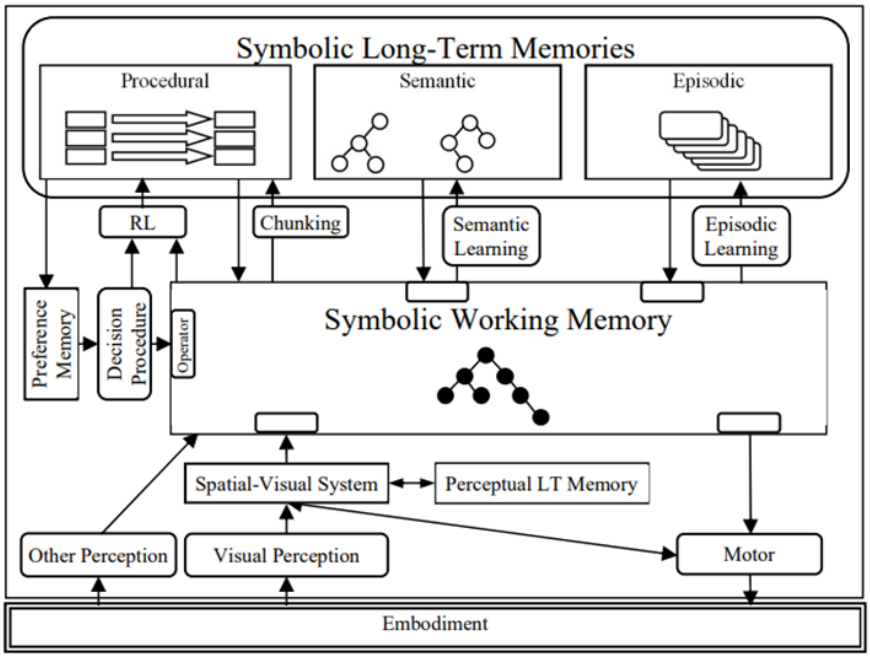


Figure 1.2: The SOAR cognitive architecture.

works like [Alami 1993, Alami 1998, Chatila 1992] propose architectures to provide autonomy to mobile robots, focusing on three levels: decision, execution, and functional. Diverse components that allow robots to negotiate increasingly more complex indoor and outdoor environments have been considered, and they have improved those architectures over the years. As a result, current robot architectures integrate multiple sophisticated algorithms for real-time perceptual, planning, and action processing, from 3D object recognition to simultaneous localization and mapping to navigation and task planning to action sequencing. However, classical robotic architectures like the ones mentioned above typically lack components for high-level cognition, such as general-purpose reasoning and problem-solving. To address this issue, studies on cognitive robot architecture began mainly with the SOAR (depicted in figure 1.2) and ACT-R architectures [Laird 1987, Anderson 2004]. Often based on the structure of the human mind, such cognitive architectures aim to endow robots with high-level capabilities like learning, inferring, and reasoning about how to behave in response to complex goals in complex worlds.

[Lemaignan 2017] identifies relevant collaborative cognitive skills and integrates them into a proposed architecture. The skills include geometric reasoning and situation assessment based on perspective-taking and affordance analysis; acquisition and representation of knowledge models for multiple agents (humans and robots, with their specificities); situated, natural, and multi-modal dialogue; human-aware task planning; human-robot joint task achievement.

[Thierauf 2024] proposes another integrated cognitive robotic architecture more focused on self-awareness. It allows the robot to assess its own performance, identify

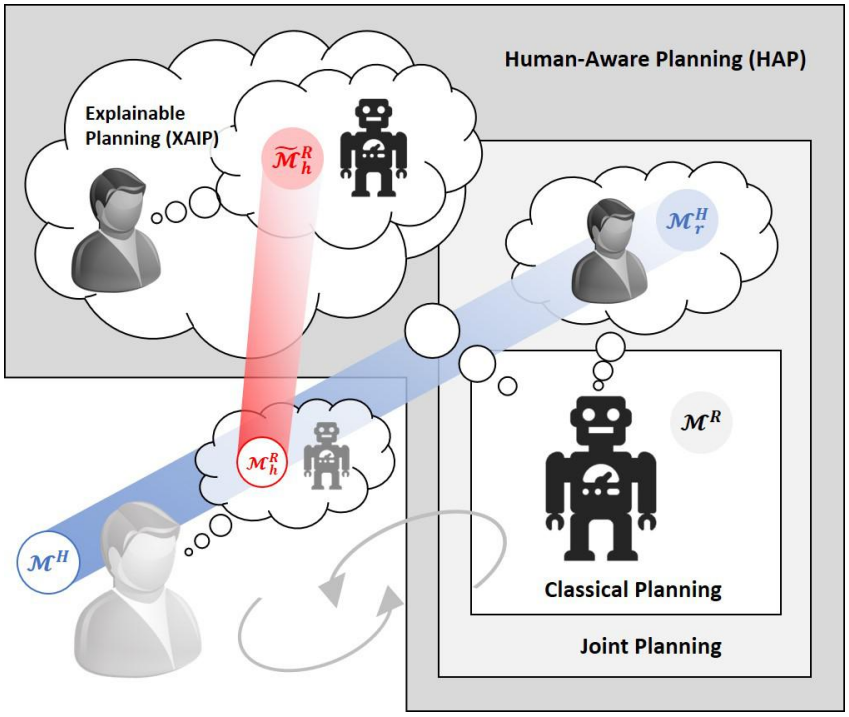


Figure 1.3: Agent models from Chakraborti *et al.* notations.

task execution failures, communicate them to the humans, and resolve them, if possible.

1.3 Models for Interaction

A robotic agent interacting with a human must coordinate its actions with them. Moreover, joint action theory exhibits that humans interacting together represent the task as a whole and plan not only for their actions but also for the actions of other agents. Thus, we think that for a human to perform the most efficient and satisfactory joint task with a robot, this robot must explicitly model human actions and plan not only for its actions but also for the human ones. This is why we present in this section some notations to clarify the different models used in this thesis, and then we present in more detail how to model tasks.

1.3.1 Human and Robot Agents

Chakraborti *et al.* introduced notations to differentiate between the models used in this thesis in their work [Chakraborti 2015], also used in Buisan's thesis [Buisan 2021]. These notations summarize and differentiate elegantly the different models manipulated in the HRI/HRC field. These notations are depicted in figure 1.3. At the bottom are depicted the human agent (on the left) and the robot agent (on the right). When solving a task alone, the robot uses its own model referred to as $\mathcal{M}^R$. This case can be considered as a Classical Planning problem.

Then, $\mathcal{M}_r^H$ is an estimation by the robot of the model of the human. Finally, $\tilde{\mathcal{M}}_h^R$ is an estimation of the robot model the human has. These models are likely to include knowledge about the world from the agent's perspective, an action model describing the agent's capabilities, and an agenda capturing the goal and motivation of the agent.

It is important to remember that when discussing a task planner, it is considered part of the robot. Thus, $\mathcal{M}^R$ is the robot's ground truth. As a consequence, if there is a belief divergence between $\mathcal{M}_r^H$ and $\mathcal{M}^R$, we always consider that $\mathcal{M}^R$ is the truth. Otherwise, it would make no sense to keep this information in $\mathcal{M}^R$ while having access to the one in $\mathcal{M}_r^H$.

1.3.2 Task Modeling

A common way of representing human activity ($\mathcal{M}_r^H$) and interaction with computers at a high abstraction level is by using task models. The hierarchical structure of human activity was first exploited by Annett and Duncan [Annett 1967]. They state that tasks can be described at several levels of abstraction until a certain criterion is met. Each task can thus be refined into subtasks detailing the procedure the human follows to achieve the higher level task. Task modeling has then evolved to introduce interaction with systems, produced and needed information, potential errors, and a wide variety of operator specifying how tasks interact with each other during their execution. Task models are now commonly used in user-centered and user-interface design processes. Most advanced notations include ConcurTaskTrees [Paternò 2004] and HAMSTERS [Martinie 2019]. These models are used to design or evaluate interactive systems. They allow the designer to understand the user task better or to study the user workflow using their system. However, these models contain too little information for a system to be able to reason and make decisions on them (either in planning or acting).

1.3.3 Hierarchical Models

In classical planning, each action of an agent is atomic and needs some conditions to hold in the environment to be executed, and then it changes the environment when applied. The planning process then has to find the right sequence of actions, applicable one after the other, to change the environment and reach a specific goal state.

However, humans tend to work more abstractly and decompose tasks hierarchically into smaller tasks until the action level is reached. In practice, using Hierarchical Task Network (HTN) allows the domain designer to help the plan search by inserting expert knowledge via a hierarchy linking the actions [Erol 1996]. A task network consists of tasks organized in a fully or partially ordered manner, and each task can be either abstract or primitive. Primitive tasks are elementary tasks that can be achieved by performing one associated action. On the other hand, abstract tasks are tasks that first need to be decomposed into other subtasks, "more

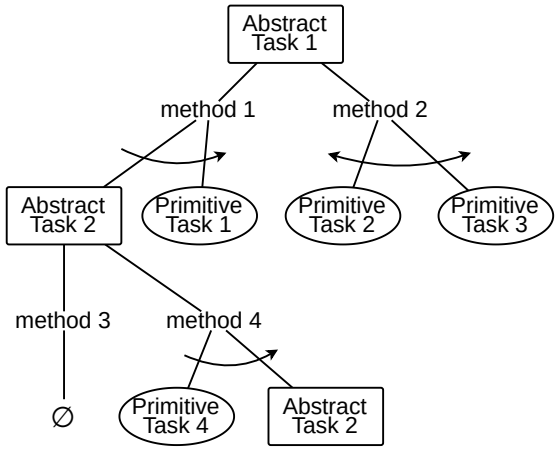


Figure 1.4: An HTN example. Rectangles represent abstract tasks. Each method describes a possible way to decompose an abstract task if its preconditions are met. Methods can decompose abstract tasks into primitive tasks (ellipses) and/or into other abstract tasks. The obtained subtasks can be fully ordered, such as with *method 1* (represented with a one-way arrow), or partially ordered, like *method 2* (represented with a two-way arrow). Note that methods can also decompose a task into nothing like *method 3*, for instance, when the task is already done, and they can be recursive like *method 4*.

primitive". The planner's goal is not to find the sequence of actions to reach a goal but to select recursively for each task the suitable decomposition ending (if possible) with a network of actions applicable from the initial state. Ghallab, Nau, and Traverso name such a process as *planning with refinement methods* [Ghallab 2016]. This planning hierarchy allows the domain designer to guide the search by inserting some expertise into the model and enhancing explainability as the decompositions often offer a semantic to their subtasks. The 'why' of a subtask can usually be answered by going up in the hierarchy, while the 'how' is answered by going down.

To better picture HTN models, a symbolic example is depicted in figure 1.4. Rectangles represent abstract tasks. Each method describes a possible way to decompose an abstract task if its preconditions are met. Methods' preconditions are not necessarily mutually exclusive. Hence, as mentioned above, it is the planner's job to select the most suitable one when several ones are applicable. Methods decompose abstract tasks into primitive tasks, represented with ellipses, and/or into other abstract tasks. The obtained subtasks can be fully ordered, such as with *method 1* (represented with a one-way arrow) where *Abstract Task 2* has to be completed before *Primitive Task 1*. Methods can also be partially ordered as with *method 2* (represented with a two-way arrow) where *Primitive Task 2* and *Primitive Task 3* can be achieved in any order. Note that methods can also decompose a task into nothing like *method 3*, for instance, when the task is already done. Moreover, methods can be recursive like *method 4*.

Now, let us look at an example of a refinement process. We consider an initial

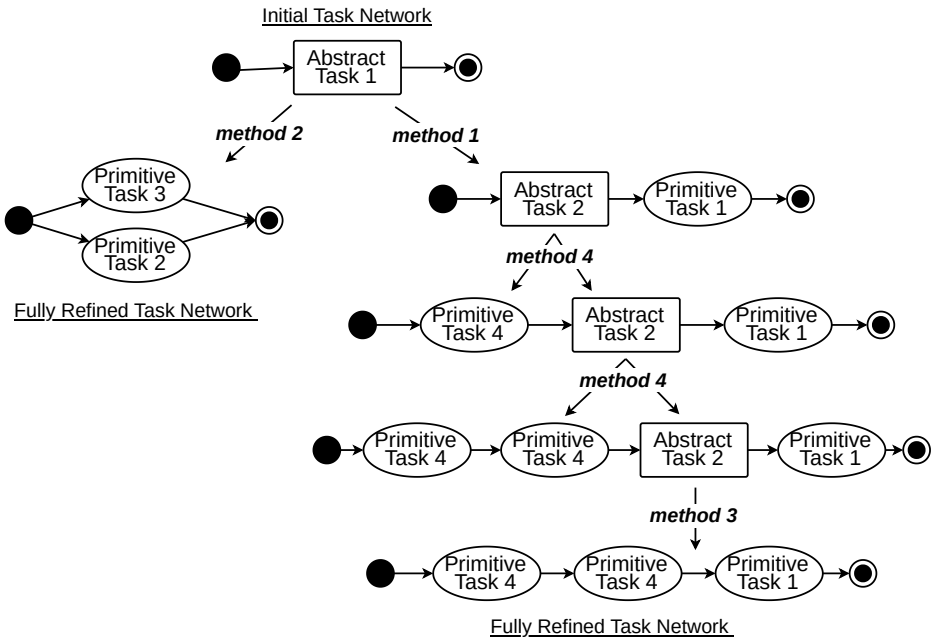


Figure 1.5: Two possible decompositions of a task network using the HTN described in fig 1.4. Both *method 1* and *method 2* can be applied, leading to two different solutions.

task network only composed of *Abstract Task 1*, and we refine it using the HTN described in fig 1.4 until the task network only contains primitive tasks (actions). Initially, *method 1* and *method 2* are both applicable and, thus, are candidates to decompose *Abstract Task 1*. Applying *method 2* leads directly to a fully refined solution task network, including two partially ordered primitive tasks (*Primitive Task 2* and *Primitive Task 3*). On the other hand, *method 1* can be applied, leading to a different task network that still includes abstract tasks. Then, *Abstract Task 2* is recursively decomposed using M4 twice. This could correspond to scenarios like filling a box with two balls. Thus, M4 is only applicable until two more balls are in the box. Eventually, only M3 is applicable and refines *Abstract Task 2* into nothing, leading to another solution task network.

1.4 Task Planning Overview

1.4.1 Classical Planning

As Ghallab, Nau, and Traverso put it, "the purpose of planning is to synthesize an organized set of actions to carry out some activity" [Ghallab 2016]. Classical planning is a type of planning that assumes deterministic and fully observable environments. It involves representing the world as a set of states and actions, with plans derived through state space search algorithms. Actions have preconditions and effects, and planning problems entail finding a sequence of actions to transform

an initial state into a goal state. Classical planning algorithms, including STRIPS, Graphplan, and Fast Downward, utilize heuristics to guide the search efficiently. While well-suited for domains with precise and deterministic dynamics, classical planning may face challenges in handling uncertainty or partial observability, leading to alternative planning approaches for such scenarios.

1.4.2 Task Planning for hrc

Classical planning has been vastly studied and can now solve various problems efficiently. However, the intricate nature of HRC scenarios demands sophisticated task planning methodologies capable of adapting to dynamic environments, understanding human intent, and promoting a fluent exchange of information. Hence, several subfields of task planning have emerged and are used in HRC. Here are a few examples:

- **Hierarchical Task Planning** is a technique that organizes tasks in a hierarchical structure presented in the previous section, allowing for the representation of complex tasks at various abstraction levels. This approach enhances modularity, flexibility, and explainability in task planning, accommodating intricate collaborative scenarios.

- **Mixed-Initiative Planning** leverages the strengths of both humans and robots by allowing for a dynamic allocation of decision-making authority. This technique promotes collaborative decision-making, enabling the system to adapt to the expertise and preferences of each agent involved in the collaborative task.

- **Human-Centric Task Planning** focuses on incorporating human factors into the planning process. This involves understanding human capabilities, preferences, and cognitive load to optimize task plans that align with human collaborators' natural workflows and expectations.

- **Learning-Based Task Planning** has emerged thanks to advancements in machine learning as a frontier in adapting to evolving environments. This technique involves training models to understand patterns in human behavior, enabling the robot to learn and adapt its task planning strategies over time. Such techniques can also be used to predict human behavior and consequently adapt the robot's actions.

- **Probabilistic Task Planning** integrates uncertainty into the planning process, acknowledging the inherent unpredictability of human behavior and environmental factors. By incorporating probabilistic models, this technique enhances the robustness of task plans in dynamic and uncertain collaborative settings.

- **Task and Motion Planning** combines symbolic and geometric reasoning to plan agents' actions. In our context, it can be helpful to consider safety spatial areas near humans and adapt both the robot's motion and decisions.

Overall, **Human-Aware Task Planning** is the process of considering the presence and behavior of humans in the planning and execution of robot tasks. It involves taking into account cues from the shared environment and the dynamics of human-robot interaction. The goal is to generate robot policies that are adaptable, robust, and efficient in crowded and dynamic environments. A detailed presentation and discussion about existing task planning works for HRC is provided in the related work section (2.4) of Chapter 2.

1.4.3 Other Use Cases

Despite being designed for Human-Robot Collaboration, the task planning techniques presented in this work can be used in other interactive contexts.

For instance, instead of planning robot actions, we could plan verbal answers in a dialogue. This approach is used in [De Carolis 2000] and [De Carolis 2001]. Hence, some algorithms and models proposed in this manuscript could be used to anticipate the possible human sentences and plan the relevant robot communications.

Additionally, one could think about a smart environment, e.g., a domotic house, where various sensors and actuators are connected. When humans explore and operate in such an environment, it could be relevant to plan domotic actions according to human behavior, e.g., proactively making coffee, opening stores, activating the robot vacuum cleaner, etc. This context is addressed in [Pecora 2012] to provide proactive human support.

1.5 Human-Aware Navigation

In my work, I studied the decision-making challenge, mainly in the field of task planning. Nevertheless, I also worked on decision-making processes for navigation, more precisely, on how to simulate a navigating interactive human agent endowed with decisional capabilities to challenge robotic navigation systems. Hence, to better understand this contribution, this section provides some context on robot navigation, especially on state-of-the-art techniques and existing benchmarking tools.

As stated in [Buisan 2021], robot navigation aims to make the robot base (the whole robot) move from one place to another while avoiding static and moving obstacles. However, other constraints must be added when the robot has to move in an environment where humans are evolving. Humans should not just be avoided as other moving obstacles, and their psychological and mental state must be taken into account. Hence, the robot should neither move threateningly, block the humans, nor induce drastic changes in their motion. Taking all these aspects into account is what is called human-aware robot navigation.

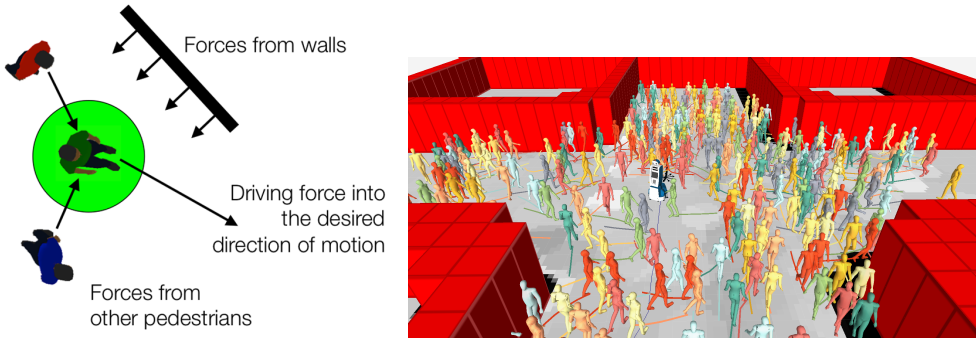


Figure 1.6: Social force model approximating pedestrian motion by a sum of forces.

1.5.1 Robot Navigation techniques

State-of-the-art techniques for robot navigation involve two kinds of motion planners: a global and a local planner. The global planner is in charge of finding the best overall trajectory to lead the robot to its goal and producing a global plan. This planner usually only takes into account static obstacles described by a given map of the environment. Then, the local planner is responsible for producing velocity commands sent to the motor controllers to follow the produced global plan. To produce the velocity commands, the local planner may produce a local plan with only a few seconds of time horizon that follows the global plan while considering obstacles detected in real-time by the robot sensors, including moving obstacles. This way, the robot should reach its goal while reacting to moving obstacles.

However, as explained above, such approaches are insufficient in human-populated environments. Humans must be detected and treated differently during the motion planning process. Human-aware approaches detect and track nearby humans and try to estimate their trajectory to plan the robot's one accordingly. This is achieved in works like [Singamaneni 2021], where the human trajectory is estimated using goal recognition processes and elastic bands methods. Then, the robot's motion is planned using tuned elastic band methods to account for the robot's goal, the estimated human trajectory, and other social norms.

1.5.2 Benchmarking tools and metrics

Where it is easy to benchmark robot navigation on objective metrics like the time to reach a goal, the distance traveled, and the number of collisions [Perille 2020], it is more challenging to benchmark their human-aware properties. First, there is no consensus on the metrics to evaluate a navigation system's human-aware properties. State-of-the-art metrics involve proxemics [Samarakoon 2022]. However, other relevant metrics, such as the deviation imposed on human motion and the feeling of threat produced, can be used. Also, finding a usable system that will effectively challenge a Human-Aware Navigation (HAN) system is challenging.

Typical approaches involve reactive-only techniques such as social force models [Helbing 1995, Chen 2018]. The social force model assumes that a sum of differ-

ent forces can approximate pedestrians' acceleration, deceleration, and directional changes, each capturing a different desire or interaction effect. For instance, as depicted in figure 1.6, one force corresponds to the acceleration towards the desired velocity of motion; second, repulsive forces reflect the agent keeping a certain distance from other agents and obstacles; third, attractive forces represent the goal and motivations of the agent. Eventually, using standard physics equations, the sum of these dynamic forces describes the agents' motion. Such reactive models are easy to use and efficient for crowd simulations. Interestingly, in crowded or evacuation scenarios, social force models exhibit several so-called "self-organization phenomena" such as lane formation, zipper effect, intermittent flow, or turbulence. Unfortunately, such a model can perform very poorly in intricate and non-crowded scenarios involving some decision-making. Thus, there was a lack of intelligent simulated agents to challenge effectively HAN system. A few recent works also propose simulating human agents endowed with some reasoning processes. However, I did not find the time to compare my contribution with theirs, and it remains an interesting possible future work. Nevertheless, this shows that this is a subject of interest. Some related works will be discussed in Chapter 6.

Part I

Human-Aware Task Planning

Chapter 2

A Human-Aware Task Planner Emulating Human Decisions and Actions (HATP/EHDA)

Contents

This chapter presents the HATP/EHDA task planner. This planner has been the keystone of most of my work. Hence, the reader should understand the motivation and methods of this task planner.

2.1 Introduction

I was introduced to task planning with the work of a PhD student from my lab, Guilhem Buisan. I slightly contributed to the original version and then proposed two extensions of his work. However, Guilhem mainly designed and implemented this novel **H**uman-**A**ware **T**ask **P**lanning approach dedicated to Human-Robot Interaction (HRI), which plans the robot's actions while estimating and **E**mulating the **H**uman **D**ecisions and **A**ctions, namely **HATP/EHDA**.

We believe this planning approach suits the needs of HRI scenarios well; thus, it became a laboratory to address relevant challenges of task planning for HRC. Two of my main contributions include addressing such challenges and implementing the solutions as extensions of HATP/EHDA. Consequently, it is essential to understand this work's motivation and methods well before introducing my proper contributions. This section introduces, motivates, and explains the HATP/EHDA approach

as a background to the other chapters. A detailed description of this prototypical planner is already given in Buisan's thesis [Buisan 2021]. Thus, large parts of this section are directly retrieved from Buisan's thesis, but they are essential to have in mind. Some notations are adapted to match the descriptions of my contributions in the following chapters.

2.2 A Hierarchical Agent-Based Task Planner (HATP)

My work and HATP/EHDA are part of a line of work that started with the Hierarchical Agent-based Task Planner (HATP) [Alili 2009, Lallement 2014].

Based on Hierarchical Task Network (HTN), this planner can elaborate a multi-agent plan based on a single HTN tree. Moreover, it maintains one belief base per agent, allowing the writer to write task decomposition rules and action preconditions and effects in any agent belief base. This approach produces a joint plan that includes actions from all involved agents.

Finally, to make HATP suitable to HRI scenarios, specific mechanisms to compute costs and filter plans have been included. To this end, HATP allows balancing efficiency, wasted time, agent effort, intricacy, and undesirable states or sequences. This way, social rules can be defined to produce a collaborative plan that humans will likely accept.

However, HATP assumes a shared goal has been established between humans and robots before planning. The generated plan will be shared with and accepted by humans before execution. Indeed, HATP does not represent humans as agents having separate decision-making processes that may lead to diverging plans without robot communication. Hence, any human deviation from their generated stream of action needs supervision to perform repair action or request a replanning. Such an approach can work well, but it assumes that communication can easily be done at any point in the plan. However, this assumption is not always verified for various reasons, *e.g.* noisy environments, making communication costly. Additionally, any human deviation, *e.g.* due to inattention, will put the robot in a failure state, which needs to be fixed before continuing (replanning).

2.3 Rationales of the HATP/EHDA Approach

The HATP approach produces what is estimated to be the best joint plan for solving the task. Hence, this approach assumes that humans will likely accept and follow the produced plan. This approach leaves no room for online human decisions and assumes a fully committed human through a previously established shared goal. This approach also considers one shared task representation and knowledge base. To cater to the limitations of HATP, I participated in the development of a new approach, HATP/EHDA, which tries to satisfy several objectives:

1. **Plan without assuming a prior shared goal.** In HRI scenarios, the robot and the human do not always share a goal. The robot can, for example, plan

to perform a task around humans that are not involved at first, or it may be requested by a human to do a task without wanting to take part in it. HATP/EHDA can balance between integrating the sharing of a goal with a human (assumed to be collaborative) in the plan and making the robot do the task alone or integrating the eventuality to ask for punctual human help.

2. **Model the human decision processes.** When taking part in a task, a human (assumed willing to collaborate with the robot) will also plan to reach their (potentially shared) goal. HATP/EHDA must be able to account for this to provide plans that are expected and explainable by the human partner.

3. **Help the human decisions, but not compel them.** Unlike HATP, HATP/EHDA should account for human decision-making flexibility. While modeling the human decision processes, it is possible to narrow down the possible human actions, and the generated plans must help the supervision (execution of the plan) avoid replanning or repairing during the execution by considering several human actions.

4. **Model the potential human reactions.** It is possible to predict that the human may react to some situations, interrupting or helping their current task. Two causes have been identified for these reactions. First, they can result from specific world states humans perceive and interpret. Then, they can also originate from explicit communications issued by the robot. These communications can either be a belief alignment, updating the human knowledge and impacting their decisions, a request to perform a specific action, or a request to help the robot with a shared goal, needing the human to plan for it.

5. **Act and decide on the different agents' beliefs.** It is crucial to be able to represent actions as having different effects on the beliefs of the robot or the human. Indeed, some robot actions are partially or not observable by humans; humans cannot know the complete new world state when performing them. Besides, these effects and their observability often depend on the current world state, whose representation must be supported by the planner. Then, planning decisions may require reasoning on both the robot's and human's beliefs. This is especially true with communication actions aiming to align knowledge or ask questions. Finally, some actions of pure decision have no direct effect on the world but only on the internal beliefs of the agents. For example, observation actions will only update the beliefs of the agent doing it.

6. **Decide not only on the world state but also on the decision processes of the agents.** Some decisions made during the planning process require access to the agents' beliefs representing the world state and the estimation of their planning processes. For example, the decomposition of a task by the robot may be impossible if some other task is already performed in its partial

plan. Other decisions may also need the estimation of the current human planning process. For example, if it were estimated earlier in the plan that the human would perform a specific task decomposition, the planner would assign a complementary task to the robot.

7. **Adapt to the human experience, trust, and preferences.** We also want the planning process to be adjusted depending on the actual human it is planning with. It must perform its planned search differently, whether the human has the habit of performing this particular task with the robot or not. Moreover, the human model can be adjusted to the human's trust in the robot and their preferences.

2.4 Related work

An approach to solving a collaborative task is to produce a joint plan that includes coordinated robot and human actions. This plan must be shared, accepted, and followed to solve collaboratively the common goal. This approach is used in the HATP planner presented above. By assuming the plan must be followed, the human cannot deviate from the generated plan. Hence, this approach assumes that the human is controllable. In [Johannsmeier 2016], the authors use the same assumption and propose a task allocation framework for human-robot collaborative assembly line tasks. They propose representing the task through an AND/OR graph and solving the optimal sub-task allocation problem considering a given cost function. Action sequences are extracted from this allocation and then shared with the respective agents to be executed. Hierarchical and concurrent hybrid state machines handle the execution of these sequences. This copes with unpredictable events likely to happen in dynamic and partially known environments, especially in the presence of humans.

Additionally, some works are focused on HRC's psychological aspect, like the plan's acceptability and explainability. This is beneficial to the approach described just above. In [Chakraborti 2017], the authors propose to improve the explainability of a robot plan by using both a robot model $\mathcal{M}^{\mathcal{R}}$ and the estimation of the model the human has of it $\mathcal{M}_h^R$. This approach, called model reconciliation, aims to make identical the optimal plans generated using both models, i.e., $\mathcal{M}^{\mathcal{R}}$ and $\mathcal{M}_{\langle}^{\mathcal{R}}$. They define a list of operators to modify the different models until the plans match. Although this approach interestingly improves the explainability of the robot's plan, it does not consider the two agents to collaborate directly. Indeed, the plan produced only contains robot actions.

Producing a joint plan that considers the human controllable is efficient and acceptable in industrial contexts. Indeed, the human is working and thus highly committed and focused on the task. Even if humans appreciate flexibility, in this context, their priority is instead task efficiency. However, this assumption can become burdensome for humans in other contexts, such as household ones, where

humans are likely to be distracted, change their minds, and tend to prioritize minimizing their effort and flexibility instead of task efficiency.

As a result, some approaches started to consider a distinct human model $\mathcal{M}_h^R$ to plan the robot action. A first approach, described in [Hoffman 2007], proposes an adaptive action selection mechanism for a robotic teammate, making anticipatory decisions based on the confidence of their validity and their relative risk. They demonstrate improved task efficiency and fluency compared to a purely reactive process. The human behavior is modeled with a First-Order Markov Process and learned through Bayesian estimate. The probabilities constituting the human model are used in the cost evaluation of different plans, eventually leading to a robot action selection, producing an adaptive and proactive robot plan. In [Unhelkar 2020] is proposed an approach based on Partially Observable Markov Decision Process (POMDP) called CommPlan. The POMDP is built using a user-defined Markov Decision Process (MDP) representing the collaborative task and an Agent Markov Model (AMM) representing the human decision-making process. Solving this POMDP produces a robot policy that decides when the robot has to communicate about its beliefs, when to question the human about theirs, and when to ask the human to perform an action. Besides, the AMM is not only specified by an expert modeler but also refined during the interaction via learning. However, the approach considers the human model as an oracle on which reasoning is hardly possible.

Other works extend the use of a distinct human model to explicitly predict and anticipate the actions the human is likely to perform. Then, for each possible human action, the best robot's actions are determined to account for human unpredictability. This is the approach used in HATP/EHDA. This other work [Buckingham 2020] also uses this approach. They propose a unified scheme to cope with collaborative, adversarial, and non-involved human agents. This scheme considers given mental models for each human co-present with the robot, which might interact with the latter. These mental models are queried to estimate a set of actions each agent is likely to perform given a state. The robot's actions are planned according to these estimated actions to reach the robot's goal and potentially achieve the human agents' goal. In doing so, the robot's actions can influence the human ones without explicit communication, helping the robot achieve its goal. HATP/EHDA similarly queries a human mental model and uses an AND/OR graph where OR nodes represent possible robot actions in a state and AND nodes represent the possible human actions estimated by the models. However, despite saying the model can be generic, this work uses a basic breadth-first search planner to produce a set of minimal-cost plans solving the estimated human goal. Then, they extract the first actions of each plan to produce a set of actions the human is likely to perform. No details are given on the cost evaluation. This approach does not seem to consider human actions that are sub-optimal but still probable, which can be due to inattention. Moreover, the robot actions are selected by using a Min-Max approach on the AND/OR graph, minimizing the worst-case. This approach works well in adversarial setups and still allows cooperative human interventions. However, it is unclear how these

cooperative actions are taken into account in the robot decision process. Assuming that the human will be adversarial, the robot might make "wrong" decisions, possibly preventing an efficient and optimal cooperative execution. HATP/EHDA minimizes the average cost of all possible human decisions, optimizing uniformly for any human decisions. This approach considers humans congruent, rational, and cooperative but not necessarily involved in the robot's task. Hence, HATP/EHDA does not account for adversarial humans but addresses the other cooperation types better. [Koppula 2016] also explicitly estimated the most likely human actions to generate the robot policy, but they use a probabilistic and learning approach. They propose a two agent collaborative MDP model and learn robot policies by taking into account the actions that can be performed by the human. They represent the environment in terms of the object affordances and learn the activity model from RGB-D videos of a human performing the activities. Then, they use this learned task model in a distributed Q-learning algorithm to learn the robot policy for a given new environment. The human capabilities are captured in a MDP. Thus, the possible human actions are estimated by balancing the human habits and the best (ε-optimal) action given by the MDP.

Another line of work, sometimes also using human models, is focused on online and reactive planning. The robot's behavior is decided online using planning techniques but on limited horizons. This produces robot behaviors that are adaptive to human decisions and potentially unexpected environmental changes and events. However, due to the limited horizon to maintain a real-time reactive behavior, the robot behavior's optimality is not guaranteed and might lead to dead ends. Nevertheless, these online approaches usually robustify the robot's behavior and are reactive to failures. Therefore, they could/should complement offline planning approaches like HATP/EHDA. The two-sided work [Sanelli 2017] is in this line of work. First, they propose a conditional planning system that considers uncertainties, primarily due to human agents. This planning approach is based on planner Contingent-FF [Hoffmann 2005]. Then, they propose a component to translate these plans into a Robust Petri-Net Plan (PNP) to handle their execution. This translation is inspired by work [Iocchi 2016], which improves plans with execution rules. Eventually, these plans are executed with an existing module called PNPRos [Ziparo 2011]. Interestingly, the expected human actions are transformed into sub-Petri Nets where the robot elicits the action (*e.g.*, via verbal communication) if the human does not perform it by themselves. However, this approach does not model either reason on a distinct human agent model. Hence, the human reasoning process, goal, and belief are not modeled. This means that the interaction is limited to requesting the human to perform single actions without setting a proper high-level joint goal. This can be efficient in some situations, like a service robot providing information and requesting answers to some questions, such as the examples presented by the authors. However, repeated punctual interventions to perform a more extended task collaboratively can become unpleasant for the human. In [Darvish 2021], they propose a hierarchical human-robot cooperation architecture called FlexHRC+ designed to provide collaborative robots with an extended de-

gree of autonomy when supporting human operators in high-variability shop-floor tasks. This online architecture is organized into three levels: perception, representation, and action, producing robust, adaptive, and sometimes proactive collaboration. They use hierarchical AND/OR graphs where arcs are sub AND/OR graphs to reduce the complexity of task descriptions, especially when including repetitive subsequences of action (like mounting four table legs). This representation is close to the HTN one used in HATP/EHDA. A reactive human-aware task planner is proposed in [Fusaro 2021], taking advantage of the Behavior Tree paradigm. The approach plans the robot's actions online by minimizing weighted costs based on duration, ergonomics, and distance, which are updated online. Therefore, the robot can adapt to dynamic changes in the environment and to human intentions, motions, decisions, and availability. This approach permits considering different levels of engagement between robots and humans: coexistence, cooperation, and autonomous task execution.

Finally, in [Izquierdo-Badiola 2022], the authors combine the production of a joint plan to solve a common goal with the reactive aspect of previous approaches. They use a so-called "*agent state*" to model the human mental state and translate it into action costs in a PDDL domain. This model comprises the *Capacity* evaluating if the human is capable of performing an action at the specified location without difficulty, the *Knowledge* evaluating if the human has all necessary knowledge to perform their assigned actions, and the *Motivation* indicating if the agent is committed towards the common goal, active and not distracted. In their approach, the *agent state* is sensed and updated during execution. Then, whenever changes in the *agent state* are detected, a replan is triggered to consider those changes in the plan, often inducing a reassignment of the actions and avoiding the probable failure of the initial plan. However, the authors do not provide a method to effectively sense and estimate the *agent state* but prove its usefulness by simulating its acquisition.

2.5 Running Example

To highlight the potential of our approach, we present a cube-stacking scene as an example. The scene is depicted in four different scenarios in 2.1. The goal consists of stacking the colored cubes on the empty marks to match the colors on the figure's left. All cubes placed in the middle of the table are reachable from anywhere. However, when close to one side, a cube is only reachable from this specific side. Notice that one of the required red cubes is located on the opposite side of the table and cannot be grabbed by the robot in the initial state.

A first human, called *H1*, requests the robot to stack the colored cubes to match the goal pattern given. In the first case (a), *H1* set a shared collaboration goal by requesting the robot to stack the cubes together with them. However, *H1* can also just give their request to the robot and leave. In such a case, the robot has three possibilities. First, in (b), the robot can solve the task alone but must move to the other side of the table, which is slow and costly. Secondly, in (c), the robot can ask

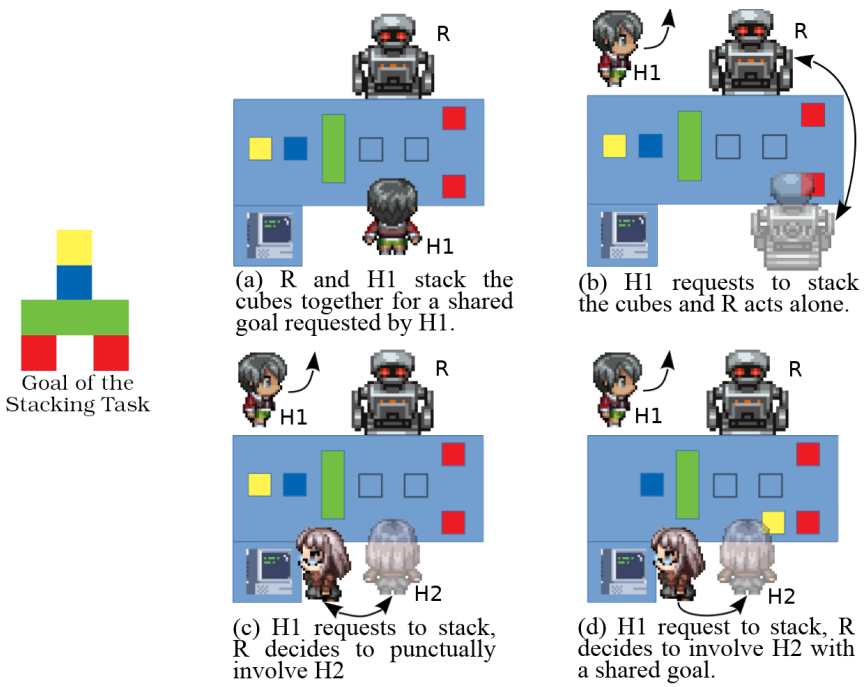


Figure 2.1: Cube stacking scene: A different plan is selected for each scenario, involving nearby humans in the least disturbing way possible

.

another nearby human, *H2*, for punctual help with the unreachable red cube. *H2*'s reaction can be to put the red cube directly in the stack or only make it reachable to the robot. Finally, the robot can set a shared goal by requesting *H2* to help it build the stack together.

The HATP/EHDA approach explores and evaluates all these kinds of scenarios to produce the robot policy.

2.6 Formalization

This section describes the formalization introduced by the HATP/EHDA planning approach and used or adapted in my contributions. It includes descriptions of the problem specifications and the solution produced by the planner.

Problem Specification

The notations from Buisan's thesis have been adapted to match the notations in this thesis and ease readers' comprehension. We start from the classical planning formalization described in [Ghallab 2016].

Definition 1 ***(Classical Planning Domain*** Σ***.)*** *A* classical planning domain *is a state-transition system in the following form:* $\Sigma = (S, A, \gamma)$*. S is a finite set of* states *in which the system may be, A is a finite set of* actions *that the agents*

may perform, $\gamma : S \times A \to S$ *is a* state-transition function. *Each* state $s \in S$ *is a description of the properties of various objects in the planner's environment.*

To represent the objects and their properties, we will use two sets B and X: B is a set of names for all the objects, plus any mathematical constants representing the properties of those objects. X is a set of syntactic terms called state variables, s.t. the value of each $x \in X$ depends solely on the state s.

Definition 2 ***(State-variable*** x***.)*** *A* state-variable *over* B *is a syntactic term* $x = sv(b_1, ..., b_k)$*, where* sv *is a symbol called the state variable's name, and each* b_i *is a member of* B *and a parameter of* x*. Each* state-variable x *has a range,* $Range(x) \subseteq B$*, which is the set of all possible values for* x*.*

Here is the description of the sets B and X for the stacking example given in the introduction:

$$B = Entities \cup Locations \cup Reachability \cup Booleans \cup \{\mathsf{nil}\}$$
$$Entities = Agents \cup Cubes$$
$$Agents = \{\mathsf{R}, \mathsf{H}\} \quad \backslash\backslash\ \mathsf{R} : robot,\ \mathsf{H} : human$$
$$Cubes = \{\mathsf{red1}, \mathsf{red2}, \mathsf{green1}, \mathsf{blue1}, \mathsf{yellow1}\}$$
$$Locations = \{\mathsf{base1}, \mathsf{base2}, \mathsf{bridge}, \mathsf{top1}, \mathsf{top2}\}$$
$$Reachability = \{\mathsf{middle}, \mathsf{side\_h}, \mathsf{side\_r}\}$$
$$Booleans = \{\mathsf{true}, \mathsf{false}\}$$

$$X = \{at(e), holding(a), solution(l) \mid e \in Entities, \varphi \in Agents, l \in StackLocations\}$$
$$Range(holding(\varphi) \mid \varphi \in Agents) = Cubes \cup \{\mathsf{nil}\}$$
$$Range(at(\varphi) \mid \varphi \in Agents) = \{\mathsf{side\_h}, \mathsf{side\_r}\}$$
$$Range(at(c) \mid c \in Cubes) = Reachability \cup Locations$$
$$Range(solution(l) \mid l \in StackLocations) = Cubes$$

Definition 3 ***(Variable value assignment function*** val***.)*** *A* variable value assignment function *over* X *is a function* val *that maps each* $x_k \in X$ *into a value* $z_j \in Range(x_k)$*. With* $X = \{x_1, ..., x_n\}$*, this function can be written as a set of assertions:* $val = \{x_1 = z_1, \ldots, x_n = z_n\}$*. For legibility purposes, we use the following notation to access the value* z_j *of a state-variable* x_k *in the state* s_i*:* $val_i(x_k) = z_j$*.*

Definition 4 ***(Action*** a***.)*** *An* action *is a tuple* $a = (head(a), pre(a), \mathit{eff}(a))$ *where* $head(a)$ *is a syntactic expression of the form* $act(z_1, ..., z_k)$ *where* act *is a symbol called the* action name *and* $z_1, ..., z_k$ *are variables called parameters.* $pre(a) = \{p_1, ..., p_m\}$ *is a set of preconditions, each of which is a literal. And* $\mathit{eff}(a) = \{e_1, ..., e_n\}$ *is a set of effects, each of which is an expression of the form:* $sv(t_1, ..., t_j) \leftarrow t_0$ *with* t_0 *being the value to assign to the state variable* $sv(t_1, ..., t_j)$*. We note* $agt(a)$ *the agent performing the action* a*.*

The problem specification of HATP/EHDA is a pair of two distinct human and robot models as follows: $\mathcal{P} = \{\mathcal{M}^H, \mathcal{M}^R\}$. Each model $\mathcal{M}^\varphi$, for an agent φ, comprises the following:

- **Name** ($name^\varphi$): being the name of the agent. Hence, either "robot" or "human".
- **Beliefs** (val^φ): estimation of the world state from the agent's perspective.
- **Agenda** (d^φ): capturing the personal and/or shared goals of the agent, currently implemented as a task list/sequence but could be generalized to partially ordered task networks.
- **Partial Plan** (π^φ): storing the current partial plan on an agent. This is empty at the beginning and filled during the planning process.
- **Action Model** (Λ^φ): encoding the capabilities of the agent and used to estimate the next actions of the agent given a goal and a world state. Here, it is described by a Hierarchical Task Network (HTN) and thus a set of operators and methods.
- **Triggers** (Tr^φ): describes the reactions the agent may have which might update their agenda. The agent may react to a specific world state, event sequence, or explicit communication. For instance, consider a scenario where another agent is suddenly handing over an object to the agent. This event has nothing to do with the agent's goal, and thus, the next agent action extracted from the Action Model might not consider the other agent. However, a natural reaction to this situation is to grab the handed object. Thanks to the Triggers mechanic, we can model and predict that whatever the agent is doing, the agent will grab the object when given.

Note that most of the models' elements are static during the planning process. Only the Beliefs (val^φ), the Agenda (d^φ) and the Partial Plan (π^φ) of each agent evolve during the planning process. That is why we define specifically an **agent state** in Definition 5.

Definition 5 ***(Agent state σ.)*** *An* agent state σ^φ *comprises the dynamic information evolving during the planning process about an agent φ. It consists of the agent's beliefs, agenda, and partial plan. Hence, the* agent state *is written as follows:* $\sigma^\varphi = \{val^\varphi, d^\varphi, \pi^\varphi\}$.

Definition 6 ***(Agent model $\mathcal{M}$.)*** *An* agent model $\mathcal{M}^\varphi$ *comprises all information regarding an agent φ. It consists of the static information, such as the agent's name, action model, and triggers, and the dynamic information gathered in the* agent state. *Hence, an* agent model *is formalized as* $\mathcal{M}^\varphi = \{name^\varphi, \sigma^\varphi, \Lambda^\varphi, Tr^\varphi\}$.

The planner uses two agent models, one for the human and one for the robot. Despite their identical structure, the two models have a fundamental difference: one is a controllable agent and not the other. Indeed, the human model is only used to speculate on human decisions and actions in given situations. Then, the robot model is used to plan the robot's actions according to the estimated human actions. Note that human decisions can still be influenced by the robot's actions, but they cannot be compelled. It is also important to remember that the two agents are not equivalent. The robot's role is to help, assist, and facilitate humans. Therefore, it should exhibit pertinent, legible, and acceptable behavior.

Definition 7 ***(State s.)*** *A* state $s_i \in S$ *is a tuple composed of two* agent states *capturing the state in which the planning problem is, s.t.* $s_i = (\sigma_i^H, \sigma_i^R)$.

From the robot's perspective, the state of the world is captured by the variable value assignment function $val_i^R \in$
$agentstate[R][i]$. Since the planner is assumed to be part of the robot, the robot's beliefs are assumed to be the ground truth and are sometimes noted as val_i. Similarly, $val_i^H \in \sigma_i^H$ represents the estimation of val_i from the human's perspective, also called the estimated human beliefs. Therefore, we can define estimated human **false belief** in Definition 8.

Definition 8 ***(False belief.)*** *We say that a state* $s_i \in S$ *contains* false beliefs*, or* belief divergences*, if* $\exists x_j \in X, val_i^H(x_j) \neq val_i^R(x_j)$.

Be careful not to confuse ***states*** and ***beliefs***. A *state* is a state in which a given planning problem is. It comprises the robot and human *agent states*. Each *state* is connected to another through agent actions. On the other hand, the *beliefs* refer to the state of the world from an agent's perspective. Both human and robot *beliefs* are part of a *state*.

As an example, considering the presented stacking example, the associated initial state s_0 would be as follows:

$$
\begin{aligned}
s_0 &= \{\sigma_0^R, \sigma_0^H\} \\
&\quad \sigma_0^R = \{val_0^R, d_0^R, \pi_0^R\}, \quad \sigma_0^H = \{val_0^H, d_0^H, \pi_0^H\} \\
&\quad d_0^R = \{Stack\}, \quad d_0^H = \{\} \\
&\quad \pi_0^R = \pi_0^H = \{\} \\
&\quad val_0^R = val_0^H = \{at(\mathsf{R}) = \mathsf{side\_r},\ at(\mathsf{H}) = \mathsf{side\_h},\ at(\mathsf{red1}) = \mathsf{side\_r}, \\
&\qquad at(\mathsf{red2}) = \mathsf{side\_h},\ at(c) = \mathsf{middle} \mid c \in Cubes \backslash \{\mathsf{red1}, \mathsf{red2}\}, \\
&\qquad holding(\mathsf{R}) = holding(\mathsf{H}) = \mathsf{nil}, \\
&\qquad solution(\mathsf{base1}) = \mathsf{red},\ solution(\mathsf{base2} = \mathsf{red}),\ solution(\mathsf{bridge} = \mathsf{green}), \\
&\qquad solution(\mathsf{top1}) = \mathsf{blue},\ solution(\mathsf{top2} = \mathsf{yellow})\}
\end{aligned}
$$

Note that our model permits interaction with only one human at a time. Hence, in scenario (a) on figure 2.1, $\mathcal{M}^H$ corresponds to *H1*. In all other scenarios, it corresponds to *H2*. Therefore, d_0^H is empty except in scenario (a), where *H1* establishes a shared goal. Also, the sets B and X have been slightly modified for legibility reasons because the cubes are, in fact, explicitly associated with colors used in the task decompositions. As a result, the solution is expressed using colors and not the cube names. Finding the exact cube to place is described in the action models.

Planning and Estimating Actions

As mentioned above, the two models $\mathcal{M}^R$ and $\mathcal{M}^H$ are fundamentally different. The human one is used to estimate the actions that the human is likely to perform in a given situation. The robot one is used to plan the best robot actions according to the estimated human ones. Nevertheless, to simplify the description, we tend to refer to both cases as "estimating an agent's next actions".

The exact process of estimating the next actions that an agent $\varphi \in Agents$ is likely to perform in a state $s_i \in S$ will be detailed later. Here, we only consider an overview to introduce some notations. The process roughly consists of using the agent's static action model (Λ^φ) and the dynamic agent's beliefs (val_i^φ) to refine the agent's agenda (d_i^φ). This refining process returns a so-class *refinement* defined in Definition 9.

Definition 9 ***(Refinement*** $ref.$***)*** *A* refinement *is a list of 2-tuples for each estimated action, a, and the associated new agenda, d, after being refined, s.t.,* $ref(d_i^\varphi, val_i^\varphi) = \{(a_1, d1), ..., (a_k, dk)\}$. *It is computed using the agent's action model* Λ^φ.

In our cooking example, we obtain the following refinement if the starting agent is the human:

$$ref(d_0^H, val_0^H) = \{(add\_salt(), d1), (move\_to(\mathsf{kitchen}), d2)\}$$

The execution of an action a in HATP/EHDA is seen and known by both agents. Thus, $\forall \varphi \in Agents, \forall x \in X$:

$$val_{i+1}^\varphi(x) = \begin{cases} w, & \text{if } x \leftarrow w \in \mathit{eff}(a) \\ val_i^\varphi(x), & \text{otherwise} \end{cases} \tag{2.1}$$

Solution description

HATP/EHDA produces a robot policy extracted from an implicitly coordinated joint solution tree. This solution tree is an AND/OR tree where AND nodes correspond to human decisions, all possible human choices are preserved, and each OR node corresponds to a robot action selected among the possible robot actions. This representation assumes a turn-taking fashion where agents act one after another alternatively. The solution is formally defined in definition 10.

Definition 10 ***(Implicitly Coordinated Joint Solution.)*** *The solution for P is represented as a tree, i.e. $G = (V, E)$. Each vertex ($v \in V$) represents the robot's belief state, starting from the initial belief. Each edge ($e \in E$) represents a primitive task that is either a robot's action a^r or a human's estimated and emulated action a^h. G gets branched on the possible choices (a_1^h, a_2^h, ..., a_m^h).*

In practice, it produces a tree of sequential actions where every human action is succeeded by one optimal robot action. Additionally, a new branch is created for every estimated possible human action, ensuring it covers all possible human choices. Each branch is a possible course of alternating robot and human actions leading to the goal. Each branch in the solution tree is a sequence of primitive actions, say $\pi = (a_1^h, a_2^r, a_3^h, ..., a_{k-1}^h, a_k^r)$, that must satisfy all the solution conditions of ${}^m athcal P$. Here, each a_i^h represents a choice, often out of several, the human could make. And each a_{i+1}^r is the optimal robot action according to a_i^h.

2.7 Exploration

To start planning, HATP/EHDA must be given the two action models (the robot and the human HTNs), the initial beliefs of both agents (which can differ), and the initial agenda of both agents. The initial agenda of the robot represents the task to decompose, while the agenda of the human represents any task the human is estimated to be committed to. If a shared goal has been established prior to planning between the robot and the human (*e.g.*, the human asking to perform a task with the robot), the agenda of both agents will be filled with the same task.

The planning process is done in three parts: (1) both HTNs are explored in a turn-taking fashion, resulting in a valid joint plans tree; (2) based on this tree, robot actions are selected according to action, plan-wide and social costs, resulting in a conditional plan, where at each step multiple human actions can be performed but only one robot action is set; (3) causal and threat links are added between actions of the conditional plan to ease its execution.

The robot HTN exploration is a pretty standard depth-first algorithm. The first task λ from its agenda d^R is popped, then if it is an abstract task $\lambda \in Ab$, all the applicable methods are applied, and their results are prepended to the agenda, thus giving new agents state (with the same beliefs as the previous ones but with the robot agenda updated) and branching our search space. We recursively iterate with the new task popped from the new robot agenda. Eventually, the popped task will be a primitive one $\lambda \in Op$, and its function will be applied to the currently explored agent states. If it returns *false* ($\perp$), the action is not applicable, and the exploration backtracks to another decomposition of an abstract task. However, if the action is applicable, it is added to the robot plan, and the triggers are run for each agent, updating their agenda if necessary. The human HTN is then queried to get their possible next actions from this new state. The possible actions found are added to the human plan, and, for each possible new state, we apply each agent's triggers and then continue the robot HTN exploration. This exploration continues

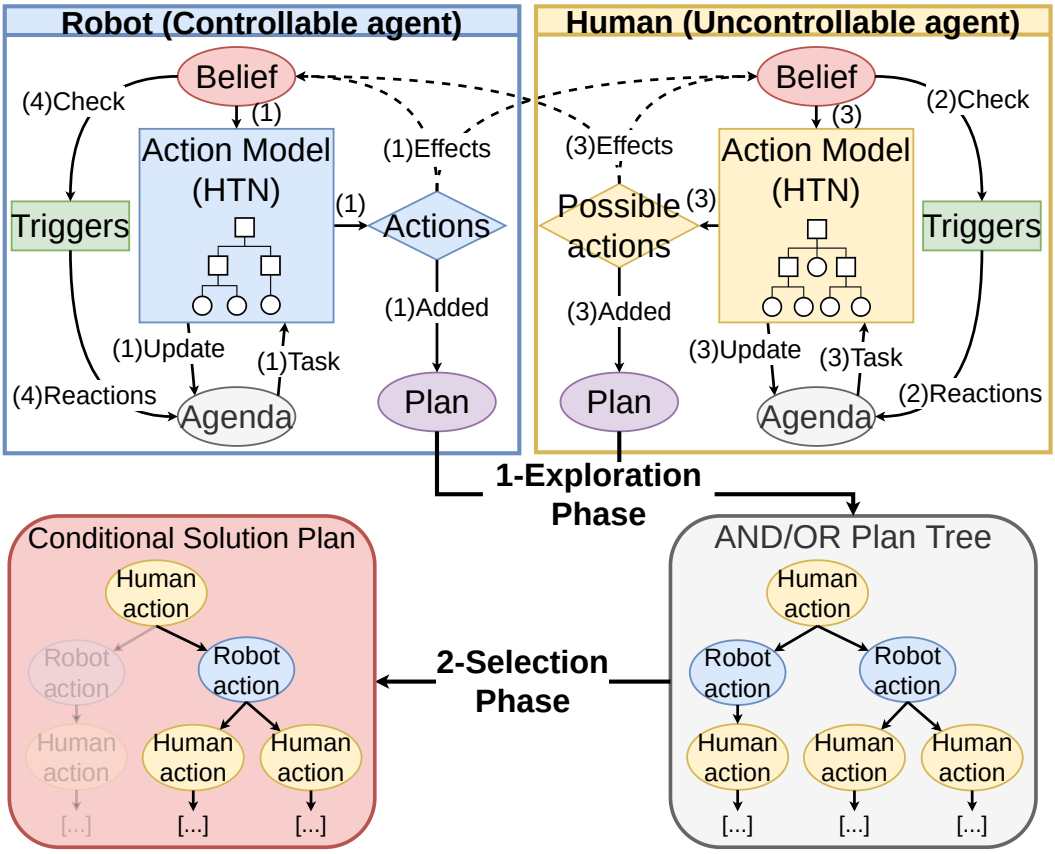


Figure 2.2: The HTNs exploration consists in iterative loops of four steps : (1) Get possible robot actions from the robot HTN, add them to the plan and apply their specific effects on the H & R beliefs, (2) Check Triggers and add the reactions in the corresponding agendas, (3) Get possible human actions based on their estimated beliefs, add them in the plan and apply their effects on the H & R beliefs, (4) Check Triggers again and add the reactions in the corresponding agendas. Here the robot is starting, but the human could with the following ordering: (3)>(4)>(1)>(2).

until the robot agenda is empty, or all the branches return *false*. The exploration process is summarized in Fig. 2.2.

The human HTN exploration differs from classical HTN planning as the goal is not to produce a complete plan but to list all the actions the human is likely to perform in a given agent state. We recursively decompose the first task of the human agenda d^H with every applicable method until we reach an applicable operator. All the operators from all the applicable decompositions are returned to the robot HTN exploration and applied.

Two special cases are handled during the exploration. When an agent's agenda is empty, the exploration returns a default passive action *IDLE*, which has no effect. It only indicates that the agent has nothing to do and will likely remain still. Besides, if no applicable action is found for an agent, the exploration returns a default passive action *WAIT*. Similarly to *IDLE*, it has no effect but represents the agent's

impossibility to act in the current situation.

Once both agendas are empty, the state is set as a success, the plan is added to the valid plans tree, and the search can continue until no decomposition is left for any task.

2.8 Plan Evaluation and Selection

In human-aware task planning, plan evaluation is a trade-off between efficiency and social criteria. The robot should be efficient but also behave in an acceptable, legible, and accommodating manner.

Cost evaluation is tricky because objective metrics are easy to use for efficiency. However, social criteria are more challenging to evaluate because they are hard to generalize. These social rules can be very context-dependent, making them hard to generalize and thus to take into account reliably.

Plan cost is a mix of all the following:

- Length of the plan (number of actions or temporal duration if available).
- Sum of individual action cost: the cost of each action can be estimated to translate the effort required to perform it. It can reflect several aspects and constraints, such as pure physical strength required, duration of the action, or energy consumption.
- Undesired states: Common sense and social norms can be used to define several rules defining undesired states. This can cover various aspects such as hygiene or safety. For instance, despite being possible and maybe efficient, we would not like a robot holding a dirty dripping mop in one hand and the sandwich we asked for in the other hand. Another example is that we would not like a robot dropping a knife just on the edge of a table or counter because it may fall and be dangerous.
- Undesired sequences of actions: For the same reasons, we can also define undesired sequences of actions. This can express preferences regarding the ordering of different subtasks, *e.g.*, since we do not want the robot to hold our sandwich while cleaning the house, we would also not like the robot to clean first and then make a sandwich because the robot is likely to be dirty while making the sandwich.

I implemented in HATP/EHDA a way to specify and take into account undesired state and action sequences. Detecting any of them in a possible plan would penalize the plan cost of the specified amount. Here, the undesired elements must be specified in the problem specification and are abstracted in the planner. However, as stated above, it is hard to generalize undesired states and action sequences and integrate them directly in the planner to avoid specifying them in the problem specification.

In HATP/EHDA, once the exhaustive exploration has been done, the result is a valid plan tree of alternating feasible robot and human actions along with their current beliefs leading to task completion. This second planning step aims to select robot actions, such as each human action in the plan has only one robot action as a child. To do so, we define a cost function $cost : S \times Op \mapsto \mathbb{R}^+$ representing the cost of an action in a specific state. The data structure is now similar to a two-player game tree. However, *MinMax* approaches are unsuitable here, as we are not in an adversarial setup but more in a collaborative one. Indeed, trying to minimize the maximum possible cost assumes that humans will always do the actions that lead to the worst plan. This defensive behavior could lead to non-optimal plans. We thus propose to explore this tree differently.

Moreover, like in HATP, we can define *social costs* functions. These functions take a complete human and robot sequence of actions (π^R and π^H) and return a cost ($\mathbb{R}^+$) which is added to the cost of the plan previously determined. By doing so, we can penalize non-acceptable sequence of robot actions (*e.g.*serving a meal just after taking out the trash) or non-satisfactory human required contribution (*e.g.*the robot requesting the human to perform small tasks multiple times instead of giving the big picture of the actual task to perform).

The approach we propose for plan selection is to minimize the average cost. It represents the human potentially selecting any course of action in their stream (while still respecting the action model defined in their HTN). The algorithm is given the root action of the plan tree previously generated. It returns the cost of the conditional plan selected while selecting the robot actions in the plan tree to minimize the average plan cost.

2.9 Qualitative Results

Each scenario is commented on below with their corresponding selected plan shown in TABLE 2.1.

The partial robot and human action models and their exploration are presented in Fig. 2.3.

(a) *H1* and R act together First, the human sets a shared goal by asking the robot to stack cubes with him. Since it is a shared goal, human and robot agendas are initialized with the “Stack" task. Thus, the robot anticipates that the human will pick the unreachable second red cube by querying the human action model. Table 2.1(a) shows the selected plan to collaboratively stack the cubes.

(b) *R* acts alone This time, the human asks the robot to stack the cubes but then leaves the scene, and the robot must act alone. Hence, the only applicable method to make the second red cube reachable is to move to the other side even though the movement action is expensive (we can imagine a table way longer than shown in the figure).

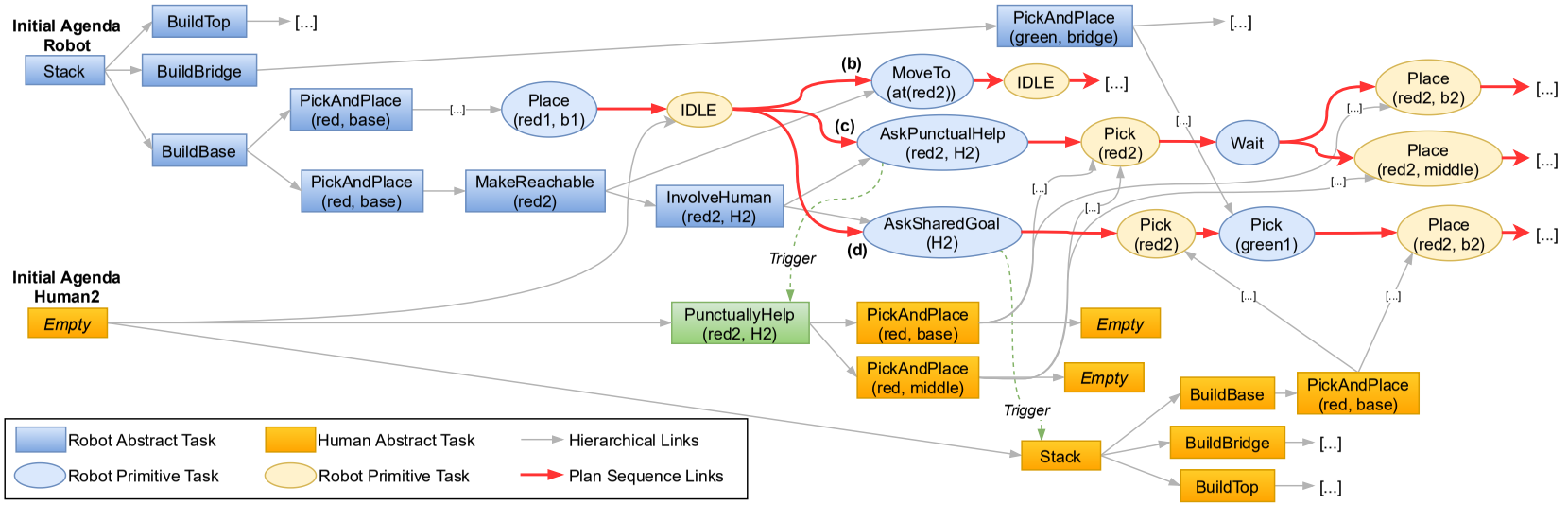


Figure 2.3: Illustration of the incremental exploration of various courses of actions corresponding to scenarios depicted in Fig. 2.1(b), (c), and (d). Since *H1* requests the robot to complete the task without establishing a shared goal, the robot agenda only contains the task to achieve, and the agenda of *H2* starts empty.

(a) *R* and *H1* build the stack together as a shared goal requested by *H1*.	**(b) *H1* requests to stack the cubes and *R* acts alone.**
R-PickAndPlace(red, base) H1-PickAndPlace(red, base) R-PickAndPlace(green, bridge) H1-PickAndPlace(blue, top) R-PickAndPlace(yellow, top)	R-PickAndPlace(red, base) R-moveTo(red) R-PickAndPlace(red, base) R-moveTo(init) R-PickAndPlace(green, bridge) R-PickAndPlace(blue, top) R-PickAndPlace(yellow, top)
(c) *H1* requests R to build the stack, *R* decides to punctually involve *H2*.	**(d) *H1* requests R to build the stack, *R* decides to invite H2 to a shared goal.**
R-PickAndPlace(red, base) H2-IDLE R-AskPunctualHelp(red) H2-PickAndPlace(red, base) R-PickAndPlace(green, bridge) H2-IDLE R-PickAndPlace(blue, top) H2-IDLE R-PickAndPlace(yellow, top)	R-PickAndPlace(red, base) H2-IDLE R-AskSharedGoal() H2-PickAndPlace(red, base) R-PickAndPlace(green, bridge) H2-PickAndPlace(blue, top) R-PickAndPlace(yellow, top)

Table 2.1: Execution trace of a selected plan for each scenario.

(c) *R* asks punctual help The first human (*H1*) requests the robot to complete the task, and another uninvolved human ($h2$) is present. The robot starts exploring its HTN and, thanks to the presence of the other human, a new method is applicable, allowing the robot to ask for help. It can ask for punctual help or a complete commitment of *H2* to the task. Of course, asking for help for one cube is less costly than building the whole stack together. However, asking for help from someone not already involved in a common task is still expensive since they must put themselves in the task's context. Nevertheless, this punctual help is less costly for the robot than moving to the other side, so this solution is selected. Note that we model that after being asked to help punctually, the human can either stack the cube themself or make it reachable to the robot by placing it in the middle. Only the first branch is shown in table 2.1(c), but the selected plan is in fact conditional with two branches as depicted in Fig. 2.3.

(d) *R* invites *H2* to share a goal Same initial setup, but now two cubes are out of reach. Asking for punctual help is still less costly than moving around the table. However, each new request to *H2* is assumed to be more and more costly, making repeated queries expensive. Therefore, due to the two unreachable cubes in this scenario, setting a shared goal becomes less costly for the robot than asking twice for punctual help.

2.10 Conclusion

What is offered by HATP/EHDA is very interesting. We rely and reason on the human model to plan the robot's actions while never compelling the human actions. The plan produced assumes turn-taking and no parallel execution of the actions. This is not a strict constraint as a post-analysis can reason on the causal links of the actions in the plan to extract a partially ordered plan to execute, making the execution more flexible. However, It is still a limitation addressed in Chapter 4.

Chapter 3

Models and Algorithms to Integrate Theory of Mind in Human-Aware Task Planning

Contents

This chapter presents my first main contribution, proposing models and algorithms to incorporate Theory of Mind concepts in HRC task-planning. An empirical evaluation is provided and discussed, demonstrating how this contribution solves a broader class of problems than HATP/EHDA, without systematically using communication.

3.1 Introduction

False belief tasks are commonly used as tests to acknowledge the presence of Theory of Mind reasoning. The most common test is the *Sally and Anne* one, depicted in

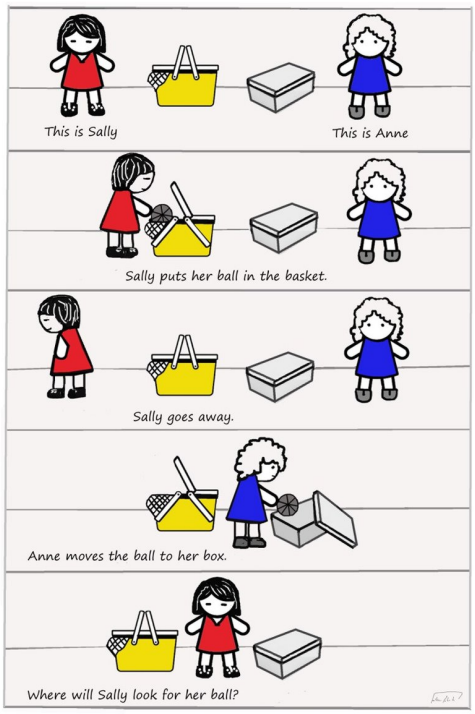


Figure 3.1: Sally and Anne Task

fig 3.1, and is used in developmental psychology to examine children's "theory of mind" understanding, which refers to their ability to understand how other people think, feel and behave. The test consists of describing the execution of a simple task involving non-observable facts and co-presence. Then, children are questioned about the beliefs of one of the characters. The task consists of the following. Sally and Anne are two co-present characters near a basket and a box. Sally puts her ball in the basket before going away. Then, in hindsight, Anne moves the ball from the basket to the box. Eventually, the following question is asked: "Where will Sally look for the ball she left?". As an observer of this scene, using Theory of Mind, we can naturally say that Sally will look for her ball in the basket because this is where she left it, and she is unaware of Anne's action. As observers, young children cannot understand that Sally has beliefs that are different from theirs, and thus, they are likely to answer that Sally will look in the box.

Theory of Mind (ToM) reasoning is critical for interaction and collaboration between agents. Thus, it is reasonable and desirable to endow robots with such skills to enhance their interactions with humans. This is the motivation of the contribution presented in this chapter. We propose an extension of HATP/EHDA to maintain human beliefs during the planning process in a principled way and to tackle estimated false human beliefs that may be detrimental to task resolution.

Some works consider ToM during execution to handle unexpected situations and contingencies like [Devin 2016]. However, such reactive approaches are not always

enough since there can be deadends requiring proactive decisions and actions to avoid.

We would want the robot to be able to reason and correctly maintain distinct human beliefs. Despite HATP/EHDA modeling distinct beliefs, this scheme does not maintain them automatically. The beliefs can only be updated in a scripted way through the description of action effects provided in the planning problem. We propose models and algorithms to integrate concepts of the Theory of Mind in the planning process of HATP/EHDA. This way, the robot can accurately estimate human beliefs and better predict their behavior. Moreover, we propose solutions for the robot when tackling estimated false human beliefs that may impact the task resolution.

3.2 Related works

This chapter's contribution is related to several topics that have not been mentioned yet. Hence, this section introduces the new topics and relevant related works to capture our contribution better.

3.2.1 Theory of Mind in HRC

Theory of Mind (ToM) refers to the ability to attribute mental states to oneself and others, such as beliefs, desires, and intentions. However, estimating the current knowledge and beliefs of the different agents is challenging. To do so, we must consider Theory of Mind concepts, especially perspective shift and the notion of co-presence.

Robots endowed with ToM abilities are more effective in proactive robotic assistance and are perceived as more socially intelligent by humans [Shvo 2022]. ToM enables robots to infer human desires, beliefs, and intentions, allowing for natural interaction between robots and humans [Yu 2023]. Robots with ToM can anticipate human strategies and incorporate them into their decision models, leading to better team performance [Romeo 2022]. The presence of ToM in robots influences human decision-making behavior and trust, making it more appropriate for human-robot collaboration [Erdogan 2022]. Computational theory of mind, based on abstractions of beliefs into higher-level concepts, facilitates collaboration on decisions and improves the quality of human decisions [Gurney 2022]. However, the lack of a unified construct and consistent benchmarking hinders progress in endowing robots with ToM capabilities.

In [Devin 2016], Theory of Mind improves shared plan execution. Humans' temporary absence or inattention during collaboration can induce insufficient knowledge about the current situation. The robot might need to detect such situations and be able to provide the information about the missing or incorrect information to its human partner without being annoying or intrusive. To do so, they propose a framework that allows the robot to estimate the other agents' mental states about the environment and the state of goals, plans, and actions. In [Lemaignan 2018],

the proposed framework named UNDERWORLDS allows cascading situation assessment to estimate other agents' beliefs during execution. The two previous frameworks can estimate and track the human partner's belief in order to reactively adapt the robot's actions during execution. However, they are not designed to be used during task planning to make proactive robot actions to avoid a potential false belief.

3.2.2 Epistemic Planning

Epistemic planning helps to plan the correct sequence of actions to reach a desired knowledge, including a desired world state. Thomas Bolander is one of this field's main contributors and describes it as follows in [Bolander 2017]. Epistemic Planning is the enrichment of planning with epistemic notions, that is, knowledge and beliefs. The human or robot might have to reason about epistemic aspects such as: Do I know at which post office the parcel is? If not, who would be relevant to ask? Maybe the parcel is a birthday present for my daughter, and I want to ensure that she does not get to know about it, and I have to plan my actions accordingly (make sure she does not see me with the parcel). The epistemic notions are usually formalized using epistemic logic. Epistemic planning can naturally be seen as combining automated planning with epistemic logic, relying on ideas, concepts, and solutions from both areas.

It enables robots to make plans to achieve the required knowledge and to reason about the knowledge and capabilities of other agents, ensuring effective collaboration and coordination in human-robot interactions [Belle 2023].

Bolander et al. proposed the Dynamic Epistemic Logic (DEL) approach [Bolander 2017] and even implemented some Theory of Mind concepts in it. However, despite being more expressive than HATP/EHDA, like the latter, this approach still does not maintain human beliefs in a principled way. They tend to use domain-specific rules given in the problem specification, usually with conditional action effects.

Muise et al. worked on multi-agent epistemic planning using a classical planning approach. Since involving nested beliefs is computationally demanding, their work proposes to convert and encode such problems into classical planning problems. Hence, state-of-the-art classical planning techniques can tackle nested beliefs of multiple agent problems.

3.2.3 Communication in HRC

Communication enables effective interaction between humans and robots, promoting inclusivity and reducing obstacles in human-robot interaction. Communication allows robots to share information about their actions and intentions, enhancing transparency and explainability [McMillan 2023]. It helps establish trust and understanding between humans and robots, leading to improved teamwork and performance [Verhagen 2022]. Adapting conversational strategies to optimize perfor-

mances and engagement is also important and addressed in [Galland 2022]. Non-verbal gestures and behavior of robots during collaboration can impact the robot's perception and influence the willingness of humans to cooperate [Arntz 2022]. Communication also allows robots to assess their own skills and limitations, propose alternatives, and adapt the execution of tasks to the capabilities of the collaborators [Ferrari 2022]. Overall, effective communication facilitates mutual knowledge, enables the exchange of information, and allows humans and robots to work together efficiently and successfully.

3.3 Maintaining the Human Beliefs

3.3.1 Enhanced Problem Specification

As an example and for illustration purposes, let us consider cooking pasta as a human-robot collaborative shared task. This scenario is depicted in figure 3.2.

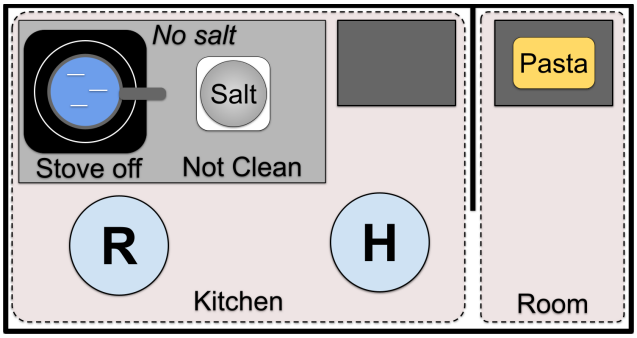


Figure 3.2: A cooking pasta collaborative shared task. It consists of turning on the *stove*, adding *salt* in the pot, *fetching* the pasta, and *pour* the pasta in the pot. Additionally, the robot has to *clean* the counter. However, there are two seperated rooms and both the presence of salt in the pot and the clean aspect of the counter are not observable facts, possibly causing false beliefs.

The robot has to turn on the stove (*stoveOn*) and clean the counter (*counterClean*), but the latter is not a part of the shared task. The human takes care of fetching the pasta while both agents can add salt into the water (*saltIn*). Before pouring the pasta into the pot, the human must know the facts, *stoveOn* and *saltIn*. Unlike *stoveOn*, the facts *saltIn* and *counterClean* are not directly observable. Hence, by acting while the human is away fetching the pasta, the robot may induce false beliefs, which may be detrimental to the shared task (e.g., the human adding salt again).

To describe this problem we start from the HATP/EHDA problem specification, which is the one described in Chapter 2. For our collaborative cooking example,

the sets B and X for the collaborative cooking example are the following:

$$
\begin{aligned}
B &= Entities \cup Places \cup Booleans \cup \{\mathsf{nil}\} \\
&\quad Entities = Agents \cup Objects \\
&\quad Agents = \{\mathsf{R}, \mathsf{H}\} \;\; \backslash\backslash \; \mathsf{R} : robot, \;\; \mathsf{H} : human \\
&\quad Objects = \{\mathsf{salt}, \mathsf{pasta}, \mathsf{counter}\} \\
&\quad Places = \{\mathsf{kitchen}, \mathsf{room}\} \\
&\quad Booleans = \{\mathsf{true}, \mathsf{false}\}
\end{aligned}
$$

$$
\begin{aligned}
X &= \{at(e) \mid e \in Entities, \; saltIn, \; stoveOn, \; counterClean\} \\
&\quad Range(saltIn \mid stoveOn \mid counterClean) = Booleans \\
&\quad Range(at(\mathsf{R} \mid \mathsf{H} \mid \mathsf{pasta})) = Places \\
&\quad Range(at(\mathsf{salt} \mid \mathsf{counter})) = \{\mathsf{kitchen}\}
\end{aligned}
$$

Our first contribution starts from here, where we augmented the specification by associating each state-variable $x_i \in X$ to a location and an observability type. Thus, we define two other functions besides the *variable value assignment* function (Definition 3) to assign observability types (Definition 11) and locations (Definition 12) to state-variables.

Definition 11 ***(Variable observability asssignment function** obs**.)*** *A* variable observability assignment function *over X is a function obs that maps each $x_k \in X$ into an observability type $t_j \in \{\texttt{OBS}, \texttt{INF}\}$, s.t., $obs = \{(x_1, t_1), \ldots, (x_n, t_n)\}$. With $obs(x_k) = \texttt{OBS}|\texttt{INF}$, x_k is said to be respectively observable | inferable. We use the following notation to access the observability type t_j of the state-variable x_k in the state s_i: $obs_i(x_k) = t_j$.*

Definition 12 ***(Variable location assignment function** loc**.)*** *A* variable location assignment function *over X is a function loc that maps each $x_k \in X$ into a $l_j \in Places \cup \{\texttt{nil}\}$, s.t., $loc = \{(x_1, l_1), ..., (x_n, l_n)\}$. $Places \subseteq B$ captures a group of constant symbols such that each member is a predefined area in the environment. Agents are always either "situated" in a place or moving between two places. We consider x_i to be located in every place $\in Places$ if $loc(x_i) = \texttt{nil}$. We use the following notation to access the location l_j of a state-variable x_k in the state s_i: $loc_i(x_k) = l_j$.*

The observability and location of the state-variable are internal to the robot but assumed to be common to both agents. Consequently, we update our *state* definition (def. 7) to take them into account in the new definition 13.

Definition 13 ***(State** s**.)*** *In this contribution, in addition to both agent states, a* state *now comprises the two novel assignment functions obs and loc, s.t., $s_i = \{\sigma_i^R, \sigma_i^H, obs_i, loc_i\}$.*

This new definition still consider the two agent states s.t. $\sigma_i^\varphi = \{val_i^\varphi, d_i^\varphi, \pi_i^\varphi\}$. Now, in each state, we keep track of each state-variable's observability type and location, and we can reason on them to update the human beliefs (val^H) accordingly.

We remind definition 8 stating that a state $s_i \in S$ contains *false beliefs*, or *belief divergences*, if $\exists x_j \in X, val_i^H(x_j) \neq val_i^R(x_j)$. That is, if any state-variable has a different value in the human beliefs w.r.t. the robot one, we assume the human has a false belief about those particular state-variables. Since the planner is part of the robot, we assume the robot beliefs are true and correspond to the ground truth. Consequently, any divergence between the two beliefs is assumed to be a human false belief, as it would make no sense to keep false information in the robot's beliefs.

The initial state of our cooking example can be described as follows. There is no initial belief divergence, the initial partial plans are empty, and both agents perform the shared cooking task named *CookPasta*. In addition, the robot must clean the kitchen counter once the cooking is done. Hence, the task *CleanCounter* is added to the initial robot agenda after the shared task. The only *non-obversable* facts are the presence of salt in the pot and if the counter is clean. All entities, including the counter, the stove, and the salt, are in the kitchen. Only the pasta is in the adjacent room. More precisely, the initial state s_0 can be written like this:

$$
\begin{aligned}
s_0 &= \{\sigma_0^R, \sigma_0^H,\ obs_0,\ loc_0\} \\
&\quad \pi_0^R = \pi_0^H = () \\
&\quad d_0^R = (CookPasta, CleanCounter) \\
&\quad d_0^H = (CookPasta) \\
&\quad val_0^R = val_0^H = \{at(\mathsf{R}) = at(\mathsf{H}) = \mathsf{kitchen},\ at(\mathsf{pasta}) = \mathsf{room}, \\
&\qquad\qquad saltIn = \mathsf{false},\ stoveOn = \mathsf{false},\ counterClean = \mathsf{false}\} \\
&\quad obs_0 = \{(saltIn, \texttt{INF}),\ (counterClean, \texttt{INF}), \\
&\qquad\qquad (stoveOn, \texttt{OBS}),\ (at(e), \texttt{OBS}) \mid e \in Entities)\} \\
&\quad loc_0 = \{(saltIn, \mathsf{kitchen}),\ (counterClean, \mathsf{kitchen}),\ (stoveOn, \mathsf{kitchen}),\} \\
&\qquad\qquad (at(e), val^0(at(e))) \mid e \in Entities\}
\end{aligned}
$$

When refining the human agenda in s_0 we obtain the following *refinement* comprising two possible actions. The human can either begin by *adding* salt to the pot, or they can move to the other room to *fetch* the pasta.

$$ref(d_0^H, val_0^H) = \{a_1, d_1, a_2, d2\} = \{(add\_salt(), d1), (move\_to(\mathsf{kitchen}), d2)\}$$

3.3.2 State Transitions and Beliefs Updates

We now describe how the agents' beliefs are updated when executing a planned action and, thus, how the transition occurs from one state to another. The fundamental principle is that the human agent learns from observing an action execution or their environment. First, we will give the three assumptions we made regarding this approach.

Assumption 1: We do not consider uncertainties. Thus, agents are either wrong or right about the state of the world but never uncertain. This would be an interesting future work.

Assumption 2: We do not consider cases where the robot's beliefs can diverge. The actual ground truth is unknown since the planner is part of the robot. We can only assume that the robot's estimation of the state of the world is correct and then reason using this estimation.

Assumption 3: Coming from the two previous assumptions, we assume that humans only make deterministic moves when not observed. Hence, regardless of being co-present, the robot's beliefs are always updated with the effects of the action.

Thus, Assumption 3 indicates that $\forall x \in X$ we always have,

$$val_{i+1}(x) = \begin{cases} w, & \text{if } x \leftarrow w \in \mathit{eff}(a) \\ val_i(x), & \text{otherwise} \end{cases} \tag{3.1}$$

The place associated with a state-variable can be modified by the action's effect, e.g., when an agent moves to another room while holding an object. However, this must be specified in the action's effects. In this work, we assume that the observability type of each fact is constant during the task. Adapting the approach to allow dynamic observability types is feasible, which will be discussed later. So, $\forall x \in X$,

$$obs_{i+1}(x) = obs_i(x) \tag{3.2}$$

$$loc_{i+1}(x) = \begin{cases} l, & \text{if } x \leftarrow l \in \mathit{eff}(a) \\ loc_i(x), & \text{otherwise} \end{cases} \tag{3.3}$$

The new agenda of each agent (d^R_{i+1}, d^H_{i+1}) are created by the HTN refinement algorithm, and thus, they are directly retrieved from the obtained refinement. This refinement decomposes abstract tasks in the agenda until the first task is a primitive action. Every applicable method is applied, leading to a set of possible actions (and refined task networks).

The new estimated human beliefs val^H_{i+1} is the two-step result of our Situation Assessment processes that models the human's real-time sensing and reasoning capabilities about their surroundings.

First, let us define the notions of *co-presence* and *co-location*, which will be key to maintaining the evolution of agents' beliefs as planning progresses.

Definition 14 ***(Co-presence & Co-location.)*** *In a state $s_i \in S$, two agents, φ_1 and φ_2, are considered to be co-present if $val_i(at(\varphi_1)) = val_i(at(\varphi_2))$. This relation is noted $\varphi_1 \curlywedge_i \varphi_2$ in the rest of the paper. Similarly, we say that an agent φ_1 is co-located with a state-variable $x \in X$ if $val_i(at(\varphi_1)) = loc_i(x)$, noted $\varphi_1 \curlywedge_i x$.*

Now, we can define two Situation Assessment (SA) processes that will maintain the estimated human beliefs.

Definition 15 ***(Inference Process.)*** *An agent observes the execution of an action by being either co-present with the acting agent or by being the acting agent. If so, the agent infers the new values of every state-variable in the action's effects.*

Based on the above definition, the human beliefs are updated a first time as follows when action a is executed in state s_i,

$$val'^{H}_{i+1}(x) = \begin{cases} w, & \text{if } x \leftarrow w \in \mathit{eff}(a) \text{ and} \\ & (H = \mathit{agt}(a) \text{ or } H \curlywedge_i \mathit{agt}(a) \\ & \text{ or } H \curlywedge_{i+1} \mathit{agt}(a)) \\ val^{H}_{i}(x), & \text{otherwise} \end{cases} \tag{3.4}$$

To change its *place* in the environment, agents would use a dedicated *"move"* action, such that its effect only updates the agent's location.

Definition 16 ***(Observation Process.)*** *An agent observes its surroundings and assesses the exact value of each state-variable located in the same place (i.e., each state-variable the agent is co-located with).*

After applying the effects of an action with the equations 3.1 to obtain val_{i+1}, and running the inference process (def. 15) with equation 3.4 to obtain the partial human beliefs val'^{H}_{i+1}, the observation process (def. 16) is executed. It updates again the estimated human beliefs with the facts currently observable by the human and provides the human beliefs to store in the state s_{i+1}. We have, $\forall x \in X$:

$$val^{H}_{i+1}(x) = \begin{cases} val_{i+1}(x), & \text{if } H \curlywedge_{i+1} x \text{ and} \\ & \mathit{obs}_{i+1}(x) = \texttt{OBS} \\ val'^{H}_{i+1}(x), & \text{otherwise} \end{cases} \tag{3.5}$$

Before starting the planning process, the observation process is executed once on the initial state s_0. This allows us to potentially correct the estimated human beliefs with the facts the human should initially be able to observe.

The definition of the set *Places*, i.e., how the environment is divided into different *places*, is guided by the shape of our state transition function. Hence, a $\mathit{place} \in \mathit{Places}$ is an (symbolic) area in the environment such that, when situated in it, agents are aware of each other's activity, and they can assess every observable fact located in it. Agents cannot be aware of others' activity if not co-present with them, and they cannot assess observable facts if not co-located with the facts.

Note that unlike in DEL [Bolander 2021], our knowledge representation is simple and prevents us from expressing agents being *uncertain* about a fact. In line with the classical closed-world assumptions, agents either know the truth or have a false belief w.r.t. the ground truth. We consider a straightforward scenario in which the human is *"unaware"* of non-observed changes in the environment. This results in estimated false human beliefs, helping to detect whether a non-observed robot action can disrupt a seamless collaboration.

3.4 Relevant False Human Beliefs

In this section, we explain our procedure to detect *when* a false human belief should be corrected and *how*.

3.4.1 Detection

The human and the robot carry individual distinct beliefs, while the two can be aligned or diverging when the human has a false belief. To produce a legal solution plan the robot is fine with such false human beliefs unless they are qualified as *relevant* (Definition 17). In such cases, the relevant false belief needs to be tackled.

Definition 17 *A **relevant false belief** is a false belief that influences the next action(s) the human is likely to perform in terms of number, name, parameters, or effects. This can be written as follows: A state s_i contains a relevant false belief if either (3.6) or (3.7) is true:*

$$ref(tn_i^H, val_i^H) \neq ref(tn_i^H, val_i^R) \tag{3.6}$$

$$\{\gamma(s_i, a) \mid \forall a \in ref(tn_i^H, val_i^H)\} \neq \{\gamma(s_i, a) \mid \forall a \in ref(tn_i^H, val_i^R)\} \tag{3.7}$$

We consider that as soon as a false belief affects human actions, it should be tackled. An interesting future work could be to check in a principled way the overall positive and detrimental impacts of this false belief on collaboration. However, it is out of the scope of this work.

3.4.2 Resolution with Minimal Communication

A state containing a false human belief marked as *relevant* must be handled. The first way to do it is by planning communication actions such that the robot communicates only the required facts to the human. This allows to correct relevant human beliefs, but false beliefs that are *"non-relevant"* will remain.

3.4.2.1 Modeling Communication Actions

In this context, we propose a generic communication action schema (ca). An agent φ_i can *communicate* an assertion $x = z$ (with $x \in X$ and $z \in \text{Range}(x)$) *via* the action $ca_{\varphi_i,\varphi_j}(x, z)$ if $val^{\varphi_i}(x) = z$ and $val^{\varphi_j}(x) \neq z$. The effect of $ca_{\varphi_i,\varphi_j}(x, z)$ corresponds to $val^{\varphi_j}(x) \leftarrow z$. Such actions are considered equally costly and instantaneous.

3.4.2.2 Communicate Only the Required Facts

Definition 17 indicates if there is at least one diverging state-variable in the human beliefs causing adverse effects, but without identifying which one(s). Hence, we explain a subroutine below with the three steps, describing how we identify the pertinent state-variables to align and how the corresponding communication actions are created and inserted into the robot's plan.

1. *Store* each state-variable whose value differs in the human beliefs from the robot beliefs: $X_{diff} = \{x \mid x \in X, val_i^H(x) \neq val_i^R(x)\}$.
2. *Build*, for each stored state-variable $x \in X_{diff}$, a communication action $ca_{R,H}(x, val_i^R(x))$, all stored in a set CA_{diff}.
3. *(Breadth-First Search.)* The *source* is s_i. Applying each $ca \in CA_{diff}$ generates a new state by aligning *exactly* one state-variable in the human beliefs s.t. $s'_i = \gamma(s_i, ca)$. The search continues until the first state s'_i selected to expand does not contain a relevant false belief. The communication actions used from the root until this selected state are *retrieved* in a set CA.

Once the above subroutine finishes, the retrieved communication actions in the set $CA = \{ca_{R,H}(x_1, val_i^R(x_1)), ..., ca_{R,H}(x_j, val_i^R(x_j))\}$ must be inserted in the plan for belief alignment. Thus, our definition of the conditional solution (def. 10) is redefined to be sound w.r.t. our approach. An edge can now either be a human action a^h or a robot action a^r with

a set of communication action CA. At each step, humans perform *Observation*, while the robot executes each communication action $ca \in CA$, making the human's belief to *update instantaneously*.

The set CA is inserted before the diverging human actions and after the closest state where agents are co-present. However, it could be interesting to investigate using a better plan evaluation system to find the best place to insert this set.

3.4.3 Resolution by Delaying Non-Observed Robot Actions

So far, we relied on communication. However, communication can be cognitively demanding depending on the environment (e.g., noisy). Thus, when the relevant false belief is due to a non-observed robot action, we propose also considering implicit communication by postponing the pertinent robot action until the human is estimated to observe its execution. This prevents false beliefs from even occurring.

First, a branch using communication is explored, and the state-variables concerned by the relevant false beliefs are retrieved (through all $ca \in CA$). Then, we check if a non-observed action produces the divergence. For now, it is done by checking if the relevant divergence concerns only one inferable state-variable and if it was not present in the initial state. After, we identify which action creates the divergence by regressing the current branch/trace sequentially. Hence, we can identify when the relevant divergence appears and which action should be delayed. Once identified, we create another branch in the plan just before the identified action. In this new branch, DELAY actions are inserted in the robot's plan until the human is co-present. When the human is co-present again, the identified action is inserted and observed by the human. Then, the nominal planning process is resumed.

3.5 Result

Referring to the related work section, we are not aware of an implemented planning system that can be used as a baseline. Hence, we use the HATP/EHDA solver to help present our approach's results on three *novel* planning domains.

3.5.0.1 Cooking Pasta Domain

The running example corresponds to a specific problem in this domain. In fact, agents and pasta can initially either be in the kitchen or in the adjacent room, the stove might be on or off, and there might be salt or not in the water. The results will focus on the following three state-variables from X. Both *stoveOn* (`OBS`) and *saltIn* (`INF`) are relevant to the human, unlike *Clean* (`INF`) which only concerns the robot.

3.5.0.2 Preparing Box Domain

A box with a sticker on it and filled with a fixed number of balls is considered prepared and needs to be sent. Both agents can *fill* the box with balls from a bucket, while only the robot can *paste* a sticker, and only the human can *send* the box. The bucket can run out of balls, so when one ball is left, the human *moves* to another room to *grab* more balls and *refill* it. The number of balls in the box is *inferable*, while all other variables are *observable*. In the following, three boxes have been considered.

Table 3.1: Success and communication ratio of different approaches.

Domain	**HATP/EHDA**		**Only Comm**	**With Delay**
	S	*S I.Div.B.*	*Comm*	*Comm*
Cooking	18.6%	6.9%	69.5%	65.2%
Box	25.0%	14.3%	79.7%	75.0%
Car	12.5%	0.0%	68.8%	64.1%
Average	18.7%	7.1%	72.6%	68.1%

3.5.0.3 Car Maintenance Domain

The washer fluid (`OBS`) and engine oil (`INF`) levels have to be *full* before *storing* the oil gallon in the cabinet (`INF`). Only the robot can *refill* both the tanks and store the gallon while situated at *Front* of the car. *Front-left* and *Front-right* headlights have to be *checked* and a light-bulb has to be *replaced* at *Rear*. Only humans can check and replace lights, and they can start with either of these two tasks. Both agents start at *Front*. The car's hood needs to be *closed* by the human at last.

3.5.1 Qualitative Analysis

Considering the cooking domain, we discuss in detail the plans obtained with our approach to a problem corresponding to the description given in the introduction. I.e., there is no initial human false belief, agents both start in the kitchen, the pasta is in the adjacent room, the stove is off, and there is no salt in the water. The resulting plans are shown in Fig. 3.3, and their detailed presentation explains how the approach works in practice. Since the human is uncontrollable and has different possible actions, the plan branches and the robot's actions differ in each case.

In (*left*), the human first adds salt, and then the robot turns on the stove. In both cases, thanks to the inference process, we estimate that the human will be aware of both facts about the salt (*acting*) and the stove (*co-present*). Then, while the human is away to fetch the pasta, the robot cleans the counter. Since the human is not co-present, their beliefs are not updated and now contain a false belief about the counter state. Once back, since *counterClean* is not *observable*, the observation process does nothing, and the false belief remains. However, this false belief does not affect human actions (non-relevant). Hence, there is no need to align human beliefs.

In (*middle* and *right*), the human begins by leaving the kitchen to fetch the pasta. First, let's focus on the (*middle*) trace. The robot turns on the stove and adds salt while the human is away, creating two false beliefs. When returning to the kitchen, the observation process updates the human beliefs with the observable facts located in the kitchen. This fixes the false belief about *stoveOn*. The robot then cleans the counter, which the human observes. However, without communication, the human's next action will be either "add salt" or "ask the robot", but considering the ground truth, the human could directly pour the pasta. Hence, the false belief on *saltIn* is relevant and has to be corrected. Therefore, a communication is inserted in the robot's plan, and a "delay" branch is created (*right*). In this delaying branch, the robot delays the add salt action until the human is co-present in order to make it observed (inference process) by the agent. In addition to this implicit communication, like in (*middle*), the human assesses that the stove is on and can directly pour the pasta.

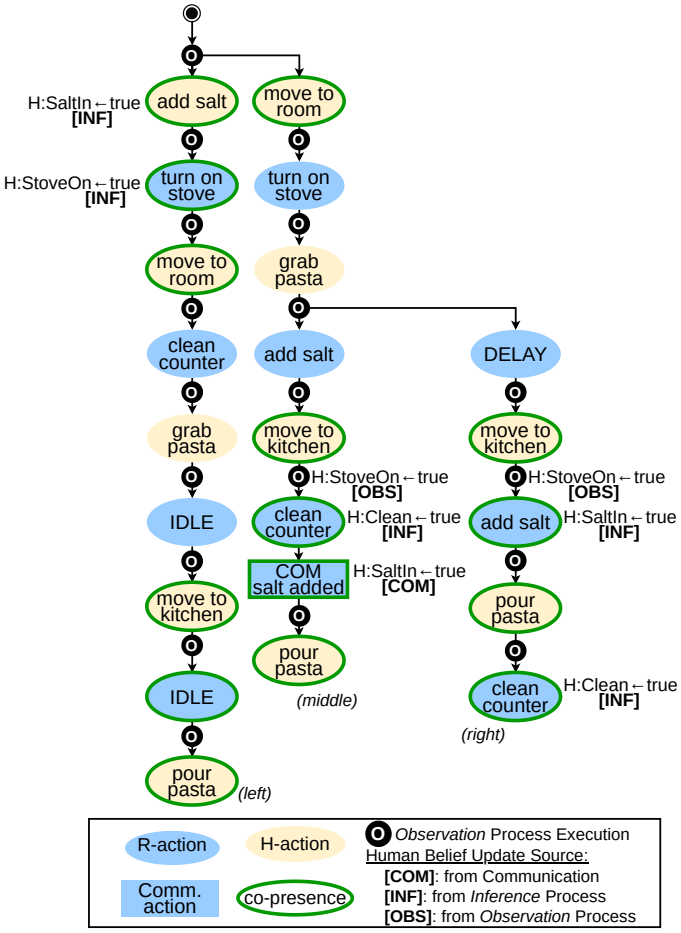


Figure 3.3: Plan obtained for the cooking scenario. 3 branches. Left: The human starts by adding salt. The only false belief is about *"counterClean"*, which is irrelevant to the human agent. Hence, no communication is added. Middle: While the human is away, the robot turns on the stove and adds salt, creating two false beliefs. Once back, we estimate that the human agent will be able to assess the observable fact *"stoveOn"* but not *"saltIn"*. Since the human agent might add salt again due to this false belief, it is relevant and fixed with a communication action. Right: The relevant false belief about *"saltIn"* is avoided by delaying the robot's action until the human is co-present.

3.5.2 Experimental Results and Analysis

In each domain, the actions and tasks remain the same. So here, a problem is defined by a starting agent (R or H) and a pair of initial beliefs (val_0^R, val_0^H). Initial ground truth $(val_0 \Leftrightarrow val_0^R)$ is defined by setting each state-variable to an initial value. Five selected state-variables can have two possible values instead of one. Among these selected ones, three can diverge in human beliefs. These combinations generate 256 pairs of initial beliefs, where 12.5% of them include initially aligned beliefs. Then, considering the starting agent, we obtain 512 problems for each domain. Each of the 1536 generated problems has been solved by HATP/EHDA, by *our approach* using first *only communication* and then using

also *delay*. The obtained quantitative results appear in Table 3.1.

The overall success rate (S) and the one for initially diverging beliefs ($SI.Div.B.$) are shown for the HATP/EHDA solver. As expected, this solver always finds legal plans when dealing with initially aligned beliefs, and the low value of $SI.Div.B.$ reflects how poorly it handles belief divergences without specifically designed action models. Our approach always finds legal plans, so we omitted its success rates in the last two columns, and we can say that it solves a broader class of problems.

Furthermore, considering the initially diverging beliefs and the divergences created along the planning process, more than 87.5% of all problems involve belief divergences. However, only 72.6% of the generated plans include communication actions when using only verbal communication. This means that *our approach* communicates only when necessary and not systematically. The amount of communication is even reduced to 68.1% when delaying actions. In the latter case, only delayed branches that do not imply the human to wait are kept.

3.6 Discussion and Limitations

This contribution has a few limitations to discuss.

First, the observability types of the state-variables are assumed to be constant. However, dynamic observability types permit modeling scenarios like an agent placing an object in an opaque box. The object would no longer be observable despite being in the same place as the agent. Since *Places* can be symbolic, with clever domain modeling, one can model objects' disappearance when placed in a drawer. Dynamic observability types would simplify such modeling and make it even more expressive. However, this extension requires further research to correctly redefine Equation 3.2.

The underlying scheme allows just a single agent to execute a *"real"* action at a time. However, a post-process can allow the execution of actions concurrently [Crosby 2014], however, note that the domain modeler has modeled $\mathcal{P}_{rh}$ as a sequential joint task. Parallelism is not considered in the current modeling and planning process, which limits the potential for concurrent executions. This challenge is addressed in the contribution described in Chapter 4.

We believe our modeling-level SA proposals could fit in any other planning approach framing multi-party systems having one controllable agent while can only hypothesize remaining agents' behaviors (e.g., human-centered AI).

Agents' SA models cannot simply refute a false belief, they can only assess new true facts to correct them. For instance, assume the human *wrongly* believes that the pasta is in kitchen. The SA does not help refute this when the agent is in kitchen because the state-variable *NotAt(Pasta)* in kitchen is not modeled in our domain. However, such issues do not affect the completeness and, if necessary, our approach *tackles* such cases as relevant false beliefs.

The relevance of estimated false human beliefs is challenging to determine. Currently, as soon as a divergence influences the next human actions, we assume that the divergence is likely detrimental to the goal and, thus, is relevant to be fixed. However, it would be interesting to find a method to evaluate the consequences of a divergence on the goal and plans in a principled way.

We have planned a user study for the future to conform our framework with reality and validate the approach.

We discussed earlier that DEL knowledge representation is more expressive, flexible, and can handle uncertainty. However, it requires an augmented action schema to accurately

maintain each agent's beliefs. Think of a specification for *"move"* action manually listing all the environmental facts to be observed by an agent for managing their beliefs. In our case, it is implicitly maintained within a state.

We can consider running a set of rules (e.g., *graph-based ontology*) to bring new interesting facts in the state based on a set of known facts. We believe that this aspect opens up new possibilities in the future for integrating human-aware collaborative planning and ontology.

3.7 Conclusion

This contribution is an extension of the HATP/EHDA human-aware task planner. The planner plans and implicitly coordinates the robot's actions with all estimated possible human (uncontrollable) behaviors that are then emulated to generate a new state. Our extension and contribution are, first, to integrate a *Situation Assessment* based reasoning system in the planner. This allows for maintaining distinct agents' beliefs based on what they can/should observe. Compared to existing epistemic planners, this simplifies the action descriptions by focusing on their effects on the world and not how they influence each agent's beliefs. In addition, we propose to detect false human beliefs and tackle only the necessary ones in a principled way. First, we propose minimal and proactive explicit communication. Second, when pertinent, we propose an implicit communication by postponing the non-observed robot action until the human is co-present to observe it.

The relevance of false belief, when to optimally communicate, and parallelization are interesting future works.

Chapter 4

Modeling and Planning for Concurrent and Compliant Joint Action Execution

Contents

This chapter presents my second main contribution, proposing a new human-aware task planning approach based on a step-based model of compliant and concurrent joint action. The approach's description is supported by empirical results proving its effectiveness in terms of the latitude of choice given to the human and the satisfaction of their internal preferences. We further validated this by developing an interactive simulator used for a user study, described in the following chapters.

4.1 Introduction

From HRI paper

In the context of HRC for a shared task [Selvaggio 2021], we believe, based on the literature on joint action [Sebanz 2006a, Sebanz 2009, Clodic 2017, Gordon 2023], that the key towards a seamless interaction is, to consider the human as an uncontrollable agent and to be fully and concurrently compliant with them. The human should not be dictated which action they must perform, as in [Roncone 2017, Buisan 2022], and the robot must comply with possible human decisions and actions during execution.

To collaborate with such humans with their (hidden) preferences, one can devise an online planning scheme coupled with a plan executor. However, in order to maintain real-time performance, online planning generally keeps a restricted horizon. Therefore, decisions taken online may lead to a dead end or may not lead to an optimal solution. Offline planning overcomes these issues.

We propose a new *offline* task planner which extends an existing human-aware planning system addressed in [Buisan 2022]. The new planner is designed to take into account a *Model of Execution*, which is in the form of an automaton and mainly inspired by the joint actions schemes. The model captures humans' latitude in their decisions. The planner's output is the robot's behavioral policy, which describes the robot's action in a state such that the action is congruent and compliant with the human's decision in this state and their (estimated) preferences and that it is also legal to be executed in parallel. Our framework also allows humans to share their (new) preferences at any time during execution while the robot's policy is adapted online to that. In addition, our approach considers social signals to enhance execution by minimizing uncertainties. Both humans and robots issue signals to clarify situations, such as performing an action, waiting for the other agent's necessary actions, or indicating a desire to remain passive.

In this chapter, we discuss relevant related work before describing the joint action model of execution that is central to our approach. We then describe the task planning problem and then introduce our novel framework. The following two sections explain how the robot policy is generated by a three-step process: *exploration*, *characterization*, and *generation*. We empirically evaluated our approach in simulation. With a BlocksWorld scenario, we show how our approach can effectively produce a concurrent robot behavior that is compliant with human online decisions and preferences.

4.2 Related works

There have been a few attempts to cater to concurrent execution, but they deal with explicit time to manage concurrency [Cirillo 2009b, Köckemann 2014]. In [Cirillo 2009a], the robot does not plan actions for humans but forecasts their actions/plans from their activities and bases its own decision on the distribution of possible human plans. Here, robots can perform actions concurrently, carefully estimating/managing the completion time of the agents' actions. We can see the human activity recognition part as a form of our Identification (ID) process of the automaton used. The robot needs such a plan/goal recognition technique to be compliant with the human's decisions. But, unlike ours, they do not consider an explicit shared goal among the agents. Hence, humans are not concerned with stuff robots might be interested in during collaboration, e.g., giving signals to be passive. We believe that a shared goal creates a different context in HRC than the robot just being compliant with an estimated human's goals/plans. Moreover, we claim that dealing with concurrent actions is inevitable in planning, even if actions are instantaneous,

to effectively deal with multiple agent systems [Crosby 2014, Shekhar 2020], especially if a human operator is involved, like in our case. We extend HATP/EHDA [Buisan 2022] to demonstrate that.

In another work, both *recognition* and *adaptation* take place simultaneously and comprehensively [Levine 2014]. It deals with the action scheduling of an already generated contingent plan comprising human and robot actions. It outputs schedules for the robot actions that can execute concurrently, but to do that, explicit temporal constraints are considered.

Ramachandruni, Kent, and Chernova (2023) [Ramachandruni 2023] propose a communication-free human-robot collaborative approach for an adaptive execution of multi-step tasks. In their approach, the robot observes and supports human decisions, actively selecting actions to optimize collaborative task efficiency. Unlike our approach, they introduce an extended collaborative HTN representation with role assignment for planning and state tracking during execution, which is more in line with [Roncone 2017]. In contrast, we employ two distinct HTNs for robot and human capabilities and use an AND/OR tree for exploration and execution tracking. While their online planning may enhance scalability, optimality is not guaranteed. Also, our scheme accommodates both verbal and non-verbal communication, allowing the human to express preferences that update the robot policy online.

4.3 Joint Action Model for Planning

4.3.1 Rationale and Example

Our task planning approach uses a model of execution to improve the fluency and amenability of HRC. This model is in the form of an execution controller and is based on several key notions and mechanisms borrowed from studies on joint actions [Michael 2016, Kourtis 2014], and adapted to Human-Robot Joint Action [Clodic 2017, Curioni 2019]. The key idea is that co-acting agents co-represent the shared task context and integrate task components of their co-actors into their own task representation [Schmitz 2017, Yamaguchi 2019]. Also, coordination and role distribution rely strongly on reciprocal information flow, e.g., social signals [Curioni 2019], prediction of other's next action [McEllin 2018].

Our proposed execution model is implemented on a robot that co-acts with a human, integrating explicit representation and exploration of the task representations for the robot and for the human. It also identifies precisely how reciprocal information flow is used in task execution (detecting and interpreting human actions, signals produced by the robot while acting, and also when the robot waits for human actions or their signals).

Another essential question is the criteria for choosing the next action, or more globally, how to share the load between the two co-actors. The choice depends on the context and actors' preferences [Gombolay 2015, Strachan 2020, Curioni 2022]. Concerning the case when one actor is a robot, we think it is important to provide a standard default behavior of the robot where the robot does its best to reduce human load but still leaves full latitude to act whenever humans want. Our scheme provides this ability and also allows humans to inform about their preferences at any moment.

To better understand the problem we are trying to solve, let's consider an example where concurrent actions can be conflicting. This example consisting of a simple pick and place task is depicted in fig 4.1. The human and the robot have to pick cubes, *c1* and *c2*, that both can reach. The cubes can be picked up simultaneously unless the agents try to

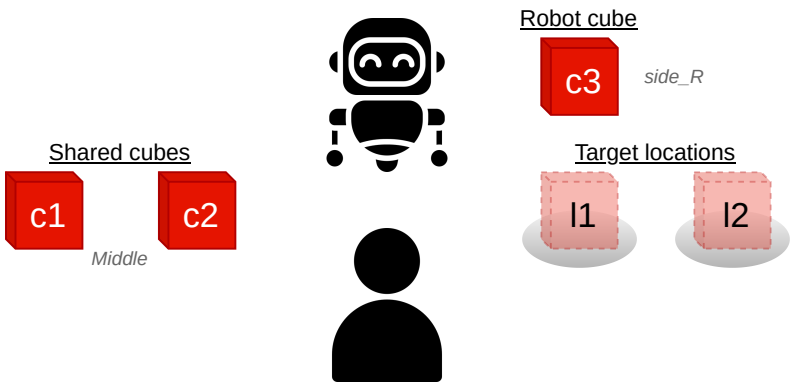


Figure 4.1: Example of conflict for a concurrent joint action. Shared cubes, *c1* and *c2*, can be picked up by both agents, and only the robot can pick *c3*. Agents can simultaneously pick up both shared cubes but must coordinate their actions. Otherwise, they might conflictingly try to pick the same cube. Similarly, another coordination is required to avoid placement conflicts between locations *l1* and *l2*. However, notice that the robot can pick *c3* without any risk of conflicts with the human action.

pick the same cube, which causes conflicts between their actions. As a result, despite being executable in parallel, the actions are interdependent. Thus, the agents must coordinate their actions for smooth execution. A similar coordination must happen when placing the cubes to avoid conflicts between *l1* and *l2*. However, a third cube *c3* is present and can only be picked up by the robot. The robot can pick *c3* without any risk of conflicts with the human action.

This example illustrates the need for coordination even for simple tasks. It also shows that considering possible conflicts can also be a criterion for optimizing. Thus, the best robot action might be to pick up *c3* instead of one of the shared cubes to avoid potential conflicts, even if *c3* is more distant and more costly to pick than *c1* and *c2*. This example also highlights the relevance of exploring several possible executions of concurrent actions while planning to anticipate and evaluate such situations and produce a robust, compliant, and efficient robot policy. This is why, based on joint action literature, we formulated a model of concurrent joint action execution described below that will guide our planning search.

4.3.2 Abstracted Joint Action Model for Planning

The model of execution we formulated, depicted in fig 4.2, is inspired by joint action literature and describes the agent's coordination during execution. It comprises synchronization and perception processes based on social signals exchanged between the agents, such as starting an action (arm motion), hand gestures, or verbal communication. This section presents a simplified and abstract version of this model that suffices to guide our planning framework. A complete version that explicitly models the human behavior and exchanged signals is provided in the next chapter and used to supervise the execution of the produced robot policy.

The model describes the possible transitions from a given state to another, step by step. The robot always informs the human about the beginning of a step and then waits for their

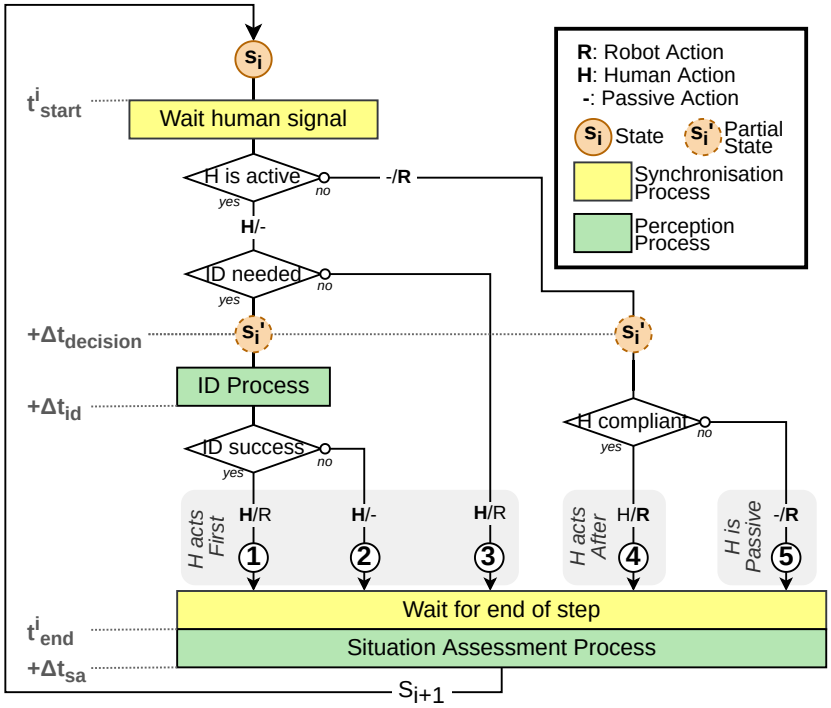


Figure 4.2: Abstracted Model of Execution in the form of an automaton run by the robot. It captures the latitude of uncontrollable humans in their actions and guides our task-planning approach. In this paradigm, the two agents can act concurrently, but one always complies with the other's decision. Here, humans are always free to decide whether to start acting first, after the robot, or not to act at all. To be compliant, the robot attempts to identify human decisions using perception and situation assessment as well as possible collaborative human signaling acts (e.g., gestures or speech).

decision by looking at social cues. The top yellow rectangle in the figure represents this waiting process. This human decision can either be to start acting or to be passive. Hence, the robot always lets humans first decide which action they want to perform, including the choice to be passive. This decision is detected with perception by tracking social cues such as human motions and hand gestures.

The diamond shapes below the first synchronization process are conditions, driving the automaton into different branches depending on specific conditions. First, we differentiate between cases where the human is passive and cases where the human is active ("H is active"). When the human is performing an action, the 'yes' branch, we first check if the human action must be identified or not. So far, only the fact that the human is acting is known, but not yet which particular action they are performing. To avoid potential conflicts, we consider that there is no need to identify the human's action if the best robot action does not depend on the human decision. For instance, consider in the cube picking example presented in figure 4.1 that the cube *c3* is easier to pick and place for the robot than the two other ones. Thus, regardless of which cube the human picks, the robot should pick *c3*. Its best action does not depend on the human decision, and there is no need to identify the human's action. However, if *c3* is effectively harder to place than *c1* and *c2*, the robot must first identify which cube the human is grabbing to pick the other one. As a result, the model differentiates the cases where the human action must be identified ("ID needed") or not. If not, then the robot can directly start acting (branch 3). Otherwise, the Identification (ID) process is executed and may either be successful ("ID success") or not. If not, to avoid any potential conflict, we decided that the robot should remain passive (branch 2). If the human action has been successfully identified, the robot can perform the best corresponding action concurrently (branch 1).

When the human decides to be passive, the robot should start acting. However, while the robot is acting, we consider that the human is free to either remain passive until the next step (branch 5) or to "change their mind" and start acting (branch 4). This is represented by the diamond shape "H compliant". When the human decides to start acting after the robot starts, the human can only perform actions that do not conflict with the already started robot action. Note that this case can be seen as a way for the human to let the robot decide and then comply with the robot's decision. Still, the human decided to let the robot decide, thus, the human is given as much latitude of choice as possible.

Overall, at every step of the execution, the human is free to decide: 1) to start performing any feasible action, the robot will comply with this decision; 2) to let the robot decide and act first to purposely be compliant with the robot; 3) to be passive and let the robot act alone.

When both agents finish their actions, the step is considered as *"over"*. The bottom yellow rectangle labeled "Wait for end of step" represents this synchronization. Then, a Situation Assessment Process is executed to assess the new world state (s_{i+1}), which is the result of the concurrent actions being executed in the state s_i. Once the next state is identified, the automaton repeats until the task is solved and a goal state is reached.

Note that if, for any reason, both agents are passive during a step, the state is unchanged, so the step is repeated.

4.4 Problem and Solution Specifications

Compared with the original HATP/EHDA specifications, this contribution slightly modifies the problem and solution specifications. This section describes the main differences.

Problem Specification

A few assumptions are used in this contribution, simplifying the problem specification described in Chatper 2, especially the *agent state* (σ^{φ}) that can be omitted only to keep *states* (s_i), also referred to as *planning states* (ps_i).

First, belief divergences are out of the scope of this particular work. Hence, for simplicity reasons, we consider the two beliefs (robot and estimated human ones) to be constantly aligned, and they are represented as a unique world state ($val_i = val_i^R = val_i^H$). Therefore, unlike Chapter 3, the two agents' beliefs are always aligned and initialized with a same initial world state using state variables. However, we are convinced that this work could be adapted easily to consider the two distinct beliefs.

Additionally, the agents' partial plans have been removed for performance reasons. This is due to our new solution format in this contribution, switching from an AND/OR tree to a Directed Acyclic Graph (DAG). The domain modeler must now define relevant state variables to track specific events instead of checking the previously accessible planned action sequence. This improved the performance a lot.

As a reminder, a *state* fundamentally differs from *beliefs*. A *state* (s_i) corresponds to a specific state in which the planning problem is while progressing toward a solution. However, *beliefs* (val_i^{φ}) corresponds to the state of the world in the agent φ's perspective. Hence, *beliefs* (here aligned) are part of the *state*.

The problem specification of the conflict example shown previously in figure 4.1 is given below for illustration purposes. First, the initial state s_0 is described by:

$$
\begin{aligned}
s_0 &= \{val_0, d_0^R, d_0^H\} \\
&\quad d_0^R = d_0^H = \{MoveCubes\} \\
&\quad val_0 = \{at(\mathsf{c1}) = at(\mathsf{c2}) = \mathsf{middle},\ at(\mathsf{c3}) = \mathsf{side\_R} \\
&\qquad\quad free(\mathsf{l1}) = free(\mathsf{l2}) = \mathsf{True}, \\
&\qquad\quad holding(\mathsf{R}) = holding(\mathsf{H}) = \mathsf{nil}\}
\end{aligned}
$$

The agents' action models are still represented with HTN. For this example, the corresponding models are depicted in figure 4.3.

Since the robot and human HTNs are very similar, they are represented as a single one for legibility purposes. The only difference between these two distinct HTNs is that the robot can pick the cube near itself $c3$. Hence, unlike the human, the robot action model includes a possible decomposition (shown in blue in the figure) picking up the $c3$ cube. Note that these action models use a recursive representation of the task. The abstract task $MoveCubes$ decomposes into several subtasks, including itself, to continue moving cubes until a given condition is satisfied. The two $MoveCubes$ methods capture this condition. The tasks decompose into picks and places if cubes are missing in the target locations. Otherwise, the task refines into nothing ($\emptyset$). Notice the $PlaceTable$ abstract subtask, which takes care of cases where an agent cannot place its cube in any target location and must place it back on the table.

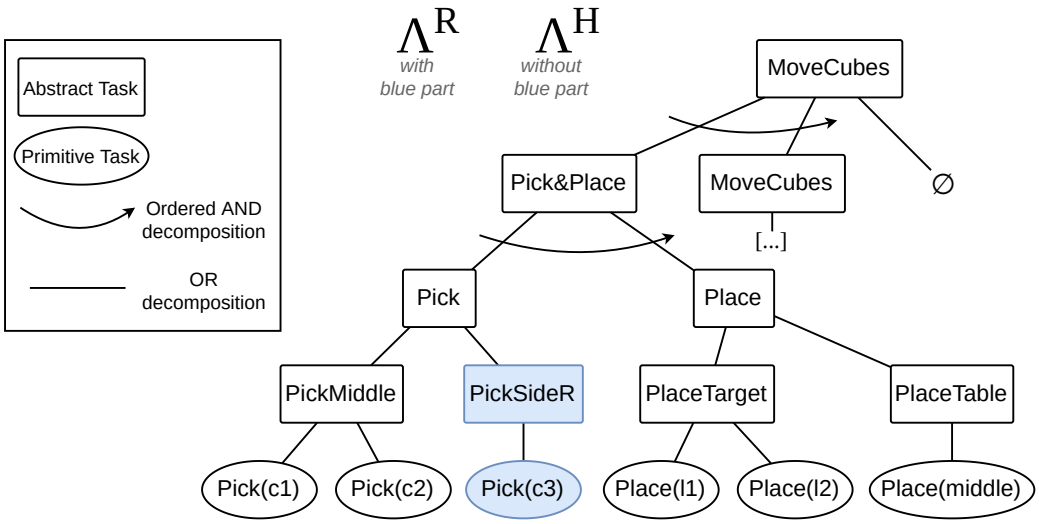


Figure 4.3: Robot and Human action models (Λ^R and Λ^H) for the conflict cube picking example. Since the two HTNs are very similar, only one is represented. Unlike the robot, the human HTN does not include the blue part.

Solution Description

Compared to Chapter 2, the solution format changed from an AND/OR tree to a Directed Acyclic Graph (DAG). This modification significantly improved the planning approach's performance because many branches of the original solution tree are redundant. The exploration now produces a Directed Acyclic Graph (DAG), referred to as the search graph, from the initial p-state to several goal p-states through sequences of concurrent human-robot action pairs. Such a graph is depicted in figure 4.4.

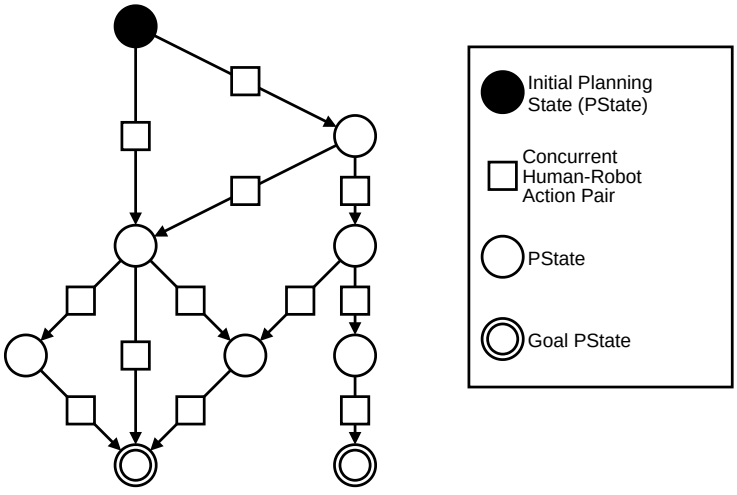


Figure 4.4: Directed acyclic solution graph

In the DAG, any path from the root to a leaf is a possible plan corresponding to a sequence of concurrent human and robot actions. Note that we consider passive actions. Hence, only one active agent may exist in a concurrent pair of actions. Moreover, a leaf is a

node without children and is considered as a goal p-state if both agendas (d^R and d^H) are empty and if val_j satisfies given goal conditions. In our example, the goal conditions are:

$$goal\ conditions: free(\mathsf{l1}) = free(\mathsf{l2}) = \mathsf{False},$$
$$holding(\mathsf{R}) = holding(\mathsf{H}) = \mathsf{nil}$$

Once the exploration is done, i.e. when the complete search graph is computed, another process extracts the optimal robot policy from the graph. In the manner of an AND-OR tree, this policy indicates for each p-state the best concurrent robot action (OR node) to execute to be compliant with any possible human action (AND node). However, this extraction is based on an estimation of human preferences. These preferences specify what metrics the human partner would prefer to maximize or minimize and in which priority. The overall planning process is depicted in figure 4.5 and both the exploration and the policy generation are detailed in the following sections..

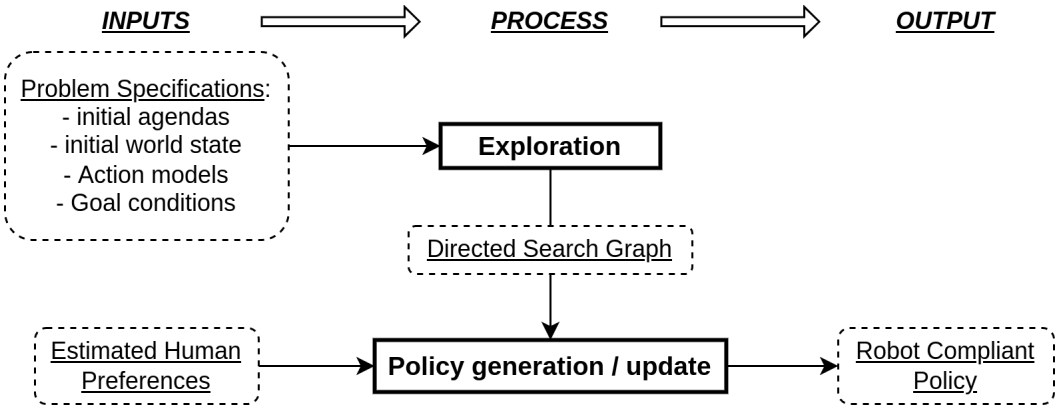


Figure 4.5: Overall planning process

Before providing more details about the planning approach, we show in figure 4.6 a partial DAG obtained for our conflicting cube example. On this partial DAG are shown several sequences of concurrent human-robot actions going from the initial state to a goal state where the task is achieved. The complete DAG is too big to be legible. We can see in the figure that passive actions *PASS* are included during the search. We also see how different ordering and concurrent execution can quickly generate numerous paths in the graph. The original AND/OR tree representation quickly becomes overwhelming and resource-consuming to manipulate.

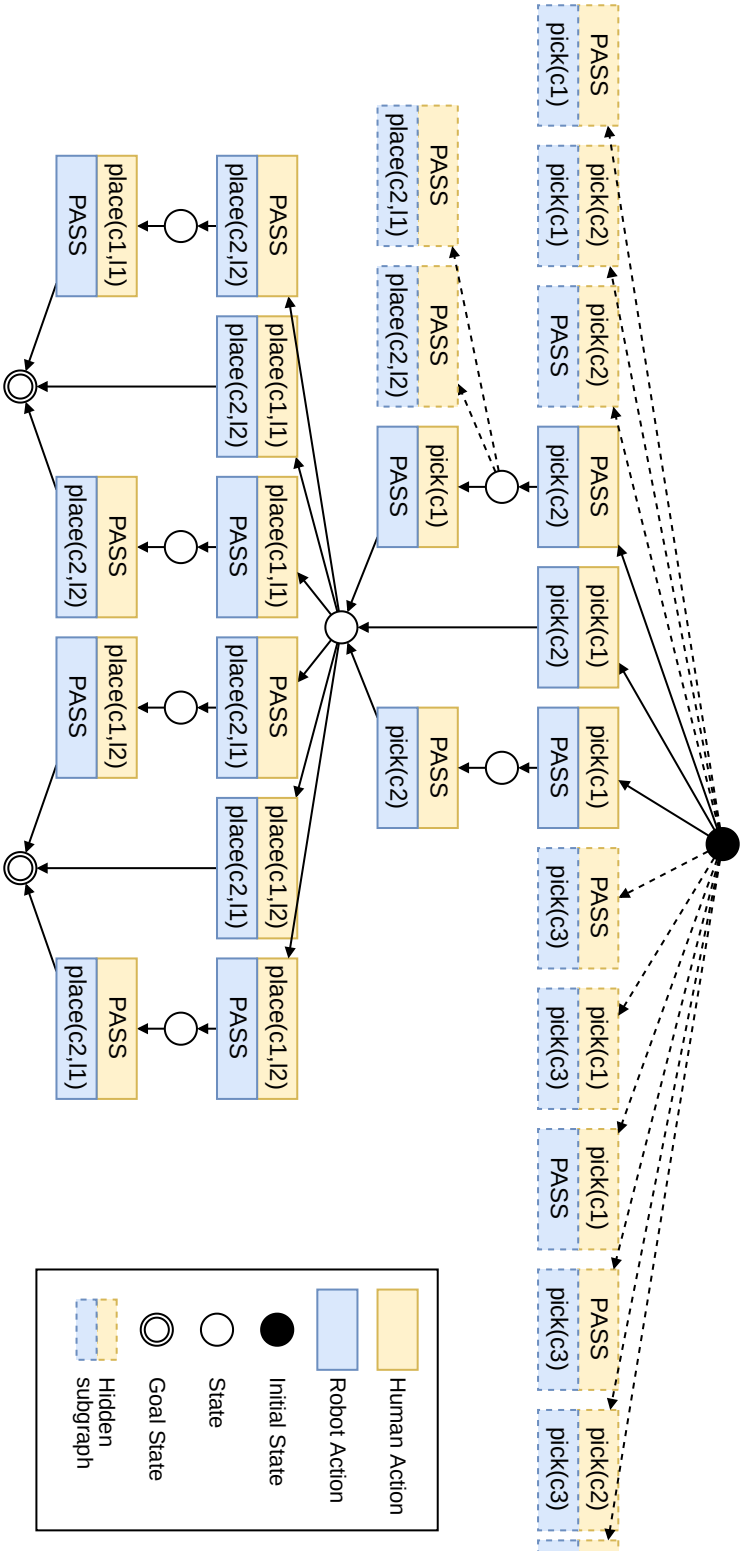


Figure 4.6: Partial Directed Acyclic Graph (DAG) for the conflicting pick example.

4.5 Exploration and Search Phase

This section details how the exploration happens and, thus, how the search graph is generated. This requires several sub-processes, each of which is detailed here. First, the overall exploration process is presented, and the following subsections provide details on the sub-processes mentioned in the overall process.

4.5.1 Overall Exploration Process

We keep track of the p-state to explore, and this set is being initialized with the initial p-state. Then, until the set is empty, we select one and explore it. First, from this selected p-state, every possible concurrent human-robot action pair is computed considering both agendas, the world state, and reasoning on the compatibility of the actions in terms of preconditions and effects. This process requires several sub-steps and is detailed later. Thus, we obtain several action pairs leading to the same amount of new p-states (with updated world states and agendas). Second, we check if a newly created p-state is similar to any existing p-state. If so, we can "merge" them to avoid redundant computations. To do so, we keep track of the unique p-states that have already been checked, and for each new p-state, we check if it is similar to one of the already checked ones. If not, the new p-state is added to the set of already checked p-states. If a similar one is found, the new p-state is deleted, and the action pair leading to it is connected to the existing p-state instead. Eventually, the remaining new p-states are added to the set of p-states to explore. The exploration is over when the set of p-states to explore is empty. The obtained DAG corresponds to the "search graph" where any path from the initial p-state to a leaf corresponds to a possible execution trace.

4.5.2 Compute Next Agent Actions

From a p-state (i.e. from a world state and an agenda), we can estimate the next actions an agent is likely to perform. Doing so is referred to as the refinement process. We use the corresponding action model in the form of a Hierarchical Task Network (HTN). Here, the agendas are considered as lists of abstract or primitive tasks, where primitive tasks can be executed (actions), and abstract ones must be decomposed into several other subtasks. The refinement process consists of applying applicable methods of the first task of the agenda until it is primitive. When several methods are applicable, they are all applied in a distinct refinement trace. Eventually, for each different sequence of applied methods, we obtain a new agenda starting with a primitive task. For each such primitive task, we create a copy of the world state contained in the given p-state and apply the corresponding action, updating the world state. Finally, a new p-state is created for each possible action with the refined agent agenda, the unchanged other's agenda, and the updated world state. Note that this process can "generate" default passive actions in several cases. First, when an agent's agenda is empty, an *IDLE* passive action is inserted as the first primitive task in the agenda. This means that the agent has nothing to do and, thus, is likely to remain passive. When there are no applicable methods or the primitive task is not applicable, then a *WAIT* passive action is inserted. This means that the agent still has something to do but cannot do it. Note that when computing the concurrent pairs of action, we also add the *PASS* action, which corresponds to the agent being voluntary passive despite having something to do. Note also that these different passive action types help to understand the generated plans better, but they are treated similarly in the planning process.

4.5.3 Concurrent Action Pairs Computation

This sub-process computes from a given p-state all possible concurrent action pairs that may be executed. It is based on the previously described refinement process. The main objective of this sub-process is to identify the next actions the agents are likely to perform to reach the goal and identify which of them can be executed in parallel. The classical way to do such reasoning is by analyzing the preconditions and effects of the two actions and determining if there are conflicts between them, e.g., the effects of the first action make the preconditions of the other false. However, our current Python implementation of the planner is convenient but has no "explicit" preconditions and effects. Everything is defined through Python functions. The effects of an action are a function with a world state as input, which returns the updated world state. Action preconditions are functions with a world state as input and return a boolean. Methods are functions that use the world state as input and return a list of tasks with which to update the agenda. Methods also have preconditions working similarly to action preconditions. Hence, extracting the explicit effects and preconditions of an action is challenging. That is why we decided to rely on an assumption to check the compatibility of concurrent actions. Consider two actions A and B that can be sequentially performed in both orders, i.e., $A \rightarrow B$ and $B \rightarrow A$. Then, we assume that there are no causal links between the two actions and that they can be performed concurrently. The only care to take is about *shared resources* such as tools that would only be used during the action, making them available before and after but not during the action. To tackle this issue, shared resources are explicitly declared in the world state and in the action models. Two actions requiring the same shared resource cannot be parallelized. Such an approach to parallelize two actions comes with a benefit. Indeed, it only needs an action precondition boolean function. As a result, action estimations can be a black box with only basic low-level action precondition descriptions. This way, we can easily replace the way we estimate the next actions each agent is likely to perform. Especially for humans, we could use a pre-trained human activity estimator using a neural network or a more classical planner like PDDL.

With the causal principle above in mind, we proceed as follows to compute the possible concurrent action pairs from a given p-state. We start by estimating all possible human actions by refining the human agenda, which generates a new p-state for each possible action. For each such p-state, we first create an action pair where the robot is passive by inserting a *PASS* robot action. This *PASS* pair is stored among all other 'human pass pairs". Second, we refine the robot agenda to obtain all feasible sequences of human then robot actions and their associated p-states. We refer to them as the sequential human starting pairs. Symmetrically, we compute the sequential robot starting pairs and "robot pass pairs" by starting with the robot and then refining the human agenda.

Eventually, every action pair present in the human and robot starting pairs is extracted and added to a set of concurrent action pairs. Additionally, here, passive actions are always parallelizable with other regular actions. The only case where this could not be the case is when considering "joint actions" requiring the two agents to lift an object together. For now, such actions are not considered. Thus, the two sets of *PASS* pairs are directly added to the set of concurrent action pairs. Lastly, a double passive pair with two PASS is generated and added to the concurrent set. This pair is special since it does not update the world or the agendas. Hence, it leads back to the previous p-state without progressing toward the goal. These pairs do not need to be explored and are helpful in different ways. First, it helps the execution, for instance, when only the human can act but decides to pass voluntarily. Then, the policy will natively stay in the same p-state. Second, it is easy to detect dead-ends because they correspond to double *WAIT* pairs. In such cases, both agents cannot

act and remain stuck without solving the task. Last, double *IDLE* pairs indicate that both agendas are empty and, thus, that the task is solved.

The obtained set of concurrent action pairs corresponds to all possible concurrent actions that the human and the robot can execute in parallel in the initially given p-state. Each possible pair leads to a new p-state with updated agendas and world state, creating a tree structure.

4.5.4 Merging p-states

Although the tree structure produced by the process described above is complete and sound, it is inefficient and scales poorly. Indeed, during the exploration, we are likely to encounter similar p-states several times. For instance, consider an action pair where both the human and robot are active, leading to a new p-state. Consider an action pair where only the human is active and a second where only the robot is active. Performing those two last pairs in both orders creates two other branches. However, even though the trace is different, the three p-states are identical (same world state and agendas), and it is highly redundant to explore each independently. That is why, after each computation of the concurrent action pairs, we check if any newly generated p-states are similar to an existing one. If so, we connect the corresponding pair to the existing p-state to avoid redundant explorations. Doing so, we transform the tree structure present in the previous chapters into a Directed Acyclic Graph (DAG) (on the condition that we do not consider double *PASS* cycles). In the manner of a tree, we will refer to nodes without children as leaves nodes, which are goal p-states. Hence, now, each leaf can be reached with several paths. When looking for similar existing p-states, we assume that p-states that are parent with the new p-state, directly or not, will necessarily be different. Thus, they are excluded from the potentially similar p-states set. This speeds up the comparison process.

Note also that for the p-states to be similar despite different concurrent executions like explained above, we had to remove the *partial plan* present in the specification described in Chapter 2. To significantly improve the performance and use the DAG, we can no longer reason on the *partial plan* of the agents. However, we can still define a dedicated state variable to keep track of specific events or action sequences if it influences the task decompositions. For instance, it would be relevant to define a state variable to track how many times the robot requested punctual human help, like in the example of Chapter 2.

4.6 The Robot Policy

First, as depicted in figure 4.7, all child concurrent action pairs of a p-state (PS/ps_i) can actually be seen as an AND-OR graph. Since we want to preserve the latitude of choice the human has at execution, each possible human choice of action is considered as an AND edge and leads to a partial p-state (PS'/ps_i^j). From a partial p-state, each compliant concurrent robot action is considered an OR edge and leads to another p-state.

Notations: A p-state is referred to as ps, and ps_i is the i-th p-state. A partial p-state is referred to as ps', and ps_i^j is the j-th partial p-state of the i-th p-state.

Thus, generating the robot policy Π consists of identifying the best concurrent compliant robot action RA^* for each possible human action or partial p-state ps_i^j.

These best concurrent robot actions are determined by aiming to optimally satisfy an estimation of the human preferences regarding the task. Eventually, at execution, the human is free to perform any of the explored actions. The robot will accordingly perform an optimal concurrent action to solve the task and satisfy their estimated preferences.

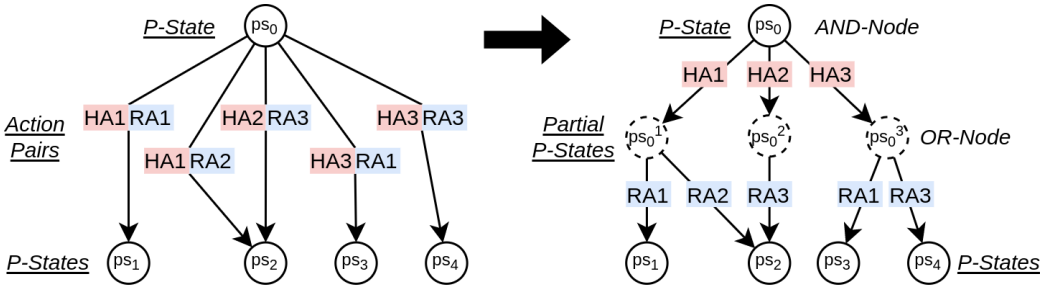


Figure 4.7: AND-OR graph representation of the action pairs. The robot policy must be compliant with any human decision. Hence, the problem can be seen as an AND-OR graph where, for each possible human choice of action (AND node), we must determine the best concurrent robot action among the possible robot actions (OR node).

Hence, before giving more details about the policy itself, we first describe the format of human preferences, how we could estimate them, and what it allows us to do. After, we describe the actual process to generate the robot policy from the search graph using the estimated human preferences

4.6.1 Using Estimated Human Preferences for Plan Evaluation

In this approach, instead of trying to minimize action and social costs, which are challenging to estimate accurately and quantitatively, we aimed to satisfy an estimation of human preferences. In a way, the costs are reflected in human preferences. Our approach is to characterize each possible trace with a set of various metrics. Objective and general metrics can be complemented with additional domain-specific metrics (marked with *), which must be given in the problem specification. The metrics used are the following:

- **Time of Task Completion**: Time step at which the task is achieved.
- **Time of End of Human Duty**: Time step after which the human can remain passive.
- **Human Effort**: Number of non-passive human action.
- **Global Effort**: Number of non-passive human and robot action.
- ***Passive While Holding**: Number of steps where an agent is passive while holding a cube.
- ***Number of Drops**: Number of times an agent drops a cube (place back a cube on the table, not in the stack).

The set of metrics helps to characterize and evaluate each possible trace. However, even if the possible plans are characterized, we so far have no way to compare them and find the best one. Doing so requires additional criteria indicating how to prioritize and compare the different metrics.

The preferences can be given verbally or estimated by any other means. Here, we consider human preferences in the following form. These preferences are an ordered list of the metrics characterizing the traces. Note that all metrics in the list above do not need to be present. This ordered list indicates if each metric should be maximized or

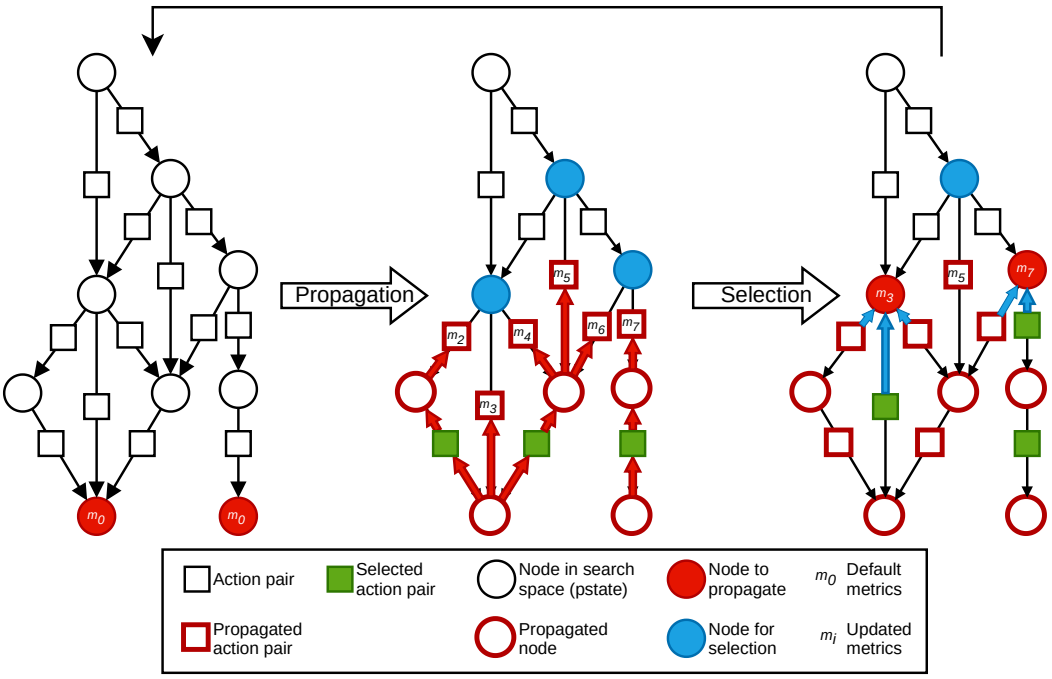


Figure 4.8: Policy generation process illustration on an arbitrary search graph. Propagation and Merge processes repeat until there are no more p-state (node) to propagate or for selection.

minimized and with which priority. For instance, assuming the preferences aim to minimize all metrics, the plan with the lowest first metric of the list will be considered as better. If the plans have an equal first metric, then we use the second metric, and so on. Here are two arbitrary examples of human preferences trying respectively to finish the task as fast as possible ($preferences\_1$) and to minimize the human effort ($preferences\_2$) (all metrics are minimized):

$$preferences\_1 : TTC > GE > HE > TEH > PWH > ND$$

$$preferences\_2 : HE > TEH > TTC > GE > PWH > ND$$

Note that estimating either human preferences or explicit action and social costs is challenging and is hardly accurate. Those are very context-dependent and can even vary over time. Being aware of this, we use the estimated human preferences as a guide for the robot behavior. However, we also make sure to comply with human activity to lessen the impact of erroneous estimations.

4.6.2 Generation

To generate the robot policy Π from the search graph, we proceed from the leaves to the root. Overall, we progressively compute the set of metrics for each possible trace. When reaching a p-state with several children, we compare the metrics of the different traces leading to this node. We can then identify the best trace leading to each partial p-state and the best trace leading to the p-state. Each best trace to a partial p-state is used to

update the robot policy, and the overall best trace is used to continue the propagation with the best reachable set of metrics. This process is achieved by repeating two sub-routines detailed just below, namely *Propagation* and *Selection*.

4.6.2.1 Format and Initialization

During this process we compute and store the best reachable set of metrics alternatively in the p-states and in the action pairs. Initially, we store default null metrics in every leaf p-state. Then, we keep track of two types of nodes. First, keep track of the nodes whose metrics should be propagated in the next step, stored in the set $\{psToPropagate\}$. This set is initialized with all leaf p-state nodes. The Selection sub-routine later populates this set. Secondly, we keep track of the nodes where the best action to perform among several must be identified. The propagated sets of metrics are used to identify the best action to put in the robot's policy. This set is $\{psForSelection\}$ and is populated by the propagation sub-routine while being emptied by the selection sub-routine. The process is over when the two sets are empty, thus, when there are no more nodes to propagate or for selection.

4.6.2.2 Propagation

The propagation sub-routine is depicted in algorithm 1 and described here. It consists of picking a node to propagate from the set $\{psToPropagate\}$. For each parent action pair of that node, we create a copy of the set of metrics of the propagated node and update the metrics according to the various rules described below. The metrics must be cumulative.

The rules to update the standard metrics are the following (all metrics start at 0):

- If the pair is not *IDLE-IDLE*, then *Time of Task Completion* is incremented by 1.
- If the pair is not *IDLE-IDLE*, if the human action is passive, and if the current *Human Effort* is zero, then the temporary metric *Number Last Passive Human Action* is incremented by 1.
- *Time of End of Human Duty = Time of Task Completion - Number Last Passive Human Action*.
- If human action is not passive, then *Human Effort* and *Global Effort* are incremented.
- If robot is not passive, then *Global Effort* is incremented.

Rules to update domain-specific metrics must be provided:

- If human action is passive while holding a cube, then *Passive While Holding* is incremented.
- *(Similarly with the robot)*
- If the human action is to drop a cube back on the table, then *Number of Drops* is incremented.
- *(Similarly with the robot)*

The updated metrics are stored in their corresponding action pair. Then, two cases can occur for each parent action pair with stored metrics, referred to as propagated pairs. First, if the parent node of the propagated pair has more than one child, then we add this parent node to the set $\{psForSelection\}$. Otherwise, the metrics of the action pair are stored in the parent node, which is also added in the set $\{psToPropagate\}$. The sub-routine repeats until the set $\{psToPropagate\}$ is empty.

4.6.2.3 Selection

When a node has several children, i.e. several possible action pairs, then we must evaluate and compare them in order to make the best robot choices and update the policy with them.

Algorithm 1 Propagation Sub-Routine

```
1: while psToPropagate ≠ ∅ do
2:     N ∈ psToPropagate
3:     psToPropagate ← psToPropagate \ {N}
4:     for each P in N.parents do
5:         P.metrics ← CopyAndUpdateMetrics(N.metrics, N, P)
6:         if HasOneChild(P.parent) then
7:             P.parent.metrics ← P.metrics
8:             psToPropagate ← psToPropagate ∪ {P.parent}
9:         else
10:            psForSelection ← psForSelection ∪ {P.parent}
11:        end if
12:    end for
13: end while
```

The evaluation is part is done by the propagation sub-routine. The Selection sub-routine checks when robot choices are ready to be made, then it updates the robot policy, and eventually prepares the next propagation phase.

The selection sub-routine is depicted in algorithm 2 and described here. It checks every node in the $\{psForSelection\}$ set to know if we are ready to make a choice, i.e., if every child pair of that node has metrics stored in it and has been propagated. If not, nothing happens, and the node remains in the set. If so, we are ready to compare the pairs and update the policy. Since we want the human to be free to perform any action, even suboptimal, we must find the best concurrent robot action for each possible human action. The first step is to organize the child pairs by similar human action. Then, for each group of pairs, we compare their metrics using the estimated human preferences and identify the best pair of the group, which is marked a "best compliant pair". After this, all marked pairs are compared, and the overall best pair is identified and marked as "best pair". Eventually, the metrics of the "best pair" are stored in the node from $\{psForSelection\}$, and the node is removed from the set and added to the other set $\{psToPropagate\}$.

4.6.2.4 Additional policy updates

After executing the main process to generate the robot policy Π, each partial p-state ps' is mapped to a robot action to execute. So, the partial p-state must be identified at runtime to execute the robot policy. The human action choice characterizes a partial p-state. Therefore, identifying partial p-state consists of running the ID process mentioned in the Model of Execution, which briefly tries to identify which action the human is starting to execute during a step. However, such identification can be challenging, so we try to avoid it when possible.

Indeed, for any p-state ps_i from the search graph, if $\forall j, \Pi(ps_i^j) = RA$ with RA being one unique/same robot action, then the best robot action does not depend on the human choice. Thus, the ID process can be avoided in such a case, and the policy is complemented as follows $\Pi(ps_i) \leftarrow RA$.

Moreover, we consider cases where the ID process failed to identify the human action and is noted as λ. To prevent potential conflicts due to identification failure, the policy is updated s.t. $\Pi(\lambda) = PASS$. Note that this work only considers identification failures and

Algorithm 2 Selection Sub-Routine

```
1: for each PS in psForSelection do
2:     if ReadyForSelection(PS) then
3:         bestPairs ← ∅
4:         for each partialPS in GetPartialPStates(PS) do
5:             pairs ← GetCorrespondingPairs(partialPS)
6:             P ← IdentifyBestPair(pairs) ▷ Compare metrics and identify best
7:             Π(partialPS) ← P.robotAction
8:             bestPairs ← bestPairs ∪ {P}
9:         end for
10:
11:        bestPair ← IdentifyBestPair(bestPairs)
12:
13:        PS.metrics ← bestPair.metrics
14:        psForSelection ← psForSelection \ {PS}
15:        psToPropagate ← psToPropagate ∪ {PS}
16:    end if
17: end for
```

not wrong identifications.

4.6.3 Execution

The execution of the policy stems from the Model of Execution automaton and is depicted in Algorithm 3.

4.7 Empirical Results

We provide results obtained after simulating symbolically the execution of robot policies produced with our approach, thus without durative actions.

4.7.1 Simulated Scenario

The execution is symbolically simulated by running an implemented version of the automaton described in the *model of execution*. This implementation is close to the presented algorithm 3 where action execution is mocked and replaced by symbolic and instantaneous actions. The ID process is assumed to be always successful Thus, the current state progresses in the search graph based on the human decision and the produced robot policy before eventually reaching a leaf node, indicating that the goal is satisfied. We then retrieve which course of action has been executed and analyze it. The human behavior is simulated and described just after.

We evaluated our approach in the BlocksWorld domain. Figure 4.9 shows one problem instance. The human and the robot are on two sides of a big table, and their shared task is to stack colored cubes, as shown in the given goal pattern. Initially, all colored cubes are arranged on the table that is divided into three zones: Each agent has a dedicated zone (RZ & HZ) and a common zone (CZ) is in the middle and accessible for both. Each agent can only pick cubes from either their own zone or from CZ. There is a box in RZ in which

Algorithm 3 Execution of the Robot Policy

```
1:  ps ← ps_0                                              ▷ Initial state
2:  while ps.children ≠ ∅ do
3:      IndicateStepStarted()                              ▷ Inform the human
4:      WaitHumanDecision()
5:      if ps ∈ Domain(Π) then                             ▷ If ID not needed
6:          Execute(Π(ps))
7:      else
8:          if HumanIsPassive() then                       ▷ Detected by WaitHumanDecision
9:              Execute(Π(ps'))
10:         else
11:             idPartialPS ← IDProcess()                  ▷ ∈ {λ} ∪ {ps'}
12:             Execute(Π(idPartialPS))
13:         end if
14:     end if
15:     WaitEndStep()                                      ▷ Human and Robot actions are done
16:     ps ← AssessementProcess()                          ▷ Identify executed pair, and next ps
17: end while
```

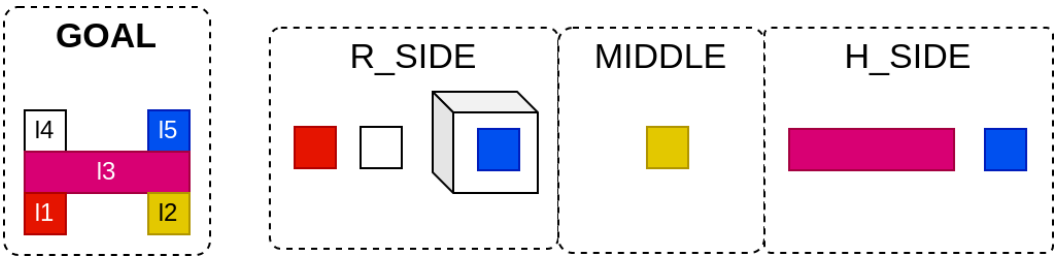


Figure 4.9: An instance of the BlocksWorld domain. The ideal plan is strongly influenced by the human desired preferences. For the earliest end of the task, the human prevents using the box. A lazy human will only place the required pink bar from their side. A human in a hurry will concurrently place the yellow cube to place the pink bar as soon as possible and be able to leave.

cubes can be inserted. The robot must first perform a dedicated action to open the box before being able to use the cubes inside it like the regular ones.

4.7.2 Human Behavior and Erroneous Preferences Estimations

To simulate human behavior, we consider and define human preferences that produce a human policy in the same manner as for the robot. The produced human policy makes the human always perform the best action regarding their defined preferences. The robot/-planner does not have access to human preferences but only to an estimation of them.

In order to evaluate the quality of the executed trace regarding the actual human preferences, we compare and rank every possible trace from the search graph, from best to worst. For legibility purposes, we normalize the ranks to obtain a score (H-score) s.t. the trace with the lowest rank has a score of 0.0, while the highest rank corresponds to a score of 1.0. This score represents a quality indicator independent of the instance's size.

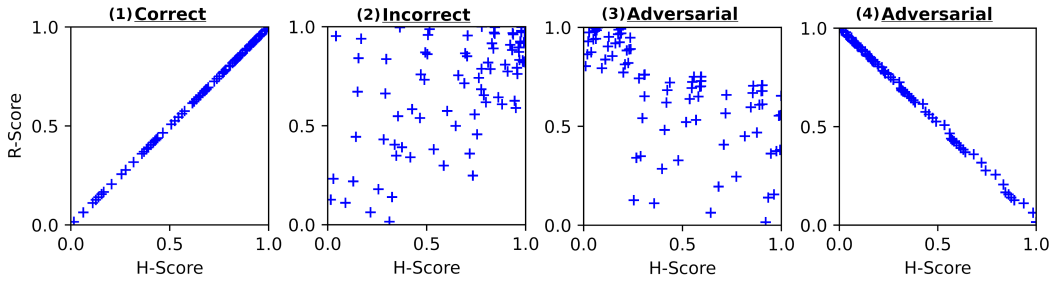


Figure 4.10: Correlation between the H-score and R-score according to different robot estimations. Four pairs of human preferences and their estimation are considered. For each pair, all possible traces are plotted as blue crosses according to their H-score and R-score. *(1)* show a correct estimation while the others show incorrect estimations. *(3)* and *(4)* depict *adversarial* estimations.

Similarly, we can do the same and acquire the score regarding the robot's estimation of the preferences (R-score). Keep in mind that the R-score is an estimation of the actual H-score, and the robot acts in order to maximize its R-score, hoping to maximize the H-score as well.

However, the estimation of human preferences can be more or less accurate, causing the robot's decisions to differ from what humans would have preferred. Once again, that is why making the robot compliant with human online decisions and actions gives the human more influence over the execution and helps to reach a high H-score even when the robot's estimation is incorrect. Despite the robot trying to maximize its R-score, it is essential to note that reaching a low R-score is fine as long as a high H-score is attained.

When solving the task, a pair of human preferences and their estimation creates a correlation between the possibly obtained H-scores and R-scores. Figure 4.10 depicts several possible correlations for the same task and the same search graph but different pairs of human preferences and their estimation. Each sub-figure shows all possible traces as blue crosses according to their H-score and R-score. Let's consider the first case where the estimation is perfectly accurate (correct). Here, the robot's choices that maximize the R-score will necessarily maximize the H-score. Indeed, when considering the few top possible robot plans (crosses with near 1.0 R-score), the human preferences are always well satisfied (near 1.0). Let's now consider the second case where the robot estimation is incorrect. When considering again the few top possible robot plans a wide range of H-score can be reached (near 1.0 as well as close to 0.0). Thus, an incorrect estimation can satisfy human preferences but not necessarily, leading to a wide range of H-scores. Eventually, consider the last two cases. Here, the lack of blue crosses in the top-right corners means that the H-score cannot be near 1.0 while also having a high R-score. Consequently, when maximizing the R-score, the robot will necessarily deteriorate the quality of the plan w.r.t. the H-score. We refer to these cases as *adversarial* estimations since the robot involuntary goes against the human will. Such cases occur when the robot estimation is far from the actual preferences and intuitively goes against the latter, for instance, the robot trying to minimize the human effort while the human is actually trying to do as much as possible.

4.7.3 Results

For the simulations, we first generated three problems of the BlocksWorld domain with different initial states and shared tasks, and we produced their corresponding search graph

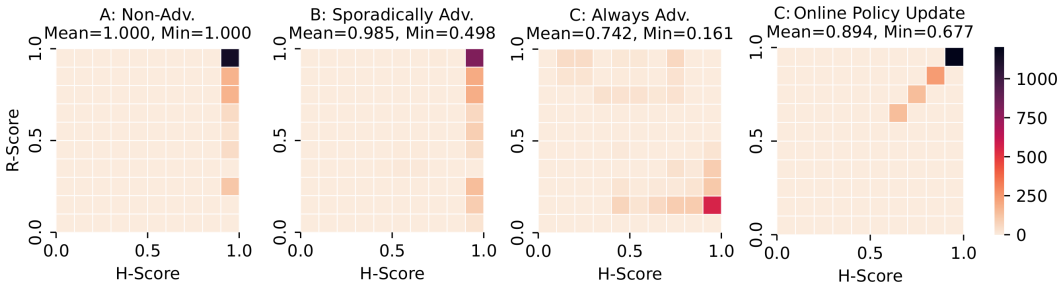


Figure 4.11: R-scores and H-scores of the obtained executed plans after simulating the execution of the robot and the human policy generated by considering three problems and three sets of pairs of preferences/estimations. The estimations in each set are (A) Never, (B) Sporadically, and (C) Always adversarial. On the right, the scores obtained using an enhanced human policy that can correct the robot's estimation online while using the set (C) are shown.

for each. After that, we generated numerous pairs of human preferences and associated robot estimation, and all those pairs were categorized into three distinct sets.

In Set A, the estimations are mostly correct and close to the human preferences (case *(1)* in figure 4.10). Set B includes incorrect estimations (case *(2)* in figure 4.10). And Set C contains only adversarial estimations (cases *(3)* and *(4)* in figure 4.10). Then, for each preference-estimation pair and each problem, we generated the associated human and robot policies. Their execution was simulated symbolically using an execution automaton directly shaped upon the presented Model of Execution, and the obtained executed traces were retrieved. The R-score and H-score of every obtained executed trace are shown as heatmaps for each distinct set of pairs in Figure 4.11. This will help us to highlight the benefits of using such an execution scheme.

In Set A, the estimation of the robot is close to the real human preferences and is never adversarial. So, the robot policies (maximizing the R-score) should naturally lead to high H-scores, which we observed. Some plans had an R-score lower than 1.0, which shows that the estimation was imperfect. Nevertheless, compliance with human actions and the robot's non-adversarial choices permit to always satisfy the maximal H-score of 1.0, i.e., human preferences are satisfied every time. With Set B, the incorrect estimations induced some detrimental robot choices, preventing the human from always reaching a score of 1.0. This is depicted by the minimal H-score of 0.498 obtained. Nonetheless, the average H-score of 0.985 indicates that the human preferences were overall largely met. Set C captures the worst possible estimations, inducing the robot to always make adversarial choices. This is depicted by the lower average H-score (0.742) and the very low minimal H-score obtained (0.161). However, we can notice that the average H-score is still high and that the R-score drops significantly. A low R-score means that the robot could not follow its policy correctly. Indeed, thanks to our model of execution, the robot complies with human online decisions and purposely deviates from its "optimal" policy to let the human follow their own optimal policy. Eventually, the relatively high H-score obtained shows that the compliance is effective and compensates significantly (of course, not totally) for a very poor estimation of human preferences.

Additionally, we can reasonably complement the human policy, which is based solely on preferences, with a *rule*. Whenever the robot performs an action that significantly degrades the best reachable H-Score, the human reacts by correcting the robot's estimation online. The rightmost sub-figure in Figure 4.11 shows the new scores obtained using the Set C and

the complemented human policy. We notice that correcting the estimation online avoids very low human scores (minimum of 0.677) and significantly increases the average H-score compared with the original Set C results (from 0.742 to 0.894). Hence, making the robot compliant with online preferences effectively improves the quality of the joint plan executed.

Overall, we can see that the compliant robot behavior regarding both online human actions and preferences benefits the collaboration thanks to the high human scores obtained.

Consider some counter-cases from social robotics (or HA collaborative planning). Assume a robot not giving the initiative to humans, always executing the best action it found. It is less acceptable and restricting for humans this way, even if the robot computed its best action by taking into account some social rules and estimated preferences. Unlike here, humans would appear compliant with the robots. In those cases, as evident from our simulation results in adversarial setups, the robot strongly impacts the solution H-score. Thus, wrong robot choices can significantly degrade human scores. In some sense, being compliant and adjusting to online preferences can be seen as some social factor that robots should maximize, and our framework helps achieve that.

4.8 Performances

In this section, we discuss the computational performances of our approach and provide some empirical measurements.

Our approach is based on exploring every decision the human is likely to make and every possible robot concurrent and compliant actions to those decisions. This offline search produces a Directed Acyclic Graph (DAG), which captures all possible courses of action. From this graph, we can easily extract online the optimal robot policy to adapt to and satisfy dynamic human preferences. However, this exhaustive approach does not scale since the more complex and lengthy the task is, the more possible coordinations and courses of action exist, and the longer it takes to produce the graph. Despite this existing combinatory explosion, I would like to discuss and explain why our approach is still relevant.

Two cases can be identified. The first one is in an industrial setup, and the other is in a household environment. Industrial tasks can be complex, constrained, and where mistakes can have heavy consequences. However, household tasks are usually simpler, shorter, and have fewer constraints and consequences. Due to the nature of industrial tasks, there is usually a known pre-defined protocol or plan to solve the task. The human is also in a working mental state, being more focused and willing to collaborate and accept robot decisions. In such setups, it is more acceptable for the robot to produce a joint plan and negotiate with the human to accept and follow it without anticipating every possible human decision, significantly reducing the planning algorithm's complexity. On the other hand, when considering household tasks, which is more our focus in this work, we want to preserve the human's latitude of choice as much as possible to allow them to change their mind, be distracted, or impose their choice. For this reason, it is adequate to run an exhaustive search and account for every possibility.

Overall Human-Robot Collaboration scenarios never require long plans, especially household tasks, and are often limited to about 20 actions per agent. For such length, our exhaustive approach performs efficiently. Our approach takes about $0.40s$ to produce the DAG for the BlocksWorld task described in chapter 6. This graph comprises 700 nodes and 6 leaves, corresponding to 6839430 different possible plans of length 19.77 ± 1.59 steps. With millions of plans, we explore sufficient possibilities to address such collaboration problems correctly. Moreover, it takes only $0.02s$ to extract the robot's policy from the produced graph.

4.9 Discussion and Limitations

This section discusses a few limitations of the proposed approach and possible future works to overcome them.

First, in order to explore relevant courses of action, we assume a step-based progression toward the goal. Hence, we assume that the human and the robot must synchronize together after every action. This can work efficiently as long as we assume that all actions have roughly the same durations. However, in practice, this is never precisely the case, and one agent must wait for the other at every step. Since the robot tends to be slower, the human might have to often wait for the robot during the collaboration with these assumptions. To be implemented on a real robot, this approach requires an additional execution scheme supervising the plan execution. In specific situations, this scheme could skip one synchronization and synchronize after the next step. Such a scheme could allow humans to perform more actions than robots before synchronizing, thus promoting smooth and flexible execution.

Speaking of execution, the results presented in this chapter have only been simulated symbolically, without durative actions nor real human decisions. The next two chapters (5 and 6) present a user study conducted on an interactive simulator where participants collaborated with a simulated robot executing the produced policy. To do so, we created an execution scheme that supervises the execution of the robot policy with agent synchronizations. This scheme is directly based on the model of execution presented in fig. 5.3. Thus, it relies on the steps and does not provide the flexible execution mentioned just above.

Additionally, in the proposed model of execution, the robot always gives the initiative to the human. We show that this decision makes the collaboration robust to erroneous human preference estimations and, thus, is beneficial. However, sometimes, it could be relevant for the robot to switch from follower to leader intelligently. Indeed, currently, even when there are no possible conflicts between agents' actions, the robot waits for the human to start acting to begin. Such synchronization is not really necessary, so the robot could start acting directly to solve the task faster.

Eventually, the plan evaluation is limited. For now, plan selection relies on the estimated human preferences, which are a list of metrics to maximize or minimize in the priority order given by the list. This means that there is no balance between the metrics so, depending on the ordering, a plan of length N where the robot is never intrusive and always compliant could be rejected against a plan of length $N-1$ where the robot is intrusive twice and never compliant. However, in our examples, we tend to explore numerous very similar possible plans. Thus, there are always several possible plans with the same top-priority metric value, e.g., plan length, and the ordering will help select the best plan that satisfies the other listed metrics.

4.10 Conclusion

We addressed the complex challenge of concurrent task planning for a shared goal in the context of human-robot collaboration, acknowledging the inherent need for autonomy in humans' choices of 'what' and 'how' aspects during task execution.

Based on studies about joint action, we formulate an execution model and present a new human-aware task planner designed to accommodate the uncontrollability factor inherent in human agents while employing this execution model leveraging social signals to facilitate the exploration of human-robot joint actions and smooth execution. We also propose a plan evaluation and selection based on estimations of the human inner preferences. As a

result, the planner produces the behavioral policy for a robot that complies with online human decisions and their estimated preferences, which can be updated online. This policy ensures that the task is solved, that the estimated human preferences are satisfied at best, and that the execution of concurrent joint action is smooth and sound.

We provide a detailed account of the novel planning process and joint action model. We demonstrated its effectiveness through symbolically simulated BlocksWorld scenarios and how our model makes the robot robust to erroneous preference estimations.

Additionally, as mentioned in the previous section, we implemented an interactive simulator in which a real human can collaborate with a simulated robot running the generated policies. We used this simulator to conduct a user study to validate our approach with real humans. This simulator and study are described in the next two chapters.

Chapter 5

Interactive Simulator

Contents

This chapter contains a technical description of an interactive simulator allowing human operators to collaborate with a simulated robot executing the policies described in the previous chapter. We used this simulator to validate our planning approach through a user study, which will be described in the next chapter.

5.1 Introduction

In order to validate the approach presented in the previous chapter (4), we would like to be able to perform a task collaboratively with a simulated robot executing the produced policy. This allowed us to conduct a user study described in the next chapter (chap. 6), which validates our planning approach.

This chapter is a technical description of the interactive simulator I developed. As depicted in fig. 5.1, it consists of several components detailed throughout this chapter. We describe the simulated scene corresponding to a BlocksWorld task similar to the one presented in the last chapter but with more cubes. After, the execution controller used is presented, corresponding to a refined and implemented version of the abstracted joint action execution model introduced in the last chapter. We continue by describing the different motion and simulation controllers used to interact with the simulator. Then, the Human-Machine Interface is presented, allowing participants to perform actions in the simulator. Finally, we present how the data retrieved during each execution are used to compute several metrics to evaluate the execution.

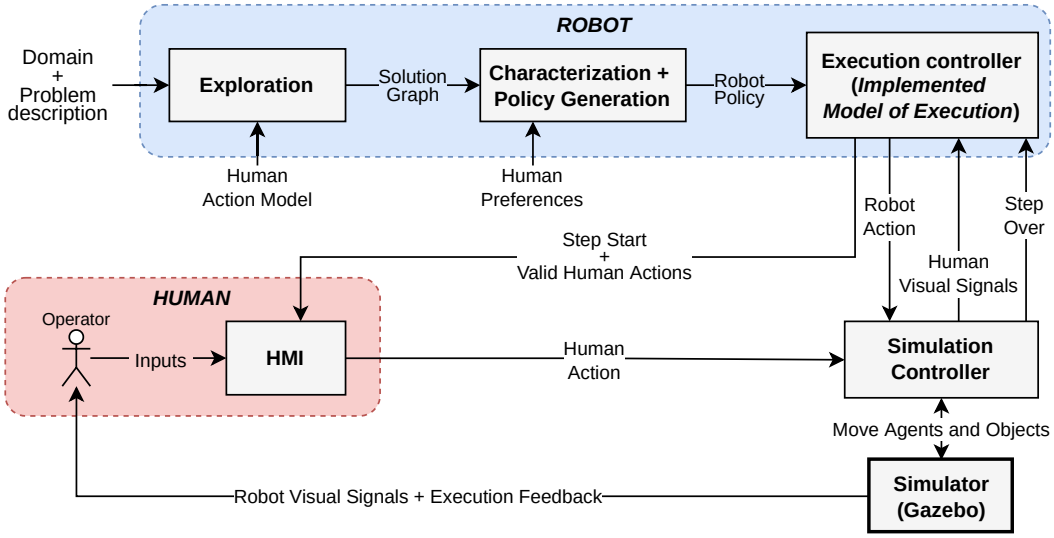


Figure 5.1: Overview of the Interactive Simulator's Architecture

5.2 Simulated Scene

This interactive simulator is based on the Gazebo Simulator. It is an open-source 3D simulator able to simulate articulated robots in a dynamic and complex environment. This section describes all aspects directly linked to this existing 3D simulator on which our framework has been developed.

The static scene consists of a large room, with a ground and four walls, and a table placed in the center with two marks on its right side. The marks represent the locations where the cubes must be stacked. The agents are represented as follows. First, the human is represented only with a floating hand as the camera simulates a first-person point-of-view. Then we used a Tiago robot from PAL Robotics because it has an arm to manipulate its environment, a head to send gaze information and signals, and because many related resources are available for it (3D models, controllers, tutorials). The human and the robot are facing each other, each from one opposite side of the table. Finally, colored cubes were disposed of on the table in three distinct zones. Cubes are either close to one agent's side or in the middle of the table. This disposition influences the reachability of the agents as one agent can only reach cubes from their side (just in front of them) or the middle, but never from the other agent's side. This scene corresponds to one specific collaborative task, but another setup could be easily implemented.

The Gazebo simulator can be integrated with the Robot Operating System (ROS), which is a set of software libraries and tools that help build robot applications. The main pros of ROS are its ability to run several sub-programs in parallel and allow them to communicate with each other. Hence, the other components of the interactive simulator have been developed with ROS.

The overall scene is depicted in the figure 5.2. The stacking goal is displayed in the top left corner, and a text prompt is shown in the top right corner so that the robot can communicate with the human.

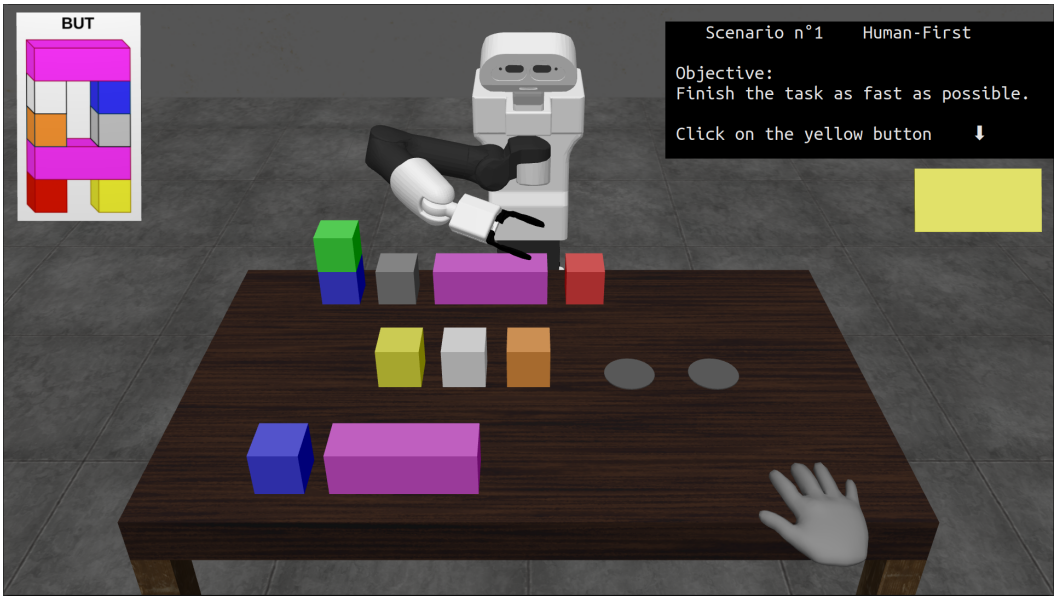


Figure 5.2: Participant view of the interactive simulator. It depicts the colored cubes on the table's left, the stacking area on the table's right (the two darker spots), the goal pattern on the top left, the collaborative robot in front, the text prompt on the top right, and the human hand at the bottom right.

5.3 Execution Controller - Joint Action Model for Execution

The model of execution presented in the last chapter was simplified and abstracted to be used to guide the planning algorithms and explore further relevant courses of action. This allows for anticipating possible coordination and compliance with online human decisions. However, we need an execution controller to supervise the execution of the produced policy. Here, we propose a refined and implemented model, matching the abstraction previously made, to be able to execute the produced robot policy and supervise its execution by synchronizing with the human through social signals.

The complete model is depicted in fig 5.3. On top of refining the 'robot automaton' present in the abstracted version, it also models the human agent's behavior with another automaton that captures the possible decisions the human may make during execution. Additionally, this complete model makes all the social signals exchanged between the agents explicit and shows how they are used for coordination. These differences are discussed in detail below. This execution controller takes as input the solution DAG produced by the approach of Chapter 4. Then, the controller progresses from the initial state to a goal in the solution graph. The controller handles the state transitions, step by step, by sending and synchronizing on social cues. Thanks to this controller and the associated simulator, we now have a durative task execution that effectively affects a simulated environment.

In the execution controller, the robot automaton is refined and more detailed. First, we show the social cues sent by the robot, which are the following: *START* corresponds to the visual and audible signal indicating to the human that the step begins; *S_RA* corresponds to the start of a robot action; *R_PASS* corresponds to the robot being explicitly passive. The other signals, including the human ones, will be described below. Also, we modeled

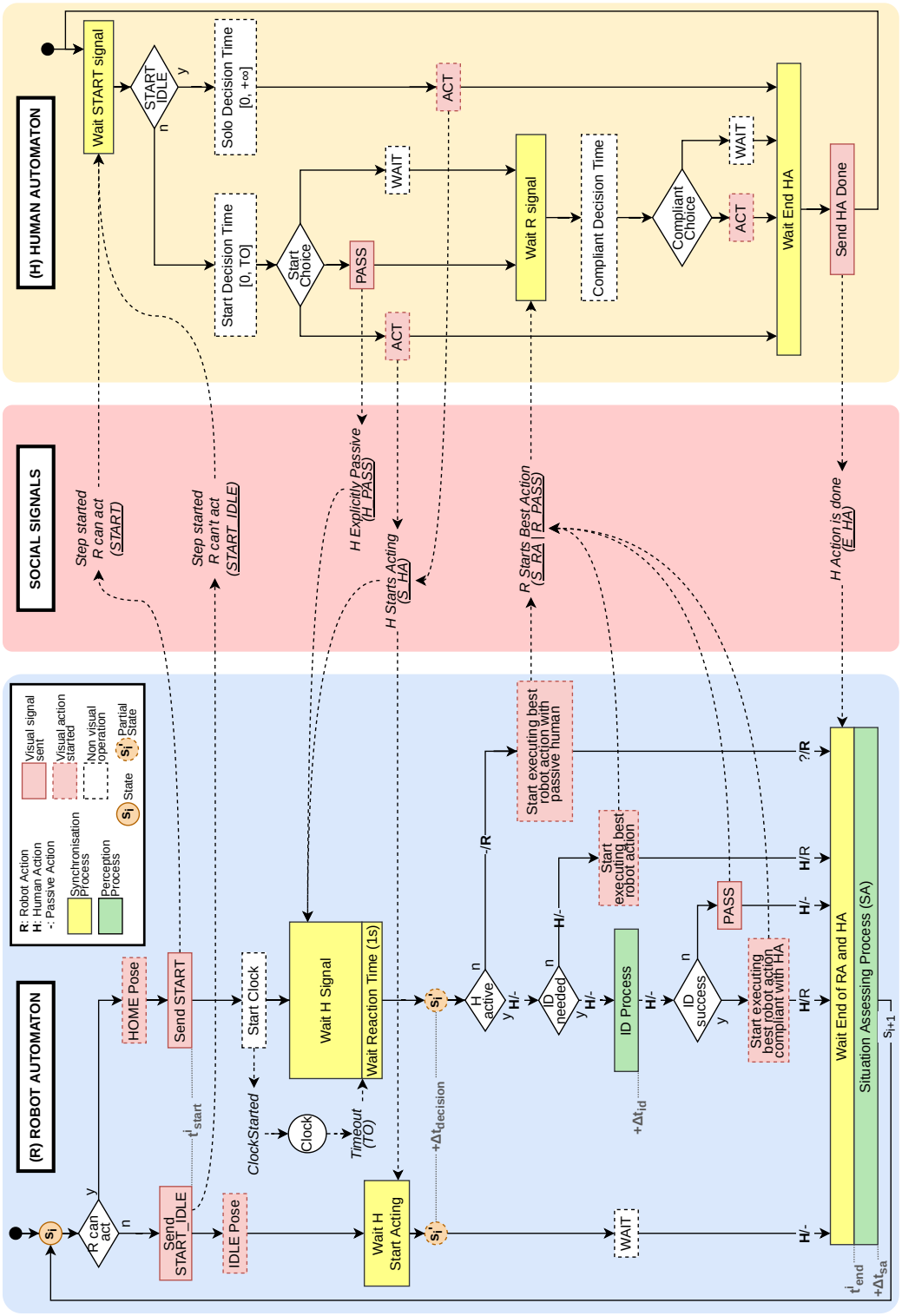


Figure 5.3: Complete Model of Execution - Dual Automaton. This version details the robot automaton and the assumed human automaton as well as the visual signals exchanges between the agents to synchronize themselves.

the case where the robot cannot act in the current step. This can be determined easily given the search graph and the current state s_i and by checking if there is at least one non-passive robot action in the following arcs. This checking process is represented in the figure by the diamond shape with the label "R can act", and the respective positive and negative answers "yes" (y) and "no" (n). When the robot cannot act, it sends a specific signal to inform the human that the step has started and cannot act (START_IDLE), then goes in the *IDLE* pose. This is a visually passive posture where the robot looks at the human with its arm retracted, indicating that it cannot act. Only a human action can progress to the next state. Thus, the robot waits for this action to start and eventually finish. Then, the Situation Assessment process identifies the executed action, and thus, the new state s_{i+1} to continue. Depending on the possible actions, the robot can stay in this *IDLE* pose for several steps. Once the robot can act again, the "yes" branch of the automaton is followed, and the robot returns to the *HOME* pose. In this posture, the robot's arm is deployed, showing that the robot is ready to grab objects on the table.

Additionally, the process of waiting for a human signal at the beginning of each step is more detailed. This process is interrupted by one of the three following signals: *H_PASS* is an explicit signal (hand gesture, verbal communication) sent by the human to share its desire to be passive; *S_HA* is the start of human actions, and is detected by tracking human's motion; *TO* is the internal timeout signal sent by a clock $4s$ after the start of the waiting process. However, the perception layer on a real robot would induce a delay in this synchronization process. An additional waiting time of $1s$ is added to compensate for the robot's reaction time, allowing the human to send signals just before the timeout. Without it, the human could start moving just before the timeout, but the robot would interpret this information too late and consider them as passive. Note that the robot only waits for a human signal if the human can act in the current step. This check is done similarly to the first diamond process. However, it is not shown in the figure for legibility reasons. Hence, it means that in situations where only the robot can act, it will not wait and will directly start acting.

The last visual difference with the abstracted model on the robot's side is the case where the human decides to be passive. In this branch, the human decision is pictured as a question mark. This is because we first identified the human as passive, and thus, the robot performs the best corresponding action provided by the policy. However, while the robot is acting, the human is free to start performing any action that does not conflict with the already-started robot's action. These possible decisions are modeled in the human automaton described after but are identified by the robot only during the Situation Assessment process.

On the other hand, a human automaton now explicitly models how humans should behave to coordinate with robots and collaborate successfully. This automaton starts by always waiting for the beginning of the current step, indicated by the robot with the signals *START* or *START_IDLE*. The simplest but less common case is when the robot cannot act during the step (after receiving *START_IDLE*). This case is depicted in the rightmost branch. Since the robot cannot act, only the human can make progress in the task by acting. Thus, the human is free to take as much time as desired, represented by the decision time in the range $[0; +\infty]$. After the start of the human action, both human and robot automatons wait for the end of the human action, and then they proceed. Let us discuss now the most common case where the robot can act. This time, the human must decide before the defined robot timeout (TO), otherwise they will be considered passive. The human can decide either to start acting (ACT), to indicate their passivity (PASS), or do nothing corresponding to being passive without informing the robot (WAIT). If the human decides to be passive (PASS or WAIT cases), they can either remain passive until the next step or

decide to perform an action not conflicting with the robot one. This is represented by the last human diamond shape labeled "Compliant choice" made after a duration referred to as "compliant decision time".

5.4 Simulation controllers

Four distinct controllers have been developed to control the simulated agents and the overall simulation execution. Each controller is described below.

5.4.1 Robot arm motion controller

First, there is one controller that moves the robot's arm. This controller is called with either a 3D position to reach (pose target) or a predefined configuration (named target). In both cases, when called, this controller uses the MoveIt framework. MoveIt creates links between several libraries in order to have access to a unified interface for Motion Planning algorithms, Inverse Kinematics, Control, or Collision Checking. This way, our controller can "simply" request MoveIt to find a trajectory to a given position and then make the robot follow this trajectory while taking into account the geometry of the robot arm.

In fact, this controller is slightly more sophisticated. Indeed, in our collaborative context, we prefer the robot to be reactive rather than to have optimal motions. This is why, through the MoveIt interface, we use two different motion planners. The first one is RRT^*, which is an optimal planner that stops only when the optimal motion plan has been found. It is desirable to have the robot exhibit consistent and efficient motions. However, this process sometimes takes too much time (4-5s), which breaks the rhythm of the interaction. As a consequence, a short timeout (0.6s) has been set for this optimal motion planner. If the optimal plan is not found within the defined timeout, we use the second motion planner. The second planner is SBL, which is not an optimal planner. That is, the planner stops when a solution is found, but there is no guarantee of optionality. The major pro of this planner is its speed. Overall, for every robot arm motion, we first try to find an optimal motion within a short amount of time. If the optimal motion is not found, we quickly find a suboptimal solution to prevent the robot from being passive for too long.

Additionally, to reduce motion planning time, a few simplifications have been done. First, collisions are only considered with the table and the robot's body. Hence, the robot arm sometimes goes through the other cubes. Also, the pick and place orientation are ignored. When picking a cube, the robot gripper just reaches the cube's center from any angle, and then the cube is attached to the gripper. When placing a cube, the robot moves its arm to the target position, and the cube is detached from the gripper. Eventually, the cube's orientation and position are overwritten to match the target location. This way, the robot always performs perfect place action, improving both the reactivity of the robot and its robustness, but the robot's motions are less realistic.

5.4.2 Robot head motions

The robot head is controlled to look at various elements during the interaction to exhibit a more intuitive and collaborative behavior. Indeed, when waiting for the human, the robot looks at the camera. As soon as the human hand moves to either perform an action or indicate its passivity, the robot starts following the hand to show it is observing the human motions and estimating their intentions. Then, the robot looks at a cube to indicate its

intention to pick it since this is faster than the arm motion. Also, the robot looks at the target position when placing a cube to indicate its intention to place the cube there.

A dedicated controller has been developed to perform those head motions based on an existing controller provided in the Tiago robot modules. The existing controller could only change the robot's gaze through a visual and clickable window, allowing an operator to manually click on the scene to change the robot's gaze. This controller has been modified to allow additional features, such as directly requesting to look at a 3D point or an object. Moreover, we can now ask to follow an object. That's how we are able to exhibit the head behavior described just above.

5.4.3 Human hand motions

Moving the human hand requires a dedicated controller but is much simpler than the robot arm motion one. Here, the hand must either move to a position or perform a PASS signal. The PASS signal is a hand gesture indicating to the robot that the human desires to be passive for the current step. This motion is performed by simply rotating the hand back and forth at a constant speed. The PASS motion has a duration of 0.7s. On the other hand, we do not use a real motion planner to move the hand to a target position. We simply compute a straight line between the current hand pose and the target pose. Then, we update the hand position at $50Hz$ to move at a constant speed set to $0.25m/s$. The planning time is thus negligible compared to the robot motion planning times.

5.4.4 Simulation controller

5.4.4.1 Action decomposition

Last but not least, I developed a fourth controller whose main job is to translate the actions from the plan produced by our task planner to low-level motions that can be executed by the controllers listed above. Hence, some parts of this controller are domain-specific, but a major part is generic for manipulation tasks. This controller received high-level agent actions to execute, such as "Robot pick(b1)" for the robot picking the cube b1. The first step is to decompose this action into low-level generic actions, which are themselves made of low-level motions. This controller also keeps track of a few low-level facts to check low-level preconditions, such as what each agent is holding or which object is still on the table. Hence, the controller throws an error when an agent tries to pick a cube while already holding one.

Let's first comment on the low-level generic actions available. This controller is given a list of the object names (cubes) and the names of some predefined locations in the scene (placing locations in the stack). Except for the last five, the following actions are not agent-specific and can be performed by any agent. This is defined by a parameter given when calling the action. The following low-level actions are available:

- **move_pose_target:** moves either the hand/robot arm to a given position.
- **move_location_target:** moves either the hand/robot arm to the position of a given location.
- **move_obj_target:** moves the hand/robot arm to the position of a given object.
- **move_home:** puts the agent in its "home" (default) configuration.
- **move_named_target:** moves the robot arm to the given configuration.
- **grab_obj:** attaches the given object to the hand/robot gripper.

- **drop_obj:** detaches the given object to the hand/robot gripper.
- **set_obj_rpy:** sets the orientation of a given object.
- **set_obj_pose:** sets the position of a given object.
- **delta_move_obj:** sets the position of a given relative to its current position.
- **human_hand_gesture:** makes a hand gesture to indicate passivity.
- **robot_head_look_pose:** makes the robot look at a given position.
- **robot_head_look_human:** makes the robot look at the human (camera).
- **robot_head_follow_obj:** makes the robot follow a given object.
- **robot_head_follow_hand:** makes the robot follow the human hand.

Then, the high-level actions from the plan produced are the following with their respective simplified low-level decomposition:

- **PickCube:** Starts by retrieving the cube's current position by sending a request to the Gazebo simulator. If it is the robot, call robot_head_look_obj to look at the cube. Then, call move_pose_target with the retrieved cube position. Once over, grab the cube with grab_obj. After, move_home is called to retract the robot arm or bring the hand to its initial position. Finally, if it is the robot, it looks to the human with robot_head_look_human.
- **PlaceCube:** Starts by retrieving the position of the target location to stack the cube. If it is the robot, call robot_head_look_pose with this position. Then, we move either the arm or the hand to that position with move_pose_target before dropping the cube in the stack with drop_obj and adjusting its position. After, we go back to the home configuration with move_home. And for the robot, we look at the human with robot_head_look_human.
- **BePassive:** The human waves their hand to explicitly be passive. Thus, human_hand_gesture is called. The robot does not move and only displays some texts saying the robot wants to be passive. More details about the text prompts will be provided later.
- **DropCubeTable:** When an agent cannot stack a cube they are holding, they can place it back on the table. First, the drop position is defined. It corresponds to the initial position of the cube being held, except for the green one, which is initially on top of another cube and has a dedicated drop position. The robot looks at the drop position with robot_head_follow_pose. The hand or the robot arm is moved to the position with move_pose_target. The cube is dropped with drop_obj, and its position is adjusted. Then, the agent is put in its home configuration with move_home, and the robot looks at the human with robot_head_look_human.

5.4.4.2 Manage steps

The simulator controller is also in charge of indicating when a step is over. Since we do not consider steps where both agents are passive, a step begins when an agent starts an action. A step is over when both agent actions are done. These rules cover many situations, such as if the human initially wants to be passive and wave their hand. As a result, the robot starts performing an action, which starts the next step. Currently, the step would be over as soon as the robot's action is over since the human is passive. However, if the human decides to start performing an action concurrently, then the step will be over when both

the robot's and human's actions are over. This may imply that if the human starts acting right before the end of the robot's action, then the robot will just wait for the end of the human action, and thus, the end of the step before the next step begins.

5.4.4.3 Send visual signals

The model of execution synchronizes the agents based on explicit visual signals such as the start of a step, the start of an action, the end of an action, or a hand gesture (PASS). Those signals are modeled explicitly inside the system and managed by the simulator controller.

Indeed, when starting an action, the associated visual signal is sent only when the agent starts to move. Hence, we track the arm and hand motions to know when the action is visible to the other agent. Then, when an action is over, the associated visual signal is sent directly. It is worth mentioning that humans will naturally have a reaction time when seeing the robot's visual signals (start/end of action, step start). However, the robot natively does not have any reaction time to such symbolic visual signals. Thus, to simulate the delay introduced by an actual perception module (here perfectly simulated), the human visual signals are delayed to the robot by a reaction time set to 0.3 seconds. This is still quite fast, but at least the robot does not interpret human motions instantly.

5.5 Human Machine Interface (HMI)

The human operator, or participant, can interact with the simulation by performing any feasible action during the process. However, humans must synchronize with robots by following the execution model. For this purpose, every step starts with a robot sound/beep. Additionally, a dedicated process receives internally the list of the feasible human actions for the current step. This list is sent by the robot execution scheme, and is extracted from the solution DAG. However, the participants do not show or know of feasible actions. Each possible human action is associated with a rectangular area/zone on the screen. When the human clicks in a zone associated with a currently feasible action, the process requests the simulation controller to perform the corresponding human action.

5.6 Logs and timeline

There is also a dedicated process to record the different events and signals during one execution. All other components can send events to log to this process to save them for later post-processing. A log event is defined with a name and a time stamp. Additionally, visual signals are also captured by this logging process and saved both as a signal and as a corresponding event. Logged events mostly help trace the robot's internal states and processes during the execution. The human ones mainly serve to identify when the human is acting or not since we do not have access to the internal reasoning process of the human participant.

5.6.1 Activities extraction

All logs are saved after each execution and can be loaded later for post-processing. The post-processing involves three steps. First, the agent activities are extracted from the event list. The human and the robot have different possible activities, but since we do not have access to internal human reasoning, there are more robot activities than human ones. The extracted activities are shown in the following table 5.1. The activities' orders try to match

Robot Activities	Human Activities
Waiting for human decision	Decision Time
Identification Process (ID process)	Human Action
Planning arm motion	
Robot Action	
Waiting for human action when itself cannot act	Being passive with signaling (after PASS hand gesture)
Being passive	Being passive without signaling (after TimeOut is reached)
Waiting for the end of the step	Waiting for the next step to start
Situation Assessment Process (SA process)	
Getting ready for the next step (GRNS)	

Table 5.1: Agent activities extracted from logs. The activities' orders and relative positions try to match the execution, but not all activities are present at every step.

what can happen during the execution, but not all activities are present at every step. Indeed, on the robot side, the ID process is not always necessary and thus not always executed, and the robot either performs an action or is passive for diverse reasons. But the robot always ends up waiting for the end of the step, performing the SA process and getting ready for the next step. On the human side, if the human performs an action or makes a hand gesture, there will be a measurable decision time. If the human remains passive without signaling the robot, then the only activity is passive without signaling. The "waiting next step to start" activity is only added if the human is active. Otherwise, being passive for a step is implicitly equivalent to waiting for the next step.

5.6.2 Metrics computation

When loading the execution logs, we extract a set of metrics from the events and extracted activities. Currently, 27 metrics are extracted, but the following table 5.2 only shows 11 because several sub-metrics are hidden. Indeed, for four of the listed metrics, we compute the sum, average, standard deviation, min, and max values of the corresponding metric.

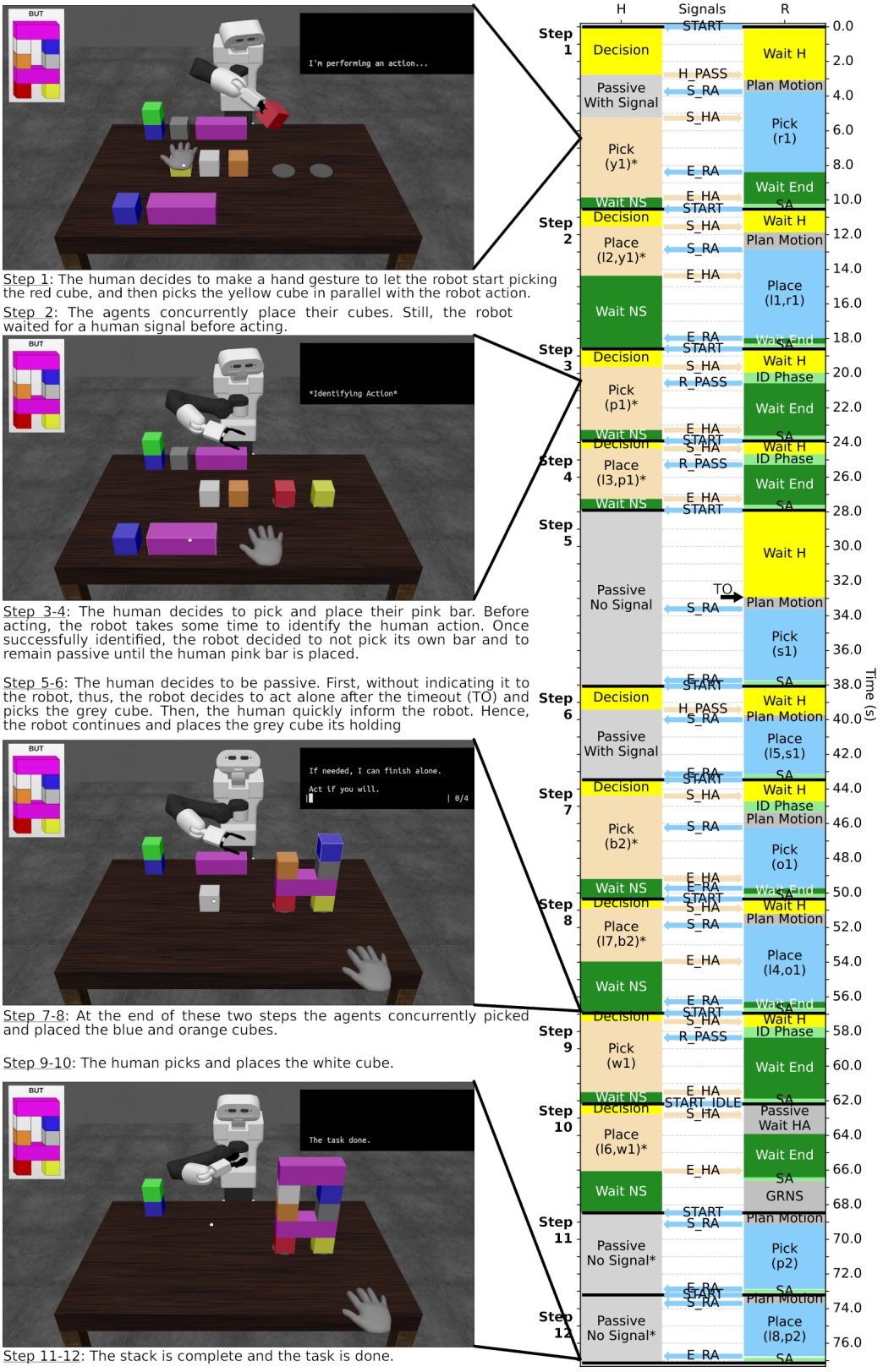
5.6.3 Execution example with timeline

A complete scenario is presented and commented on in this section. Using the previously extracted agent activities, we can draw a visual timeline depicting the different steps, the extracted activities, and the visual signals exchanged between the agents. Figure 5.4 depicts the complete commented timeline, including agents' activities, exchanged signals, and snapshots from the simulator. The table 5.3 describes the different signals exchanged during the execution and shown on the timeline.

In this execution example, according to the extracted metrics, the task has been completed in $77.14s$ in 12 steps. The average human decision time is $1.00s \pm 0.69s$ with $max = 2.77s$ and $min = 0.42s$. The human waited on average for the start of the next step during $1.68s \pm 1.28s$. There were 8 human actions with an average duration of $3.67s \pm 0.72s$. According to the objective/preferences to finish the task as fast as possible, the human acted 9 times optimally out of 12 steps (optimal ratio= 75%). Indeed, the human agent purposely decided to be passive during steps 5 and 6, but they could have picked the orange cube in parallel to complete the stack faster. In this example, the task description forbids agents

Metric	Description
Task Completion Time	Time for the task to be completed.
Number of steps	Number of executed steps.
Number of optimal human actions	Number of human action being optimal regarding a given objective.
Ratio of optimal human actions	Number of optimal human action divided by the number of steps.
Human decision time*	Duration for the human to make a visible decision in a step.
Waiting next step*	Duration during which the human waits for the next step to start.
Number of human action	Number of non-passive human actions executed in the scenario.
Human action duration*	Duration of the human actions.
Number of robot action	Number of non-passive robot actions executed in the scenario.
Robot action duration*	Duration of the robot actions.
Time human free	Time after which the human is free, i.e., as soon as the robot can finish alone.

Table 5.2: Metrics extracted from the execution logs for each scenario. Marked items with a star (*) correspond to five sub-metrics computed over all steps: sum, average, standard deviation, min, and max values.



Step 1: The human decides to make a hand gesture to let the robot start picking the red cube, and then picks the yellow cube in parallel with the robot action.

Step 2: The agents concurrently place their cubes. Still, the robot waited for a human signal before acting.

Step 3-4: The human decides to pick and place their pink bar. Before acting, the robot takes some time to identify the human action. Once successfully identified, the robot decided to not pick its own bar and to remain passive until the human pink bar is placed.

Step 5-6: The human decides to be passive. First, without indicating it to the robot, thus, the robot decides to act alone after the timeout (TO) and picks the grey cube. Then, the human quickly inform the robot. Hence, the robot continues and places the grey cube its holding

Step 7-8: At the end of these two steps the agents concurrently picked and placed the blue and orange cubes.

Step 9-10: The human picks and places the white cube.

Step 11-12: The stack is complete and the task is done.

Figure 5.4: Complete execution timeline commented.

Signal name	Description
START	Visual and audible signal from the robot indicating that the current step has started.
START_IDLE	Visual and audible signal indicating the start of a step where the robot cannot act.
S_RA / E_RA	Signals sent respectively from the start (S) and the end (E) of a robot action (RA).
S_HA / E_HA	Signals sent respectively from the start (S) and the end (E) of a human action (HA).
H_PASS	Explicit hand gesture from the human informing their desire the be passive.
TO	The TimeOut is an internal robot signal indicating that the absence of human signal.

Table 5.3: Signals exchanged between the agents during the execution and used for synchronization and coordination.

from picking cubes in advance. They must be able to immediately place a cube in the stack to pick it. This rule is detailed and justified later in the next chapter 6.3. Hence, in step 9, the human picked the white cube, but the robot could not pick the bar until the white cube was placed. Thus, the human could have let the robot place the white cube to reduce their effort without slowing the task. There were 8 robot actions with an average duration of $3.97s \pm 0.69s$. Overall, the robot took $5.25s$ to plan its arm motions.

Chapter 6

User Study to evaluate an integrated plan and execution scheme in simulation

Contents

This chapter presents a user study validating the approach proposed in Chapter 4 using the simulator described in Chapter 5. For this purpose, several scenarios have been designed using a BlocksWorld task, and human participants were asked to collaborate with the simulated robot to evaluate its behavior. We compared our approach with a baseline behavior where the robot always imposes its decisions on the human. This study uses objective and subjective metrics to show that our approach performed significantly better than the baseline.

6.1 Introduction

To validate the approach presented in the previous chapter, we conducted a user study of more than twenty participants. The purpose of this study is two-sided. First, we want to validate our overall planning approaches. Thus, we want to show how it allows successful collaboration with humans. Secondly, we want to validate our model of concurrent and compliant joint action, that is, showing how it allows the human to always be the leader and able to decide while the robot follows concurrently. We use a baseline where the robot imposes its decision on humans, and we show how our model allows satisfying human preferences better and is thus preferred.

We decided to conduct this study in simulation for various reasons. First, one of our assumptions is that all actions should roughly have the same duration. However, real-life robots are slow and not very reactive. Those aspects may bias the results of our study, which is focused on decision-making. Secondly, simulation allows several simplifications that are acceptable for study. Collision with the cubes has been disabled to make the robot faster both in planning and executing its arm movements. In addition, simulation allows for a perfect perception of the environment. In a real-life experiment, perception errors may occur, leading to replan and thus slower execution or even wrong decisions. Moreover, our model assumes that both agents synchronize after each step. Hence, it was easy in simulation to prevent the human from acting too soon and synchronize automatically their actions. In a real-life scenario, we could not physically prevent the participant from acting. This would imply a heavier training process for the participants to avoid desynchronizing with the robot. In practice, an additional execution supervisor should be developed to permit desynchronizing as long as they are not too big and hence, prevent the system from crashing. This would require a significant technical effort to implement.

To conduct this study, I developed a dedicated interactive simulator using a Tiago robot. In addition, the automaton described by the MoE has been implemented and integrated with the simulator to provide a proper execution and supervision scheme. Eventually, through carefully designed scenarios and using a shortened version of the PeRDITA questionnaire [Devin 2018], we gathered the feelings and impressions of the participants regarding the different robot behaviors. We also recorded logs from each executed scenario, allowing us to draw a timeline of the execution and compute objective metrics for each scenario, among which can be found the time to complete the task, the human decision time, or the time for the human to be free. Several relevant facts and conclusions can be extracted from the collected results, which are discussed in this chapter.

This chapter is organized as follows. First, the interactive simulator functionalities and operations are described. Then, the methodology of the user study is provided along with anonymous information on the participants. After that, the results obtained are presented and discussed, validating the proposed approach and our model.

6.2 Existing HRI questionnaires

Many questionnaires are used in the field of HRI. The main ones are GodSpeed, HRIES, PeRDITA, RoSAS, and Trust Perception Scale-HRI.

Each questionnaire has specificities and helps to measure certain aspects of the robot. Many include appearance items to evaluate the look of the robot. Since our focus is on robot decision-making, we decided to base our questionnaire on PeRDITA. Indeed, this questionnaire has been designed to evaluate the pertinence of robot decisions in a Human-Robot Joint Action Context, which is exactly our case. Yet, the full questionnaire is a bit heavy and also covers communication, which is not our topic in this work. That is why we decided to shorten the questionnaire by removing the section on communication and a few redundant items. Redundant items are helpful to evaluate the consistency of a questionnaire, and this has already been done in [Devin 2018]. Hence, to avoid participants getting bored and lost, we filled out the whole questionnaire. After every scenario, we kept 12 items covering the following dimensions: robot perception, interaction, collaboration, and acting.

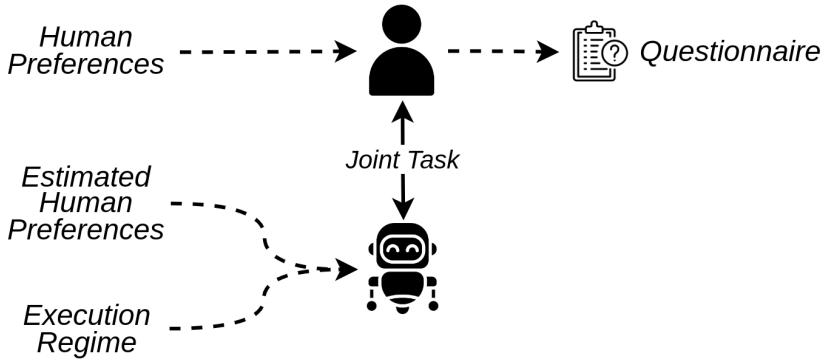


Figure 6.1: A scenario of the User Study Protocol. Each participant goes through six scenarios and answers six questionnaires to evaluate each different robot's behavior.

6.3 Study protocol

In this study, each participant is made to collaborate six times with a simulated robot to achieve a shared task. Each occurrence is referred to as a scenario. The robot exhibits different behaviors in each scenario. After each scenario, the participant evaluates the robot's behavior through the PeRDITA questionnaire [Devin 2018], and logs about the execution are saved.

Beforehand, every participant answers a few demographic questions and is familiarized with the simulator functionalities through an integrated tutorial. Only then the participants start the six consecutive collaborative scenarios, answering a questionnaire to describe the interaction every time. Eventually, every participant is asked to share their general feelings and impressions about the overall interaction with the simulated robot, and they are asked to tell which scenario they preferred the most and the least.

We now provide details about the task, the scenarios, and how the different robot behaviors are generated. The shared goal, which is stacking the cubes to match the given pattern, remains the same in all scenarios. The cube disposition on the table also does not change either. The task description is depicted in fig 6.2. For this problem, our planning approach generated a solution graph with 700 PStates/nodes leading to 6 different final states/leaves. This solution graph comprises 6839430 different possible courses of action. The length of the plans is about 19.77 ± 1.59 steps, with a minimal length of 11 steps and a maximal length of 23 steps.

To progress in the task, the agents can perform three different primitive actions, which are the following: *pick* a cube, *place* a cube in the stack, or *drop* a cube back on the table. These actions have a few preconditions, more or less intuitive, that are communicated and experienced by the participant during the integrated tutorial. First, one can *place* a cube if they hold it and if the targeted location is free and supported. That is, the cubes directly below the targeted location must be placed before being able to place a cube in the targeted location. Secondly, one can only *pick* a cube from their respective reachable zones of the table, i.e., Human and Center zones for the human and Robot and Center zones for the robot. Also, one can only pick a cube if it can be placed immediately. Thus, one cannot pick a cube "in advance" and must wait for its placement condition to be true before picking it up. For instance, both pick bars can only be picked up after the yellow and red cubes

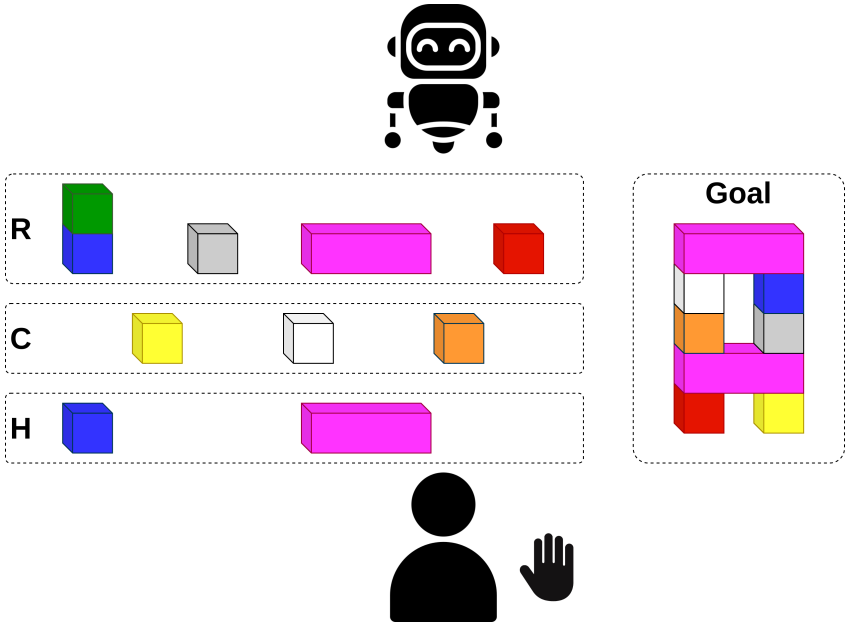


Figure 6.2: Description of the shared task to achieve in the study.

have been placed. This rule helps to create interaction conflicts serving the purpose of this study. Moreover, although the participants found this not intuitive, they got used to it quickly, and this feeling seemed to be significantly reduced during the experiment. Third, one can *drop* a cube back on the table only if they hold it and if it cannot be placed.

For each scenario, the participant is given instructions on how to solve the task. The participants are asked to consider these instructions as their own choice and preferences regarding the task resolution and, thus, to act accordingly while collaborating. The instructions for each scenario are one of the two following. On the first hand, the participant shall act in a way to finish the task as soon as possible. Here, it consists of trying to perform as many actions in parallel as possible to progress faster. These preferences are later referred to as Task End Early (TEE). On the other hand, the participant shall act in a way to be freed as soon as possible. That is, they should finish their mandatory part of the task as soon as possible so they can leave and let the robot finish alone. Here, it consists in placing the pink bar from the Human zone as soon as possible. These preferences are later referred to as Human Free Early (HFE). On its side, the robot does not directly have access to these instructions/preferences. Hence, for each scenario, the robot is given a more or less accurate estimation of the human preferences that are communicated to the participant. Note that the participants are not aware that the robot has an estimation of their preferences, nor that this estimation can be inaccurate. This way, we created three scenarios with different pairs of human preferences and associated estimation. In the first pair, the human shall finish the task early, and the robot has a correct estimation, i.e., the robot's policy helps the human finish the collaborative task early. In the second pair, the human preferences remain the same, but the robot estimation is incorrect. The robot is trying, mistakenly, to minimize the human effort. As a consequence, the robot tends to pick cubes that the human could pick, preventing the human from acting and making the task completion longer. In the third pair, the human shall free themselves early, but the robot estimation is again erroneous. The robot will try to finish the task early while its priority is to place the first pink bar, which conflicts with the given human preferences.

Table 6.1: Name of the six scenarios. Columns represent the preferences/estimation pairs, and the rows correspond to the execution regimes.

	Pair A TEE: correct	Pair B TEE: incorrect	Pair C HFE: incorrect
Human-First	S1	S3	S5
Robot-First	S2	S4	S6

Additionally, in each scenario, the robot follows one of the two following execution regimes:

- **Robot-First (RF)**: the robot always initiates actions first, and the participant takes action afterward.
- **Human-First (HF)**: the robot always lets the participant take the initiative and acts after.

The *Human-First* execution regime corresponds to the Model of Execution described in the previous chapter. At each step, the robot waits for the human's decision and will execute the best action that complies with it. The human always starts acting first, and the robot follows. On the other hand, the *Robot-First* regime corresponds to a naive and straightforward policy execution where, at each step, the robot directly starts executing the overall best robot action given by the policy. The robot always starts acting, forcing humans to comply. The *Robot-First* regime serves as a baseline to evaluate the proposed *Human-First* regime, described by our Model of Execution and used in policy generation. Eventually, we associate each of the three previous pairs of preferences and estimation with one of the two different execution regimes. As a result, we obtain six different scenarios with six different robot behaviors named in table 6.1.

Note that our goal is to evaluate and compare the different robot behaviors. However, at the beginning, the participants do not have any references to compare with, which can influence their answers in the very first scenarios. One solution is to ask the participants to answer all six questionnaires at the end after being familiar with the six scenarios. We consider that this option demands a too heavy mental workload to recall accurately each specific scenario and may bias the answers. As a consequence, we decided to ask the participants to answer the questionnaire after each scenario as a draft. During the experiment, they can rectify their answers to match their feelings more accurately. At the end, using the drafts, they share their final answers for each scenario. We believe this process gathers the feelings of the participants more accurately. Moreover, the ordering in which the participants encounter the scenarios is uniformly randomized to prevent any order effect.

The questionnaire filled by the participants after each scenario is a shortened version of the PeRDITA questionnaire, and its items are gathered in table 6.2. In addition to the questionnaires, for each scenario, the interactive simulator produces logs from which we extract several metrics and an overall timeline of the execution. The timeline depicts the activities and actions of each agent along the progression of the task. The subjective measures done through the questionnaire are complemented with the objective metrics extracted, such as the duration to complete the task, the number of human actions, the total duration of human inactivity, and more.

Dimension	Question	Item
Robot perception	In your opinion, the robot is rather:	Apathetic/Responsive Incompetent/Competent Unintelligent/Intelligent
Interaction	In your opinion, the interaction with the robot was:	Negative/Positive Complicated/Simple Ambiguous/Clear
Collaboration	In your opinion, the collaboration with the robot to perform the task was:	Restrictive/Adaptive Useless/Useful Inefficient/Efficient
Acting	In your opinion, the robot choices of action were:	Inappropriate/Appropriate Annoying/Accommodating Unpredictable/Predictable

Table 6.2: PeRDITA Questionnaire: Participants have to place themselves between the two antonym items on a scale of 7.

There are a few restrictions on the actions that can be performed. First, an agent can only pick cubes that can be placed immediately. This means that agents cannot pick cube in advance to anticipate each other's actions. Allowing such behavior could generate a very interesting scenario. However, here, we want to purposely generate some conflicts to evaluate the robot's behavior and reactions. Without this restriction, the agents would have too much flexibility in their actions and decisions, making it harder for conflicts to happen. Additionally, when holding a cube, the agents can only place the cube in the stack on back to its original place. As a result, the agents cannot displace the cube on the table to make them reachable to the other agent. This restriction has been added for the same reasons as the first one and simplifies the conflict generation.

The participants were collaborating with the robot using a mouse. The simulation was run on a laptop connected to a bigger screen, allowing participants to see the simulated environment clearly. After each scenario, participants answered the printed questionnaire using a pencil. Figure 6.3 depicts the execution of a scenario where the participant collaborates with the simulated robot using the mouse. Once the task is completed and the scenario is over, the participant is asked to fill out the printed questionnaire on the desk with a pencil. For each scenario, a new paper sheet is provided to the participant.

6.4 Participants

This section shares and analyzes some information about the 25 participants involved in the study.

The participants were aged of 29.64 ± 10.67 ($min = 21$ and $max = 62$), as depicted on figure 6.4.

About 36% (9/25) of the participants indicated they were *Females* and 64% (16/25) indicated being *Males* (figure 6.5). Participants could choose not to share their gender, but none selected this option.

Participants were asked to rate their opinion about robotics from 1 (negative) to 5 (positive). On average, participants had a highly positive opinion about robotics: 4.24 ± 0.6, without answers below 3. Most participants were excited about this subject.

Only 6 participants were not from my lab. Nevertheless, various profiles were requested to participate. Among all participants, 8 were more or less familiar with robotic decision-making. All other participants were not familiar with robotic decision-making, like some

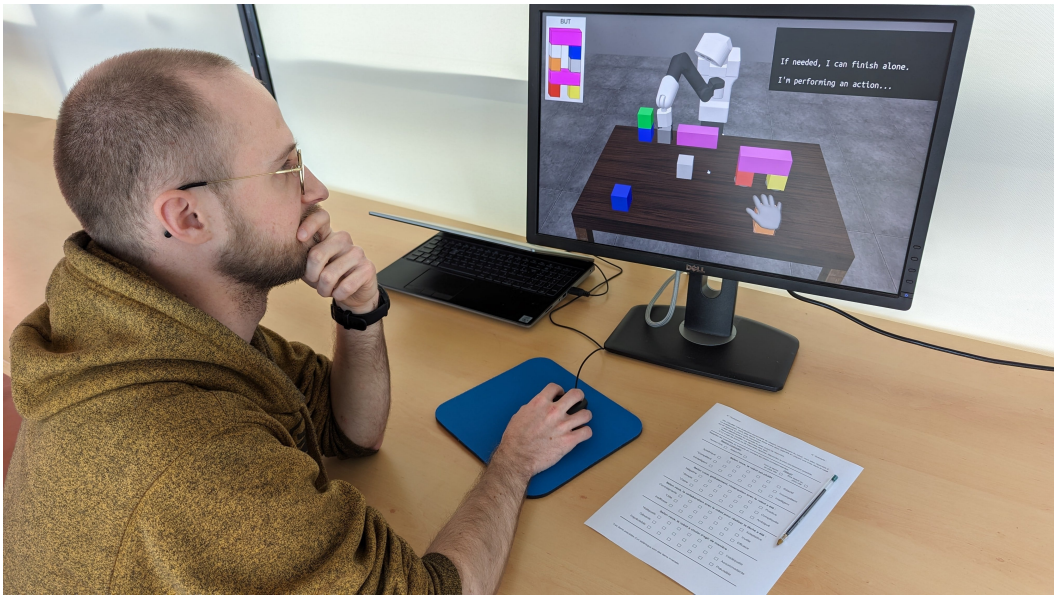


Figure 6.3: One scenario execution where a participant is collaborating with the simulated robot using the mouse. Once the task is completed, the participant is asked to fill the questionnaire on the desk with a pencil to transcribe their impressions.

biologists. However, 12 participants were generally familiar with robotics, like drone control researchers.

We conducted numerous correlation analyses between the participants' personal information (age, gender, affiliation, familiarity with robotics), the questionnaire answers, and the execution metrics. Due to the considerable possible combinations, we could not exhaustively test every possible correlation. Nevertheless, our numerous Pearson and Point biserial correlation tests never identified a significant correlation between the participants' personal information and the obtained results. The only few significant correlations are between the questionnaire answers themselves representing the "coherent" rating of the participants. That is, participants tend to rate similarly similar situations. For instance, the better a participant rates the interaction in the first scenario, the better they tends to rate the interaction in the third scenario, in which execution is similar.

6.5 Study results

In this section, we analyze the results of the study. First, we share some technical comments regarding the experiment. After, we analyze the results obtained from the execution logs. Then, we discuss the answers to the questionnaire. Finally, we discuss the participant's comments regarding the experiment. Note that all the numeric results of the study are given in appendix A, including questionnaire answers, execution metrics, comments, and scenario preferences.

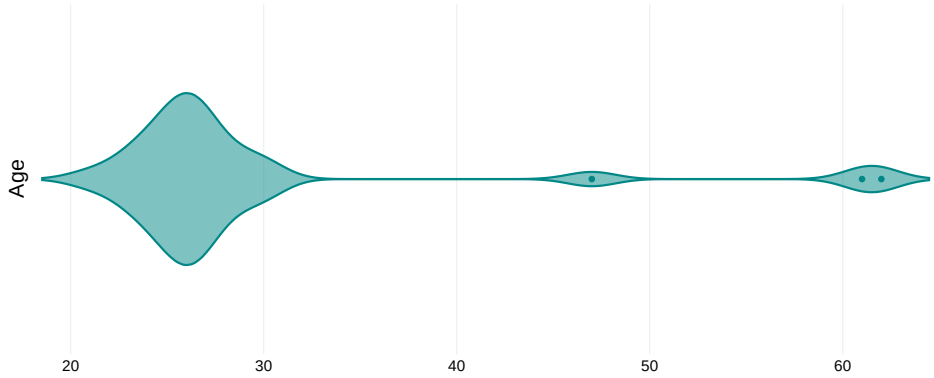


Figure 6.4: Participants' age.

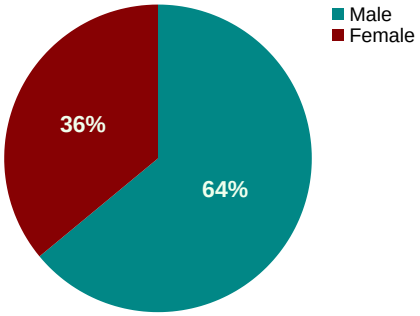


Figure 6.5: Participants' gender.

6.5.1 Technical comments

Numerous scenarios were executed in the simulator to conduct this study. More precisely, 150 scenarios were executed, and a total of 1914 steps were executed. It is interesting to share a few technical comments about how those executions.

To begin with, very few technical issues or crashes occurred during the study. About 2 or 3 crashes were due to a failure in the HMI that had happened when participants clicked at a very specific instant. I was not able to identify the origin of the issue, but this only happened a few times, considering that 1048 human actions were performed during the whole study. This means that 0.29% of the human action failed. Then, about 4 to 6 crashes occurred due to a failure of the robot arm motion controller. The arm motions were planned successfully, but the controller failed to execute the planned trajectory in the simulator, which led to a crash of the controller and the robot freezing. This kind of failure was specific to the MoveIt framework, and I could not find a solution to it. But again, those issues were quite rare considering that the robot performed a total of 1586 actions during the study. This means that 0.38% of the robot action failed. Overall, less than ten scenarios crashed during the study, i.e., less than 6% of failures. In practice, recovering from a crash was quite easy and fast. After a brief intervention of less than $30s$, the participants were able to start again from the scenario that crashed. Sometimes, this implies that the participants repeat a large part of the crashed scenario, which affects the participant's impression (less novelty effect). However, none of these crashes significantly changed the participant's actions when repeating such a scenario.

On the other hand, it is worth mentioning and discussing the durations of the different

processes run by the robot. At every step, the robot has to decide which action to perform, move its head, plan its arm motion, and move its arm. First, the decision time of the robot is negligible because it is given by the policy computed previously by our planning approach. Before every step, the robot identifies the current state. Given a state, the policy dictates which action the robot should perform, including if the human action must be identified first. Since all this is precomputed, the decision time is negligible. Head motions are also not demanding, and their execution occurs in parallel with the other robot processes. Hence, they can be neglected. However, planning the robot's arm motions is heavy computing and takes about $0.56s \pm 0.28s$. Notice that the standard deviation (SD) is quite high, about half of the average value. Indeed, the motion planning is based on algorithms using randomized exploration, making the solving time random, sometimes beginning very fast ($min \approxeq 0.001s$) and sometimes quite slow ($max = 5.37s$). Nevertheless, we were able to plan the arm motion online. Eventually, the robot arm motion durations are about $4.09s$ (max=$9.09s$, min=$1.46s$) and will be discussed more precisely below

Additionally, since the task is quite simplistic, repetitive, and deterministic, one could say that we could have pre-computed the robot arm motions in order to lighten the execution and avoid technical problems linked to motion planners. First, we insist on the fact that the arm motion failures were due to execution failure, not planning. Thus, it is unclear if pre-computed trajectories would have helped regarding those failures. Doing so would certainly make the simulator less demanding in terms of computation power. But here, we wanted to keep a generic simulator able to conduct any other task. Thus, movements could not be pre-computed.

6.5.2 Statistical assumptions

Our data are close to following a normal distribution (checked using Kolmogorov-Smirnov, Shapiro-Wilk, and Anderson-Darling tests). Thus, parametric tests can be applied, and we used both paired t-tests and Analysis Of the VAriance (ANOVA) with repeated measures to analyze the collected data, more precisely, to identify significant differences between different group of measures. In the last case, Bonferroni Post-hoc-Tests are performed to identify exactly which groups are significantly different from others.

It has been commonly assumed that a statistical test demonstrates a significant difference if a p-value lower than 0.05 is obtained. However, obtaining a value lower than 0.001 is desired. To make the p-values more legible, the following standard notation is commonly used and will be used below:

$$
\begin{aligned}
p > 0.05 &\Rightarrow ns \text{ (non significant)} \\
p \leq 0.05 &\Rightarrow * \text{ (significant)} \\
p \leq 0.01 &\Rightarrow ** \text{ (very significant)} \\
p \leq 0.001 &\Rightarrow *** \text{ (highly significant)}
\end{aligned}
$$

Additionally, the value of a metric x will often be given in the following format depicting the average value M and the associated standard deviation (SD) σ: $x = M \pm \sigma$.

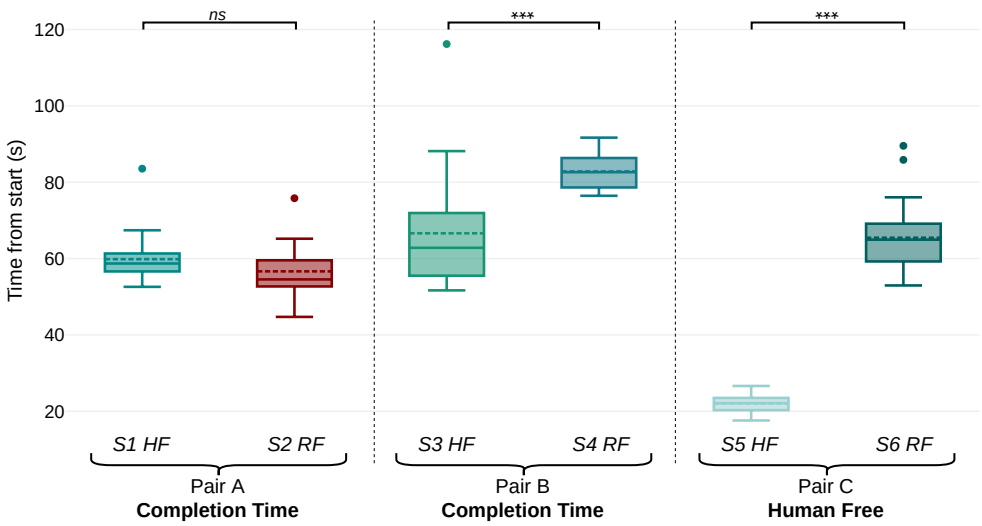


Figure 6.6: Human preference satisfaction. According to the scenarios, it corresponds either to completing the task as fast as possible (Pair A and B) or being free as soon as possible (Pair C). Using t-tests for paired samples, we can identify in pairs B and C that the criteria of preferences are significantly better satisfied. In pair A, the difference is not significant, but the completion time is slightly shorter when using RF.

6.5.3 From execution logs

This section is focused on analyzing the results obtained through the execution logs saved after each scenario.

Preferences satisfaction (task completion time + time to be freed)

In this study, the human preferences consist of either finishing the collaborative task as soon as possible or being free as soon as possible while letting the robot finish alone. Thus, to evaluate how the human preferences were satisfied, we can measure the time to complete the task in the first case and the time after which the human can leave in the second case.

Figure 6.6 depicts through box plots the corresponding relevant metric for each pair of scenarios to evaluate the human preferences' satisfaction. We used t-tests for paired samples for pairwise comparison. For each pair, the tests for normal distribution suggest that the data does not significantly deviate from normality, and thus, parametric tests such as t-tests can be conducted.

In Pair A, in addition to completing the stack, the human wants it to be completed as soon as possible. The robot has a correct estimation of human preferences. The completion time using HF and RF are shown. The completion times in S1 and S2 are roughly similar with the respective values: $59.84s \pm 5.83s$ and $56.64s \pm 6.46s$. The completion time of Scenario 1 is higher than Scenario 2. However, a t-test for paired samples showed that this difference was not statistically significant ($p = 0.055$), and there was a small effect ($d = 0.4$) according to Cohen's d [Cohen 1988] (small effect = 0.2, medium effect = 0.5, large effect = 0.8). Thus, the RF regime allowed participants to solve the task slightly faster than

the HF regime. Therefore, human preferences were satisfied slightly better than using the HF regime. In both scenarios, the collaboration goes smoothly, and the task is achieved without trouble.

In Pair B, the human still wants the stack to be completed as fast as possible. However, the robot has an erroneous and adversarial estimation of their preferences. This time, the HF regime in S3 had lower values ($66.62s \pm 14.87s$) than the RF regime in S4 ($82.82s \pm 4.42s$). This difference is statistically significant ($p < 0.001$), and there was a large effect ($d = 1.07$). This indicates that in S4, the participants' preferences were significantly less satisfied than in S3. Indeed, in this pair, the robot erroneously thinks that the human wants to minimize their effort. Thus, the robot ends up trying to "*steal*" cubes from the human to prevent them from acting, thus minimizing their effort. With RF, the human has no choice and cannot act most of the time, leading to a high completion time with a low SD due to the restricted human choices, which leads to very similar executions. With HF, the robot always acts compliantly in parallel right after the human. Hence, the human is able to pick the cubes they want and that the robot wants to pick, *forcing* the robot to adapt and pick other cubes. This eventually leads to executions close to S1. However, if the human decides not to pick a cube, the robot will likely pick it, preventing the participant from acting. A few participants were distracted and let the robot pick the common cubes, leading to significantly different executions than the non-distracted participants, which explains the high SD. Comments about the feelings of the participants in each of these scenarios are given in the next subsection using the answers to the questionnaire. Overall, S4 was perceived as frustrating, and S3 was perceived similarly to S1 and S2.

In Pair C, the human prefers to be freed as soon as possible. Hence, we measured the time after which the human is not required to finish the task, i.e., the time after which the robot can finish the task alone. Scenario 5 (HF) allowed the human to be free earlier ($22s \pm 2.35s$) than Scenario 6 (RF) ($65.45s \pm 9.08s$). This difference is statistically significant ($p < 0.001$), and there was a very large effect ($d = 4.43$). This indicates that the HF regime allowed the participants to satisfy their preferences significantly better than the RF regime. Here, the erroneous estimation of human preferences makes the robot try to place its pink bar first, which implies that the human should place their own at the top of the stack as the last cube. Such a plan forces the human to stay until the end of the task which is in direct contradiction with the actual human preferences. Hence, after placing the yellow and red cubes concurrently, both agents tend to pick their pink bar. At this point, in S5 (HF), the robot waits for the human's decision, and the human can place their bar and free themselves from the task. The robot compliantly drops its pink bar before finishing the stack alone. However, in S6 (RF), the robot does not wait for the human decision and places its pink bar before the human can do anything, forcing them to stay until the end to place the pink bar. As a result, the S6 values are significantly higher than S5. Moreover, the participants had various reactions to the frustrating robot action of placing the pink bar before them. Some remained passive until the end while holding their bar, while some others dropped it to help the robot, aiming to place the bar as fast as possible anyway. These various reactions led to various executions, explaining the high SD in S6.

Overall, RF tends to slightly better satisfy human preferences only when the estimation is correct (Pair A), yet the difference was not significant compared to the HF. On the other hand, when the estimation is erroneous, HF satisfies human preferences significantly better than RF due to how compliant the robot is when using HF. This indicates that using our model of execution instead of a simplistic baseline (RF) is beneficial for collaboration in terms of satisfying human preferences.

Ratio human optimally

The participants were given in every scenario an objective to satisfy, to consider as their own preferences regarding the task, and that should guide their behavior. However, in practice, the explicit actions to conduct were not given, and the participants were free to act as they would. Naturally, not all participants behaved in the same way. There were differences in the decision time of each, as well as in the action decisions, leading to different execution traces. Since different execution traces significantly influence the timeline metrics, it is worth discussing how the participants behaved.

First, table 6.3 depicts the number of different execution traces per scenario and overall. There were 45 different execution when considering all scenarios, which can appear quite low compared to the 6839430 possible plans. This also means that our exploration covers enough possibilities for this task. Additionally, it is worth noticing the high number of different plans in S6 and the low number in S1. In S6, the robot acts quite frustratingly, leading to various reactions from the participants. On the other hand, it seems that S1 was quite clear since participants performed only four different sequences of actions. It is also worth mentioning that since there are only two different human objectives or preferences, we would expect only two different optimal traces, one satisfying each objective. All other 43 other obtained traces are due either to "wrong" robot decisions or suboptimal human decisions.

	Total	S1	S2	S3	S4	S5	S6
Number of different executed plan	45	4	9	10	6	7	16

Table 6.3: Number of different plans executed in each scenario and overall.

In the same manner as for the robot, an optimal human policy is generated for each scenario (considering the actual preferences given to the participant). Hence, it is possible to check at each step if the participant performed the optimal action or not and, thus, compute an optimal ratio, which is the number of optimal human actions performed divided by the total number of human actions performed. This helps us analyze the results and explain some outlier values.

Though there are no significant differences between the different scenarios, some scenarios still have a lower average optimal ratio and high SD, meaning that participants tend to have more varied behaviors in these specific scenarios. The average number of human actions per scenario is about 7, from 2 to 10.

	S1	S2	S3	S4	S5	S6
Mean	*95.52*	*95.4*	*92.04*	*98.03*	*96.74*	*87.09*
Std. Deviation	*7.56*	*7.78*	*8.28*	*4.58*	*7.65*	*13.48*
Minimum	*71.43*	*71.43*	*78.57*	*81.25*	*75*	*61.54*
Maximum	*100*	*100*	*100*	*100*	*100*	*100*

Table 6.4: Optimal human action ratio per scenario

As depicted in the table 6.4, S6 has the lowest average optimal ratio and the highest SD. In this scenario, the robot places its own pink bar even though the human holds one already, preventing the human from placing it and forcing them to drop it back on the table. This surprising behavior seems to cause frustration and confusion, which led to various human decisions and actions, and more likely to deviate from the optimal course

of action. In practice, many participants get confused and are passive during several steps after the frustrating robot action. Some even remain passive almost for the whole task, waiting for the robot to stack the cubes alone until the human pink bar has to be placed. This diversity in the participants' reaction is reflected in the high SD of S6.

The low SD of S4 is also noticeable. Indeed, here, the robot tends to steal the cube from the human's reach. This behavior prevents participants from acting and, thus, from making decisions. As a result, fewer decisions are taken by the human in this scenario which results in less possible deviation from the optimal course of action.

Decision time

Participants' decision time fluctuates a lot, especially with HF. Indeed, at every step, the HF robot waits for a defined amount of time to observe the human decision and acts accordingly. Any human visual signal received interrupts this timer. This amount of time will be referred to as the HF Timeout because after it is reached, the robot considers the human to be passive. This timeout was initially set to $3s$ with the hypothesis that it should be quite small to allow fluent interaction. With a precise action in mind, humans act first and fast. Otherwise, the robot fluently takes the lead and acts first. However, during the preliminary tests, the participants felt in a rush and oppressed by this relatively low timeout. Indeed, when they did not have a precise action to perform when the step started, they did not have the time to think properly and tended to be rushed by the timer, progressing towards the timeout. Hence, we decided to increase the timeout from $3s$ to $4s$, which made it feel way more comfortable.

One could think about comparing the total (sum, cumulative) decision time over each scenario. However, since different human actions can lead to various number of steps, this is not representative.

We compare the average human decision times, measured similarly with HF and RF, and as follows. After one participant finished one scenario, we measured their decision time on each step. To do so, we first consider the time when the step begins for each step, which is signaled with text, a gaze, and a sound from the robot. Then, we consider the time when the human sends a signal by either starting an action or by waving their hand. The duration between these two times is considered as the human decision time. Note that if the human remains passive (no signal until) for a step, no decision time is computed for this specific step. Then, we extract the average decision time of the participant on the scenario from all the computed ones, compute the SD, and get the maximum and minimum values.

A one-factor analysis of variance with repeated measures showed that there was a significant difference between the variables, $F = 5.99$, $p =< .001$, with an effect size Eta squared $\eta^2 = 0.2$, which corresponds to a large effect. When doing pairwise comparisons with t-tests, the following results were obtained:

In pair A, S1 (HF) had lower values (0.66 ± 0.41) than S2 (RF) (1.04 ± 0.58). This difference is statistically significant ($p = 0.002$) with a medium effect ($d = 0.68$).

In pair B, S3 (HF) had lower values (0.54 ± 0.51) than S4 (RF) (0.62 ± 0.55). This difference is not statistically significant ($p = 0.551$) with a very small effect ($d = 0.12$).

In pair C, S5 (HF) had lower values (0.56 ± 0.11) than S6 (RF) (0.91 ± 0.46). This difference is statistically significant ($p = 0.001$) with a medium effect ($d = 0.76$).

Considering the defined scenario pairs, the decision time with RF tends to be longer. However, this difference is statically significant only for pair C (S5-S6). This is expected because when the robot places the first pink bar, the human gets confused and takes time to adapt to the situation. On the other hand, this is not reflected in S4 despite the similar confusing robot actions. Indeed, in S4, the robot "steals" cubes from the human reach,

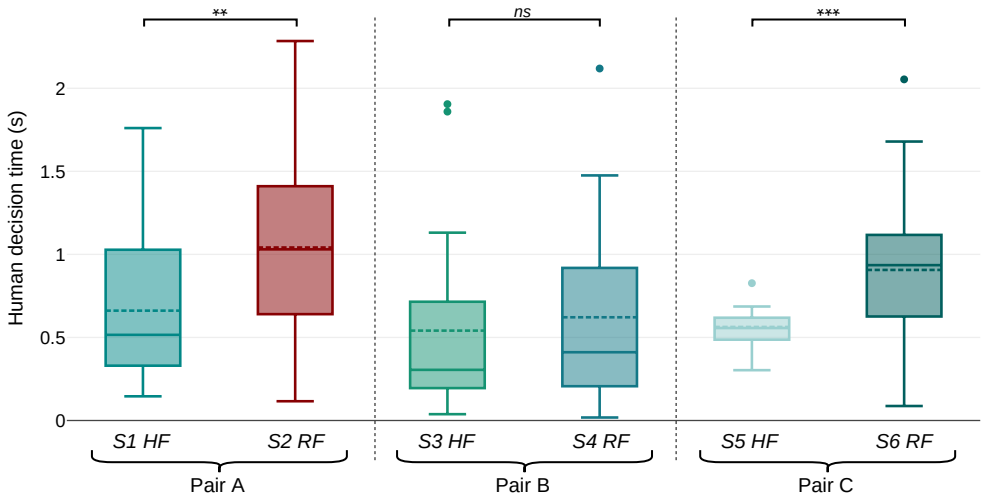


Figure 6.7: Average human decision time in the six scenarios. This decision time tends to be lower when using HF than with RF.

which is confusing. Since this prevents participants from acting, no decision time can be computed.

I think the overall slower human decision time in the RF scenarios is because the human acts after the robot. This way, the human has to pay attention to the scene and the robot's action, which is longer than only looking at the scene, like in HF scenarios.

Overall, the decision time of the participants is an average of about $0.72s \pm 0.49s$.

Agent actions' duration

As depicted in fig. 6.8, the human actions are significantly faster than the robot ones on average. In addition, the duration of the robot's actions tends to fluctuate more than human ones. This can be explained by the difference in motion execution between the avatar and the robot. The human has a simplified motion planner that simply moves the hand at a constant speed and in straight lines to the cubes or the stack. However, the robot uses a real motion planner to move its arm, which is longer than human motion. The motion planning process does not always find the same solutions, nor in the same amount of time. Meaning that both the motion planning duration and the motion execution duration can fluctuate. Here, only the motion execution duration is considered in this metric. Note that to avoid having too much difference between the human and robot action durations, collisions with the dynamic objects are not considered in the robot motion planner, nor the objects' orientation. Hence, the robot can pick or place cubes from any angle and pass through the other cubes. When placing a cube, its orientation is corrected. Collisions with the table were kept, preventing the robot from picking cubes from below.

Overall scenarios and steps, mean = $3.27s$, the maximum human action duration is $4.63s$, and the minimum is $2.55s$. For the robot, mean=$4.09s$, the maximum action duration is 9.09, and the minimum duration is 1.46.

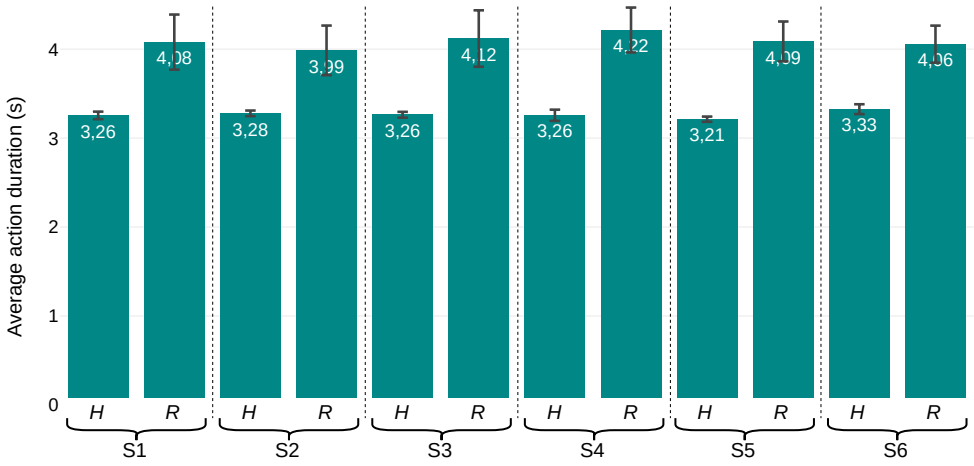


Figure 6.8: Average action duration of the human and robot agents over the six scenarios, with standard deviations. Compared to the human action durations, the robot ones tend to be longer and have various durations.

6.5.4 From questionnaires

This section is focused on providing the results obtained by analyzing the answers to the questionnaires filled out by the participants after each scenario.

To help the reader understand the following plots, we list here the items of the questionnaire from the table 6.6 with their associated numeric ID in table 6.5. These IDs will be used in many plots on the x-axis to analyze the questionnaire's answers.

Robot perception	Interaction	Collaboration	Acting
1 Responsive	4 Positive	7 Adaptive	10 Appropriate
2 Competent	5 Simple	8 Useful	11 Accommodating
3 Intelligent	6 Clear	9 Efficient	12 Predictable

Table 6.5: Questionnaire items with their associated IDs.

6.5.4.1 Overall analysis

We start by commenting on the overall questionnaire's results using relevant average values and standard deviations before having a deeper statistical analysis in the next subsection.

Figure 6.9 depicts the answers obtained for each question of the questionnaire w.r.t. each scenario. This figure provides a very visual overall summary of the study. On the top part, for each of the 12 questions on the x-axis, the average answers obtained are plotted for each scenario, 7 being the maximal or best value and 1 being the minimal or worst value. We can see that four scenarios obtained quite similar high answers, whereas scenarios S4 and S6 have noticeably worse answers. Those scenarios are the Robot-First scenarios of pairs B and C, where the robot has an erroneous, and even adversarial, estimation of human

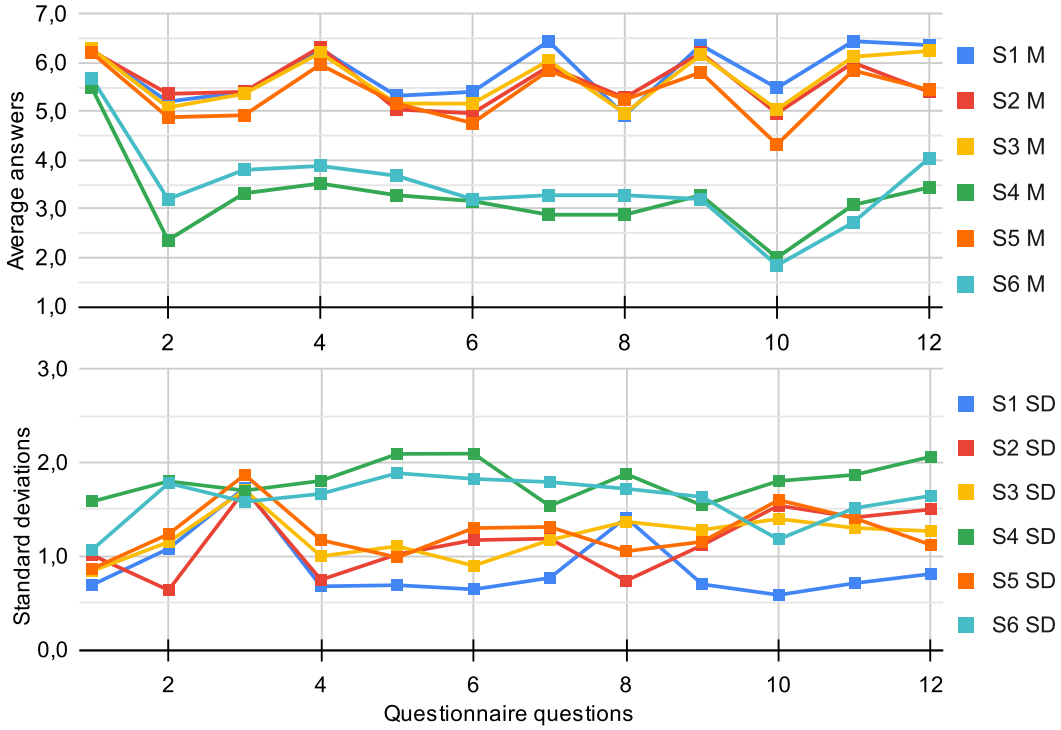


Figure 6.9: W.r.t. each scenario, average answers (M, top) and standard deviations (SD, bottom) obtained for each question of the questionnaire.

preferences. Note that question 1 (Q1), which evaluates the reactivity of the robot, is the only question whose answers are relatively high in every scenario. Considering the answers to all questions other than Q1, S4 and S6 seem to deviate significantly from the other scenarios, which can be analyzed as follows: First, it means that with a correct estimation, both HF and RF regimes are roughly perceived similarly. Second, an erroneous estimation does not seem to induce lower answers, and thus, despite the wrong estimation, the robot in S3 and S5 is roughly perceived similarly to the one in S1 with a correct estimation. On the other hand, when using RF, an erroneous estimation seems to have a significant detrimental impact on how the robot is perceived by the participants. All these preliminary conclusions will be confirmed in the statistical analysis below.

It is also worth commenting on the standard deviations obtained, depicted in the bottom part of figure 6.9. The standard deviations depend a lot on the scenarios and go from 0.6 up to 2.1. There are two noticeable facts to comment on. First, question 3, evaluating how intelligent the robot is perceived, is the only question with a relatively high SD for every scenario. Participants had various definitions of "intelligence", which led to a wide range of answers. Some participants evaluated the robot's intelligence on its choices of actions, and thus, fluctuated depending on the scenario. Some others evaluated the intelligence of the robot on other criteria independent of the robot's decisions. Thus, they would rather indicate that the robot was always intelligent (or not) over all scenarios. Additionally, we can see that the SD of S4 and S6 seem to be higher than the other scenarios.

In contrast with the previous figure, Figure 6.10 shows the answers obtained for each question w.r.t. to each execution regime. Indeed, the HF average answers and standard deviations, shown in blue, correspond to the union of the scenario's answers using the Human-First regime, i.e., $S1 \cup S3 \cup S5$. Similarly, the RF values, shown in red, correspond

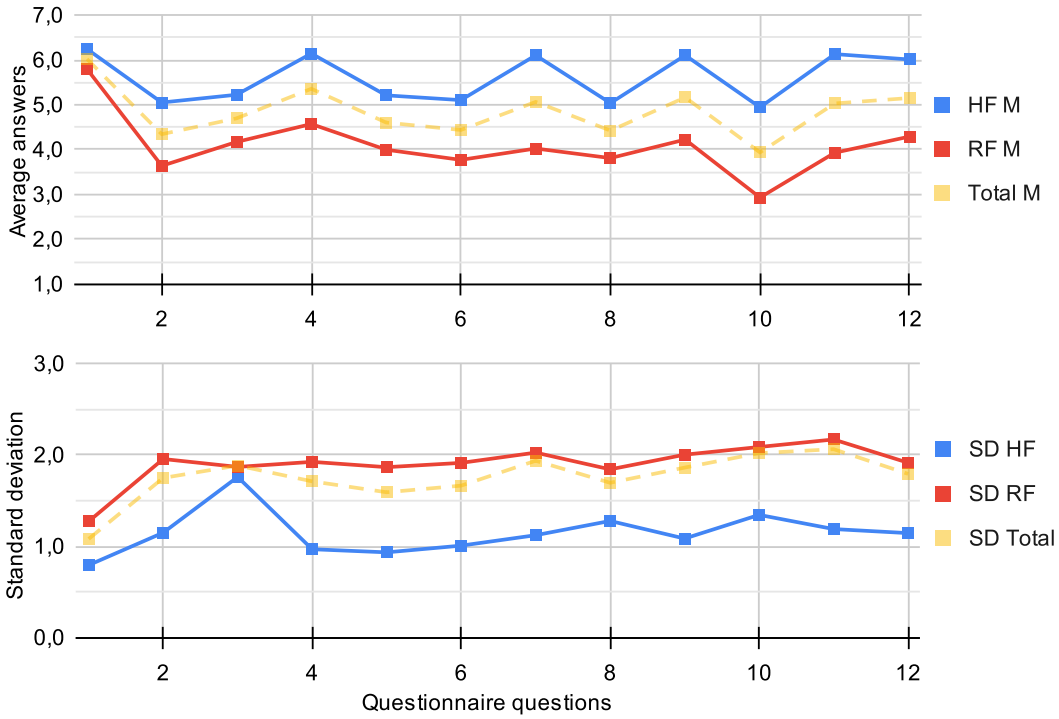


Figure 6.10: W.r.t. each regime of execution, average answers (M, top) and standard deviations (SD, bottom) obtained for each question of the questionnaire.

to $S2 \cup S4 \cup S6$ where the Robot-First regime is used. Additionally, the combined results for all scenarios are shown in light yellow. The results shown here can be deduced from the previous figure since we already commented on all scenarios. Nevertheless, this new figure highlights more visually the difference between the HF and RF regimes.

The first noticeable fact is that when using the HF regime, the average answers to each question are better than when using the RF regime. Again, the only question where the answers are roughly the same regardless of the regime is when questioning the reactivity of the robot. HF's average answers are relatively high for all questions which indicates that the collaborations with HF seem to be appreciated. On the other hand, RF's answers are average (around 4), indicating that collaborating with RF seems less appreciated.

Concerning the standard deviations, it is also noticeable that the answers concerning the RF regime have higher SD. This indicates that there was a wider range of answers from the participants when collaborating with the RF regime. This means that participants tend to be less certain about their answers when evaluating RF than with HF. This can be expected because when facing the HF regime, the collaboration is quite positive overall. Therefore, participants concentrated their answers on the higher part of the scales. However, the frustrating robot actions due to the RF regime degraded the collaboration, and participants had to evaluate how bad this degradation was. Some participants were more emotionally affected than others by the robot's actions. Hence, it led to a wide range of answers. Another interesting fact is that regardless of the regime, question 3 has a high SD. This question evaluates the *intelligence* of the robot. Participants had various definitions of intelligence, which is reflected in this high SD. Indeed, some participants evaluated the robot as unintelligent because of its nature and, thus, regardless of its actions or the scenario. Others perceived less intelligence when the robot performed frustrating actions. Also, we

can see that the SD regarding the reactivity of the robot is quite close and low for both regimes. This is a consequence of the very high average values regarding Q1 for both regimes.

Eventually, I would like to insist on the average answers concerning the RF regime. Even if these answers are mediocre, it is important to note that they are not very low. In comparison, consider another robot behaving completely erratically. This robot would randomly pick and place cubes around itself. Consequently, the robot would neither help humans nor solve the task. On the contrary, it might only disturb the human trying to achieve the task. The robot could make the task impossible for the human by picking a relevant cube and never placing it or removing already well-placed cubes from the stack. Here, during one step, the RF regime forces humans to comply with the robot's decisions, which can be frustrating. However, the robot takes human action into account and adapts its actions accordingly for the next step. This is thanks to our planning approach, which is common to both regimes and explores every possible human action to generate the robot policy. Thus, we can say that our planning approach seems to benefit the collaboration and interaction between the human and the robot.

6.5.4.2 ANOVA Analysis

So far, we only conducted a preliminary analysis of the questionnaire's answers using only the average values and standard deviations. Now, a statistical analysis must be conducted to confirm the preliminary comments of the previous subsection. For each question, and thus each item of the questionnaire, we performed an analysis of the variance with repeated measures. Each analysis led to p-values $<= 0.001$, indicating that there is a significant difference between the six scenarios. To evaluate the strength of this significant difference, the effect size Eta squared η^2 has been calculated where the limits are .01 (small effect), .06 (medium effect), and .14 (large effect). However, ANOVA tests can only indicate if there is a significant difference between N-samples, but it is only of interest to identify between which exact group that difference exists. In the Bonferroni post-hoc test in an ANOVA with repeated measures, multiple t-tests are calculated for dependent samples. However, the problem with multiple testing is that the so-called alpha error (the false rejection of the null hypothesis) increases with the number of tests. To counter this, the Bonferroni post-hoc test calculates the obtained p-values times the number of tests. The obtained p-values indicate in a pairwise manner between which samples the significant difference exists. The results from the ANOVA and Bonferroni post-hoc test are shown in table 6.6.

As suggested by the preliminary analysis, the robot's reactivity is the item with the lowest difference. The ANOVA indicates that answers regarding the reactivity are significantly different with a large effect over the six scenarios. However, compared to other items, the effect size η^2 is quite low, indicating that this difference is less significant than for other items. Also, the Bonferroni post-hoc test was not able to identify where exactly the difference exists, which means again that this difference is not very significant in the end.

Besides reactivity, all other items have significant differences according to the scenario, which can be exploited using the Bonferroni post-hoc test. Indeed, the pairwise comparisons indicate the existence of major significant differences for every question in the following pairs: S1-S4, S1-S6, S2-S4, S2-S6, S3-S4, S3-S6, S4-S5, and S5-S6 (in bold in the table). Having in mind that S4 and S6 are the two scenarios using Robot-First with erroneous estimations, we can see that erroneous estimation with RF systematically leads to significant differences compared to all other scenarios using HF or RF with a correct estimation (S4-S1, S4-S2, S4-S3, S4-S5, and S6-S1, S6-S2, S6-S3, S6-S5). This is a clear indicator that the

	ANOVA		Bonferroni Post hoc tests													
	p	η^2	1-2	1-3	**1-4**	1-5	**1-6**	2-3	**2-4**	2-5	**2-6**	**3-4**	3-5	**3-6**	**4-5**	4-6	**5-6**
Responsive	***	*0.16*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*	*ns*
Competent	***	*0.49*	*ns*	*ns*	***	*ns*	***	*ns*	***	*ns*	***	***	*ns*	**	***	*ns*	*
Intelligent	***	*0.33*	*ns*	**	*ns*	*ns*	*	*ns*	**	*ns*	**	**	*ns*	***	*	*ns*	*
Positive	***	***0.65***	*ns*	*ns*	***	*ns*	***	*ns*	***	*ns*	***	***	*ns*	***	***	*ns*	***
Simple	***	*0.34*	*ns*	*ns*	**	*ns*	**	*ns*	**	*ns*	*ns*	*	*ns*	*	**	*ns*	*
Clear	***	*0.41*	*ns*	*ns*	***	*ns*	***	*ns*	**	*ns*	***	***	*ns*	***	*	*ns*	**
Adaptive	***	***0.62***	*ns*	*ns*	***	*ns*	***	*ns*	***	*ns*	***	***	*ns*	***	***	*ns*	***
Useful	***	*0.43*	*ns*	*ns*	***	*ns*	*	*ns*	***	*ns*	***	***	*ns*	**	***	*ns*	***
Efficient	***	***0.62***	*ns*	*ns*	***	*ns*	***	*ns*	***	*ns*	***	***	*ns*	***	***	*ns*	***
Appropriate	***	***0.62***	*ns*	*ns*	***	**	***	*ns*	***	*ns*	***	***	*ns*	***	***	*ns*	***
Accommodating	***	***0.68***	*ns*	*ns*	***	*ns*	***	*ns*	***	*ns*	***	***	*ns*	***	***	*ns*	***
Predictable	***	*0.45*	*ns*	*ns*	***	**	***	*ns*	*	*ns*	*	***	*	***	**	*ns*	**

Table 6.6: Significant differences in the questionnaire answers between the different scenarios. For each item of the questionnaire are shown the overall p-value and η^2 (effect size) obtained after an ANOVA. Additionally, the p values obtained after conducting Bonferroni Post-hoc-Tests are shown to identify in a pair-wise manner which scenarios were significantly different from others. As depicted, scenarios S4 and S6 are distinguishable from the others, and their evaluation is significantly different on all the measured aspects (expect reactivity).

RF regime is very sensitive to the estimation of human preferences. Thus, an erroneous estimation is significantly detrimental to the collaboration and the overall interaction.

Only a few other significant differences exist in S1-S5 and S3-S5. Indeed, in S5, the robot's actions were perceived as more or less significantly less appropriate ($p = 0.004$) and predictable ($p = 0.004$) than in S1, slightly less predictable ($p = 0.017$) than in S3. Indeed, in S5, the HF robot surprisingly picks up its pink bar while the human picks up its own whereas the human wants to place their bar to be freed from the task. Using HF allows the human to place their bar anyway, but the robot actions were therefore perceived as less predictable and appropriate than in S1 or S3. We can also notice that despite S3 having an erroneous estimation in contrast to S1, there is no significant difference in answers for each question between S1 and S3. This indicates that using the HF regime allows the robot to be way more robust to erroneous estimation than RF. However, the few existing significant differences among the HF scenarios indicate that an erroneous estimation is still noticeable and can still have a detrimental influence. Thus, the robot cannot fully rely on being reactive and compliant to human actions. Estimating human preferences accurately to plan the robot's actions appropriately is mandatory for optimal collaboration.

Looking at the eta squared η^2 of every question, one can notice that we can group the items into four groups.

1. $\eta^2 = 0.16$: First, the Reactivity item is alone with the lowest effect size. This group does not help to distinguish the regimes.
2. $\eta^2 \simeq 0.33$: Secondly, we can state that when using RF, the robot was perceived as slightly less intelligent, and the interaction as slightly less simple than when using HF.
3. $\eta^2 \simeq 0.45$: Then, due to the moderate effect size, we can state that when using RF, the robot was perceived as moderately less competent, the interaction as moderately less clear, the collaboration as moderately less useful, and the robot actions as moderately less predictable.
4. $\eta^2 \simeq 0.64$: Eventually, with a high effect size, when using the RF regime, the interaction was perceived as significantly less positive, the collaboration as significantly less adaptive and efficient, and the robot actions as significantly less appropriate and less accommodation. Since the five items from the last group are the ones with the highest effect size, they can be seen as the main characteristics differentiating the HF from the RF regimes and thus are highlighted in the table.

6.5.5 From comments

At the end of the experiment, every participant was asked two questions, gathering their general impressions. First, they were asked to comment on the overall experiment and robot interaction they just had. Secondly, participants were asked to indicate which scenario they preferred the most and the least.

Participants' comments concern several aspects of the experiment and are worth discussing. Overall, the comments confirm the outcome of the statistical analysis, and they also provide some feedback about the overall experiment protocol and conditions, especially regarding the simulator itself. The comments are discussed per categories below.

6.5.5.1 Simulation

First, most of the participants found the experiment and the simulation to be a good experience. They felt committed and active during the different scenarios. The simulation

has been described several times as clear, simple, pleasant, intuitive, captivating, and funny. Some participants mentioned that it was like a video game and enjoyed it. These comments suggest that collaborating with a simulated robot has been appreciated, and they raise the question of whether the participant had felt the same way in an experiment with a real robot.

6.5.5.2 Display

Some participants think that the simulation display had too much information (goal + scene + text prompt), and a few had trouble reading the text prompt. Yet, the robot can be seen well. Indeed, the text prompts could be a bit fast and written white on black, which can be disturbing when not used to. However, I believe this did not affect much the executions, maybe slightly the decision times.

6.5.5.3 General

A few participants would have appreciated the robot giving instructions and guidance regarding the actions to perform. This is linked to another comment saying that using the Robot-Fist regime with correct estimation felt better because it makes the task simpler. If the human trusts the robot, it can be appreciated to let the robot compute the optimal plan and just follow the robot's instructions, lowering the cognitive load of the human. Indeed, some participants consider that they made mistakes and that they could have acted better in some scenarios.

6.5.5.4 Steps

The step synchronization was not appreciated by everyone. Some participants found this kind of synchronization useful as it structured the collaboration. However, many others found this a bit confusing at first and frustrating because they had to wait for the robot's actions to be done before being able to act again.

6.5.5.5 Task

The task was found to be clear and quite simple. One participant said that they felt significant emotions such as satisfaction and frustration and that if the task was less abstract and more real, these emotions would have been enhanced. Moreover, another participant said that in such simple tasks, humans think they know better how to solve the task than the robot. Thus, the robot should follow human decisions in such cases, with a hierarchy relation. A few participants also mentioned that performing the last action, i.e., placing the last cube, is very satisfying. These people liked the robot adapting to allow them to do so. The fact that both agents must perform actions to solve the task makes the collaboration relevant and useful.

6.5.5.6 Action

Many participants said that not being able to pick cubes in advance is not natural and, at first, it is confusing, frustrating, and a bit complicated. Yet, they also said that they got used to it quite fast. One also said that they felt obliged to act at every step. Indeed, a majority of the participants were signaling their passivity to the robot even when they were not able to act. About the movements, one participant stated that the actions were stiff and rigid, in contrast to being able to drag and drop the cubes thanks to the physics

simulation. Additionally, the lack of collision with the cubes felt a bit unrealistic but not very confusing. On the other hand, another participant said that the robot's movements seem real. This is probably because we used an online motion planner to move the robot arm. Thus, the movements were not always optimal.

6.5.5.7 Objective

A few participants said that the objective of "trying to be free early" is a bit frustrating since they would like to keep acting, even if not necessary. It was hard for them to consider this objective as their personal preference and, thus, to act accordingly. Additionally, one participant mentioned that Scenario 5 creates double satisfaction: being free early (preferences) and fulfilling the task.

6.5.5.8 Regimes

One participant said that they did not see much difference between the two execution regimes, HF and RF. The same participant could not indicate which scenario they preferred at the end. Moreover, some participants also indicated that the difference between HF and RF was unclear at first. However, once used to the task and the scenarios, the difference becomes clearer, and before the end of the experiment, most of the participants had a clear idea of each regime and even apprehended the RF one.

6.5.5.9 Being in control

Participants indicated that when using HF, they felt in control and free to decide which action they performed and that the robot was adapting to their decisions and actions, which was appreciated. In contrast, when using RF, participants did not feel in control and were forced to adapt to the robot's decisions. Even when the robot's decisions are good, the lack of control is uncomfortable. One participant stated that they disliked when the robot took the initiative because the robot could be wrong.

6.5.5.10 Human-First (HF) regime

Most of the participants enjoyed the HF regime and stated that they were able to fulfill their objective with it. Some comments qualify the HF regime as slower than RF and sometimes inconsistent. The latter is mostly referring to the robot picking up the pink bar in S5. However, especially when used to it, HF has been qualified as smooth, efficient, interesting, predictable, and less frustrating than RF. Several participants mentioned that they enjoyed being able to predict the robot's behavior, proving that having predictable behavior is crucial for a seamless collaboration. It has been mentioned that HF makes less wrong choices than RF. Moreover, some participants said that compared to the RF regime with a correct estimation, HF is less efficient. However, in pairs B and C, HF is more efficient than RF.

6.5.5.11 Robot-First (RF) regime

RF, bad, not in control, bad choices: having to drop bar + using common resources first

Every participant had an overall negative opinion regarding the RF regime. The latter has been qualified as very frustrating, confusing, constraining, unpredictable, inefficient, and even adversarial. A significant number of participants stated that finishing the task quickly with RF could be great, fast, efficient, and less cognitively demanding despite the

lack of control. Also, a few participants noticed that even if, during a specific step, the human is forced to comply with the robot's actions, the robot takes into account the human action and adapts its behavior in the next step. However, it has been said that RF does not consider the human's objective or preferences. Participants really disliked when the robot forced them to drop a cube back on the table (pink bar in S6) and when the robot picked cubes in the middle zone instead of its own zone. The latter was perceived as the robot stealing the cubes from the human. Due to those frustrating robot actions, the RF regime was putting the participants in an adversarial setup, and the robots were explicitly qualified as "enemies". Some participants said that they were more focused on preventing the robot's mistakes than on the actual task.

6.6 Discussion

There are several elements to discuss in this study, including several participants' comments.

First, we decided to conduct the simulation study instead of implementing the system on a real robot. The main reason for this choice is that using a real robot would have required significant additional work. We could not physically force the human to synchronize with the robot according to our step-based model. Therefore, many executions could have failed, and thus, the study would have been longer to conduct. Real robot motions are likely to be slower than our simulated ones, which could also result in bias. An in-between solution could be to use Virtual Reality (VR) to make the participant more immersed.

Our study focused on the robot decisional aspect of collaboration. For this reason, we simplified a few elements of the simulation, such as collisions and physics. Despite being noticeably less realistic than real life, participants seem to find it adequate.

The text prompts raised several questions about which information to show, when, and how. A few participants stated that it was sometimes hard to read the prompts. Further study on these prompts would be necessary.

Due to our limited number of participants, we had to focus on a single collaborative task in order to obtain significant results. Asking participants to solve various tasks would have been too cognitively demanding. Therefore, additional results from another collaborative task would strengthen our results.

Finally, using another baseline would also benefit our study. We mentioned a possible baseline where the robot behaves completely erratically. Such a baseline would have highlighted that the RF regime is not so bad and permits always solving the task, and in an efficient way when human preferences are correctly estimated.

6.7 Conclusion

Thanks to this study, we aimed to validate the overall planning approach and the model of execution Human-First, which is critical to our approach.

After statistically analyzing the execution log data as objective metrics and the questionnaire answers and participants' comments as subjective metrics, we can confidently state that this study successfully validates both our planning approach and our model of execution.

Indeed, we have solid proof that the HF regime gives humans control over the execution, which was significantly appreciated. The participants perceived the robot as accommodating, adaptive, and acting appropriately while being predictable. HF also helps to satisfy better human inner preferences, which makes it more robust to erroneous estimations and

thus more enjoyable. On the other hand, we show how the RF regime can be greatly appreciated when estimating human preferences correctly. However, we demonstrate how erroneous estimations strongly harm collaboration and interaction using the RF regime. Hence, the Human-First regime is preferred and allows for achieving smooth, efficient, and positive collaborations. Nevertheless, thanks to our planning approach, we also show that the RF regime always solves the task with humans, and thus, it is always helpful. Additionally, it also always adapts to human action in the next step.

Part II

Social Navigating Agents Simulation

Chapter 7

Challenging Robot Navigation Systems by Simulating Intelligent Human: InHuS

Contents

This chapter describes the InHuS system, addressing decision-making in navigation and challenging robot navigation schemes. This chapter also compares two robot navigation systems using InHuS, proving that our approach effectively challenges robot schemes and allows measuring and comparing human-aware navigation properties.

7.1 Introduction

Significant efforts are being dedicated today toward the development of robots that interact, assist, or work side-by-side with humans. However, people working in the field of Human-Robot Interaction (HRI) face constraining issues while testing and evaluating their systems. Apart from being mandatory to validate mature systems, experimenting using real humans and robots is burdensome: they are slow, hardly repeatable, expensive, etc. Moreover, the system needs to be run extensively for debugging and tuning before it reaches maturity. Doing so with real-life experiments is generally a long and tiresome process where colleagues

in the lab and volunteers spend unproductive hours, if not days, interacting with a robot running a system under debugging. Moreover, such methods require exclusive physical access to the robot and a place to run the tests.

Simulations are well suited for such tasks as they allow working without a real robot or a physical space. Further, they allow multiple tests to run simultaneously and with a time factor greater than in real life. The simulated test environment can be changed very quickly compared to real-life tests. However, simulating realistic human behaviors and interactions is challenging, which could make simulations unreliable. Consequently, HRI researchers face some difficulties such as: "How to test repeatedly and intensively their systems even when they are not sufficiently robust?" and "How to challenge their systems in a large variety of environments and situations?". Therefore, there is a need for an "intelligent artificial human" that would help challenge the robot's interactive and decision-making abilities.

7.1.1 Human simulations in human-aware robot navigation

Being a part of HRI, the field of human-aware social robot navigation inherits all these limitations. One way to simulate an intelligent avatar in this field is to manually control the human avatar in real-time [Echeverria 2012]. This can be done using a variety of devices like a gaming controller, keyboard, or motion capture. Such approaches require a real human operator only focused on controlling the avatar, bringing back some already-mentioned limitations like human fatigue. On the other hand, autonomous human avatars seem to offer an adequate solution to this, but they often lack intelligence and rationality.

Most of the current autonomous avatars available are either scripted or reactive. A scripted avatar executes a series of predefined actions, like following a fixed path, without being reactive to its environment, which limits interactions. Reactive agents use models like social force [Helbing 1995] or optimal reciprocal collision avoidance (ORCA) [Van Den Berg 2011]. These highly scalable systems can simulate groups or even crowds of numerous agents. MengeROS [Aroor 2018] and PedSim_ROS[1] are some examples. Despite their number, the generated agents usually fail in intricate social scenarios. Some recent works like VirtualHome [Puig 2018] and SEAN [Tsoi 2020, Tsoi 2022] discuss simulating human agents to challenge robot systems, but the navigation of the agents in these systems is still based on reactive-only models. The work presented in [Yige 2021] proposes a learning-based method to generate more realistic pedestrian navigation. This ongoing work shows an interesting navigation behavior like waiting and letting the other agent pass embedded in iGibson [Shen 2020] simulator. However, this work is more focused on motion generation than decision-making to solve conflicts.

We propose the Intelligent Human Simuator (InHuS) System to contribute to the lack of intelligent and rational human agents with conflict-resolution skills to challenge the human-aware robot navigation systems. Our contribution includes 1) an intelligent human agent controller, 2) a high-level interface to control the simulated agents, and 3) a Graphical User Interface (GUI) to plot execution data and metrics for evaluating the interaction. Such a system could help people working in the field of human-aware robot navigation to test and debug their schemes. Our system is designed to run, analyze, and evaluate repeatable and long navigation scenarios involving a robot and an autonomous reactive and rational avatar. This work focuses on intricate and narrow scenarios where, in addition to being reactive, rational decisions should be taken in order to solve the conflicts occurring. Note that our contribution is focused on navigation decision-making and not the motion generation part. Throughout this paper, we use the term 'rational' in a meaning close

[1] https://github.com/srl-freiburg/pedsim_ros

to Goal Reasoning [Vattam 2013, Johnson 2018], i.e., the ability of autonomous agents that can dynamically reason about and adjust their goals. It enables the agents to adapt intelligently to changing conditions and unexpected events, allowing them to address a wide variety of complex situations.

7.2 Description

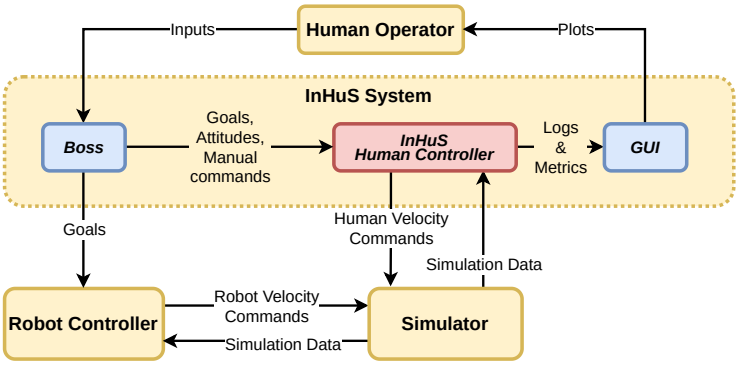


Figure 7.1: The InHuS System interacts with three external systems: the simulator, the robot controller, and a human operator. Our system is separated into three parts: the Boss high-level interface gathering inputs from the human operator, InHuS which is the actual human controller, and a GUI to plot the metrics and other data produced.

The InHuS System[2] works along with a human operator, a chosen simulator, and the challenged robot controller as depicted in Fig 7.1. The system is mainly implemented using ROS. The InHuS System is three-sided. First, the system comes with a high-level interface called Boss that helps to manage the simulated agents. Secondly, the main part is the intelligent human avatar controller itself, called InHuS. Finally, a GUI provides an interactive visualization of the data and metrics computed by InHuS during execution that can help to evaluate interactions. Below, we present some details for each component.

7.2.1 Boss

For the human operator to easily control the simulated agents and run repeatable scenarios, we provide a simple graphical user interface component called Boss. Predefined or manually entered goals can be sent to the human, the robot, or both. Goals are by default considered as "Pose goals" that only require one navigation action to be achieved. However, the human agent (only) can handle "Compound goals" that need a specified sequence of navigation and waiting for actions to be achieved. This type of compound goal is useful to emulate more complex activities. For example, "Make coffee" could be described as a sequence of three actions: nav(coffeeMachine), wait($15s$), nav(myOffice).

The Boss allows defining scenarios with start positions and goals for each agent to repeatedly generate the same situation. Running a scenario consists of first sending each agent to their respective starting position. Then, the corresponding goals are sent to the human and the robot. A delay can be specified while starting the scenario to delay either the robot's or the human's goal. This is very useful to adjust the timing of a specific

[2]https://github.com/AnthonyFavier/InHuS_Social_Navigation

situation or conflict. The Boss can also put an agent in "endless" mode, where the agent continuously gets a new goal from a given list after completing one.

Each navigation action can specify a radius for the "Pose goal", within which a new "Pose goal" is randomly sampled. This mechanism adds randomness to the execution and diversifies the situations encountered, especially in the "endless" mode. Setting the radius to zero disables the randomization and selects the given goal.

All the goals, scenarios, and endless goal sequences are defined using an XML format. Hence, defining new goals or scenarios is straightforward. There is an XML goal file associated with each map/environment. Thus, it is easy to switch between environments since the corresponding goal file is automatically loaded.

7.2.2 InHuS

The macro component InHuS is mainly in charge of controlling the avatar and generating rational behaviors. InHuS itself is made of several components, as depicted in Fig. 7.2. However, three components, namely HumanBehaviorModel, Supervisor, and Geometric-Planner, constitute the major functional part of InHuS. We discuss each of these major components in detail.

7.2.2.1 HumanBehaviorModel:

The HumanBehaviorModel is responsible for most of the rational behavior of the agent. The first role of this component is to manage the goals. Goals can either be received from the Boss component or generated by the HumanBehaviorModel using the same XML file as the Boss. When a goal is selected, it is sent to the Supervisor for execution.

This component is also responsible for detecting and handling navigation conflicts. Currently, the kind of navigation conflict handled by InHuS is path blockage (e.g. another agent standing in a doorway). While the human agent is navigating, a path to the goal is calculated at regular intervals using Dijkstra's algorithm, and its length is tracked to detect such conflicts. If the tracked path length increases significantly or the path ceases to exist, it could mean that another agent is blocking either the only possible way or the shortest way. When such situations are detected, the plan execution is temporarily suspended, and the agent performs an approach action to get close to the blocking location. This shows the agent's intention to move in a specific direction and might induce the blocking agent to react and clear the way. Eventually, once the avatar is at a specified distance from the blocking location, set to $1.5m$, the agent stops its approach and actively waits for the path to be cleared.

To generate a lot of different and specific situations, we created what we call *Attitudes*.

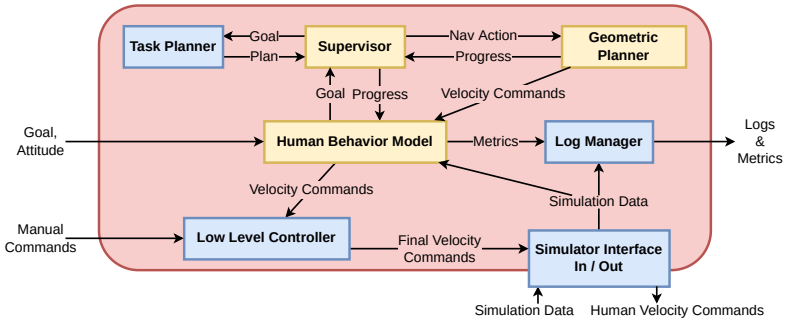


Figure 7.2: The human controller InHuS is depicted with its components and subsystems.

They are operating modes affecting both goal decisions and reactions toward the other agents. One can activate them through the Boss to generate diversified behaviors of the agent. Some of the *Attitudes* currently implemented in InHuS are the following: 1) randomly picking a new goal, like someone suddenly changing their mind; 2) harassing the robot by constantly going in front of it, like a child would do [Nomura 2016]; 3) stopping close to the robot and looking at it for a few seconds before resuming its goal which emulates a curious behavior.

The final purpose of this component is to build the perception of the human agent based on the map and information about the other agents from the simulator. We build the perception by directly accessing the simulation data rather than adding simulated sensors to the human avatar. Using this perception, we compute the visibility of the human agent and then update the human's knowledge about the robot's position and speed.

7.2.2.2 Supervisor:

The Supervisor is a central component as it coordinates different components to execute the plan and achieve the current goal. When the Supervisor receives a goal from the HumanBehaviorModel, it requests the TaskPlanner component a plan to achieve the goal. For now, the plan generation is quite simplistic. For a "Pose goal", a plan filled with a single navigation action is generated. For a "Compound goal", the navigation and waiting actions sequence is extracted from the XML goal file, and the plan is populated. Despite the simplistic plan generation, this architecture handles complex goals that require several steps to be achieved and emulate human activities.

The Supervisor then supervises the execution of each action of the plan by sending requests to other components. When a navigation action needs to be performed, the Supervisor starts by sampling a random position if the given action radius is not zero. Then, it requests the GeometricPlanner to plan for the target position without considering other agents initially. This way, the avatar starts following the shortest path, and we initialize the conflict detection. After this, the system starts to consider the other agents, and the Supervisor periodically requests the HumanBehaviorModel component to check for potential navigation conflicts. The Supervisor can suspend and resume the plan execution at any time, which can be used to resolve the detected conflicts or to generate specific reactions like the *Attitudes*.

7.2.2.3 GeometricPlanner:

The last major component is the GeometricPlanner. This motion planner component receives a target position from the supervisor to reach and generates velocity commands to make the avatar move. This component defines how the agent moves around and adapts its velocity to the other agents in the scene. Since the system is implemented in ROS, we use the standard ROS navigation stack for the GeometricPlanner.

The planner used in InHuS is a publicly available human-aware navigation planner called CoHAN [Singamaneni 2021]. It is built over the ROS navigation stack and uses a local planner based on a modified version of the timed elastic band with human-aware properties. We benefit from the high-level decision-making of InHuS and the enhanced local navigation of CoHAN with trajectory predictions. Moreover, CoHAN is highly tunable, which helps generate different agent behaviors.

7.2.3 Logs, metrics and GUI

The InHuS system logs the execution data, such as the positions and speeds of the agents, along with some computed metrics. All the logged data is sent to the GUI component, which generates interactive plots. These plots can help evaluate the interaction and, thus, the performance of the given robot controller. The snapshot of the GUI shown in Fig. 7.3 shows two kinds of visualizations. On the right side, there is a colored visualization of the paths taken by each agent. These paths are colored over time according to a corresponding legend that helps estimate an agent's position at a specific moment. The left side comprises several plots showing some computed metrics over time. The first plot is about conflict detection and solving. It shows the path length to the goal computed when checking for conflicts. Without any conflict, the path length should decrease linearly over time. If it's not the case, the avatar has been disturbed during the navigation. This plot also shows the state of conflict of the agent: Nominal (no conflict), approach (conflict detected), blocked (stopped and waiting). The subsequent plots show the speeds of each agent over time, their relative speed, the distance separating them, and a metric called time to collision (TTC). This metric estimates the time remaining before the agents collide with their current velocities. We can argue that TTC corresponds to a "threat feeling" since a low TTC value corresponds to a high collision threat. Hence, social robots should be tuned not to exceed a minimum TTC value to make humans more comfortable.

7.3 Main results

In this section, we show some results through a set of experiments to highlight how our system can help challenge human-aware robot navigation systems. First, we discuss the limits of reactive-only systems to strengthen the need for rational avatars. Then, we present how our system effectively challenges robot navigation systems, and we interpret the corresponding plots. Next, we show how the InHuS System can compare the human-aware performances of two different robot controllers. Finally, we present additional experiments showing the diverse behaviors that can be produced using the *Attitudes* and how "long runs" can benefit the development of a robot controller.

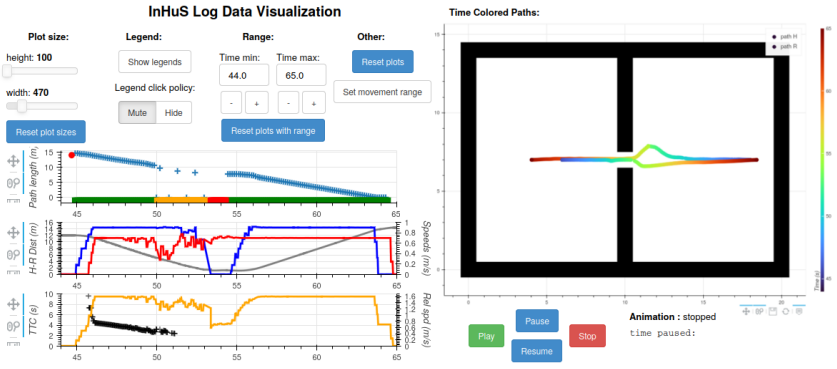


Figure 7.3: An overview of the GUI interface. On the right side, the paths taken by the agents are shown and colored over time. On the left side, several metrics and data produced by InHuS are plotted over time on graphs. Additional widgets help to configure the plots.

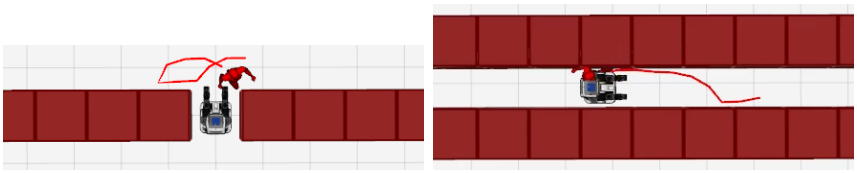


Figure 7.4: In the doorway scenario (left), the reactive-only (Pedsim) agent never stops moving while trying to go through the robot, even though its path is blocked. In the narrow corridor scenario (right), the agent squeezes itself between the wall and the robot, colliding with both.

7.3.1 Limits of reactive-only agents

Most of the current human agent simulations used by the social navigation community rely either on the social force model or ORCA. In order to highlight the limitations of such approaches, we present results obtained with a PedSim_ROS (or simply PedSim) agent. PedSim is a pedestrian simulator that uses the social force model. It is very efficient for generating crowds to test robot navigation. However, at the individual level, the simulated agents are purely reactive and have no decisional abilities like most pedestrian simulators.

Consider the doorway scene shown in the left part of Fig. 7.4. Both agents have to cross a narrow opening. Here, the robot is blocking the way that the human agent intends to cross. The PedSim agent approaches the robot and tries to push itself through, but it fails due to a very high value of social force. The agent never stops moving and tends to go right or left along the wall before wiggling again just in front of the robot. This confusing behavior can make the agent's intentions unclear to the robot planner. The narrow corridor scenario, shown in the right part of Fig. 7.4, also exposes some limits. In this scene, there is not enough space for the agents to cross each other. The only solution is for one of them to back off. Here, the path is blocked by the static robot. The PedSim agent slowly gets closer and closer to the robot before squeezing itself between the wall and the robot. For some reason, the social forces allowed the agent to pass, unlike the previous example. It highlights that the PedSim agent does not use a defined hitbox or footprint for the agent and relies only on repulsive social forces to prevent collisions. This lack of defined collision shapes makes the agent temporarily pass through the walls and other agents. Consequently, it breaks many intricate scenarios where a rational decision should be taken, resulting in unrealistic situations. Despite being efficient for large spaces or crowds, based on the above observation, we can state that such approaches can lead to confusing and even unrealistic behaviors in intricate scenarios.

7.3.2 Interpretation of plots with human-aware planner

The InHuS System is able to generate challenging situations and associated logs to allow further evaluation. Here, we present one such conflict and a detailed interpretation of the corresponding plots. The plots were produced while challenging the CoHAN system in the doorway scenario.

The robot starts closer to the opening and enters the doorway first. The execution can be analyzed with the metric plots and the time-colored paths of the agents in Fig. 7.5. We notice that the robot's speed (red line on the second graph) goes down around 50 s as it enters the doorway and creates a conflict. The conflict is detected by InHuS (zero path length = no path), and the agent switches to the approach state (green to the yellow line on the first graph). The non-zero path length in the approach state corresponds to how the approach is performed. In order to keep moving despite the blocked path, the

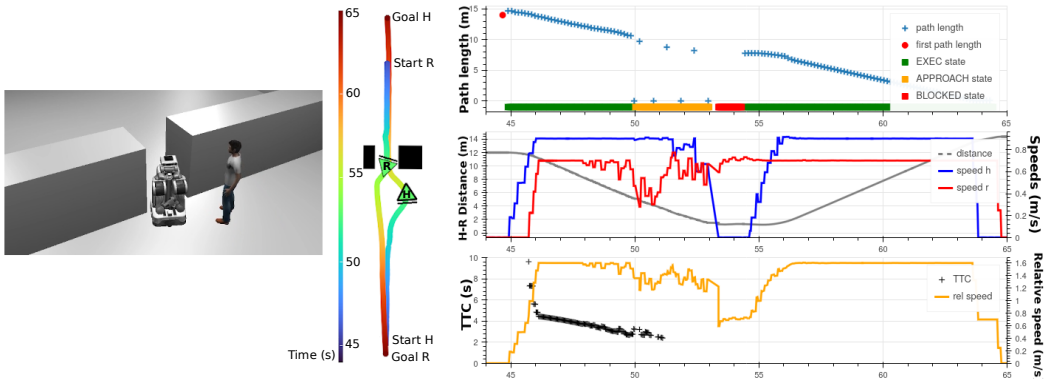


Figure 7.5: A condensed view of the InHuS GUI and MORSE simulator for the doorway scenario with a robot running the CoHAN planner. Several plots depict the detection and resolution of the conflict created.

GeometricPlanner is requested at a defined frequency to plan without considering the robot (all non-zero path length). In between these requests, to check if the path is still blocked, the conflict detection plans while considering the robot (zero path length). When the avatar is at a predefined distance from the blocking robot around 53 s, it switches to the blocked state (red line) to stop and wait for the path to be cleared. Further, the time-colored paths show that the GeometricPlanner made the avatar move aside while approaching to avoid blocking the robot. As a result, the agents were no longer moving toward each other, and thus, there was no longer any collision threat (no TTC values). When there is no more collision threat, around 51 s, the robot's speed starts to increase again. Such behavior is a good sign of human-aware properties and might increase human comfort.

From the plots produced by our system, a lot of useful information can be extracted for improving or evaluating the social robot planner's performance like a) finding ways to decrease the blocked state time for the human, b) maintaining a particular threshold for TTC, c) slowing down near the human, or waiting for the human to cross the door without blocking.

7.3.3 Quantitative comparison between two robot controllers

Our system can be used to run similar scenarios repetitively to produce robust metric values. These values can help to evaluate the human-aware performances of a given robot controller. To show this, we present a comparison between two different robot controllers. The first one is again the CoHAN system, and the second one is the Simple Move Base (SMB). It uses the *teb_local_planner* and the ROS navigation stack with default parameters. We just add an additional process to consider the human agent as a static obstacle to avoid it, so it is not human-aware. Therefore, we should be able to notice a clear difference through the metrics computed by our system. For this comparison, we used three different scenarios: 1) The doorway scenario, where the agents have to cross a narrow opening; 2) the corridor scenario, where the agents cross each other with just enough space; 3) the open space where they cross each other without any environmental constraints. We performed ten repetitions of each scenario for each robot controller. For each set of 10 repetitions, we extracted the mean values of three different metrics and presented them in Table 7.1. The metrics are the following. First, the time to goal (TTG) is the time taken by the avatar to reach its goal. Second, the minimum distance between the robot and the human (Min HRDist). And the minimum time to collision (TTC). Intuitively, we want the TTG to be as small as possible,

Scenario	CoHAN			SMB		
	TTG (s)	*min Dist (m)*	*min TTC (s)*	*TTG (s)*	*min Dist (m)*	*min TTC (s)*
Doorway	18.38	**2.32**	**1.33**	**18.26**	2.23	1.16
Corridor	**16.34**	**2.06**	**1.03**	17.05	1.59	0.81
Open Space	**9.55**	**2.52**	**1.61**	11.01	2.34	1.18

Table 7.1: Mean values of three InHuS metrics over ten repetitions in three different scenarios and with two different robot controllers. Bold values indicate when the corresponding robot controller performs better than the other.

the minimum HRDist to be as high as possible, and since a low TTC value represents a collision threat, we want the minimum TTC to be as high as possible.

At first glance, Table 7.1 shows that almost all CoHAN values are better than SMB values. Due to the nature of the doorway environment, the execution of the scenario is quite constrained, which explains why the values are not too different between the two controllers. However, we notice anyway that, compared to SMB, the CoHAN planner tends to keep a greater distance between the agents and a greater TTC (lower collision threat). The time to goal of CoHAN is slightly higher because the robot slows down when crossing and moving in the direction of the human. Thus, in this scenario, it is the price to maintain adequate TTC values.

In the corridor scenario, The SMB robot tends to wait until the last moment to move aside, which is threatening. On the other hand, the CoHAN robot proactively moves to one side of the corridor. As a consequence, it leaves more space for humans and reduces the threat of collision, which is visible in the obtained values. Also, this pro-activity has the effect of smoothing the trajectory of the avatar, which makes this last one reach its goal faster.

Finally, the open space scenario is a bit similar to the previous one. The SMB robot waits until the last moment to avoid the human, which puts the load of the avoidance maneuver on the human. As a result, humans have to move aside, extending the duration of their efforts to reach the goal. Also, the SMB robot is closer to the avatar and more threatening on average due to the same behavior. Since the CoHAN robot moved again aside early, its metric values are noticeably better than SMB.

In summary, the human-aware behavior of the CoHAN controller was captured through significant value differences in the computed metrics compared to a non-human-aware robot controller. This implies that our system can help evaluate and compare human-aware robot controllers.

7.3.4 Generating different behaviors with Attitudes

By activating *Attitudes*, InHuS is capable of producing more complex behaviors to diversify the conflicts and challenges imposed on the robot. We present the time-colored paths for the execution of two *Attitudes*: *Harass* and *StopAndLook* in Fig. 7.6. Concerning the *Harass Attitude*, by paying attention to the colors, we see that the human is always in front of the robot that continuously tries to avoid the harassing agent, causing erratic movements. The robot should be able to detect such non-cooperative behavior from humans and act accordingly. On the same figure, we see the execution of the *StopAndLook Attitude*. The color discontinuity behind the human marker shows how the human suspended its goal to stop and briefly stare at the robot before moving again. A robot that is not proactive

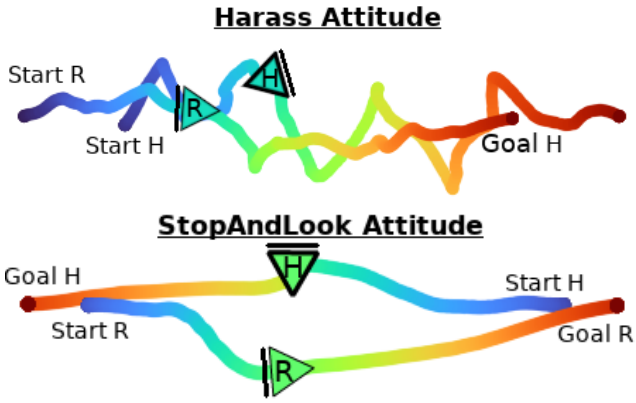


Figure 7.6: Behaviors obtained by activating the *Harass* and *StopAndLook Attitudes*. With *Harass*, the human is always in front of the robot. With *StopAndLook*, when close to the robot, the human stops to look at it for a few seconds.

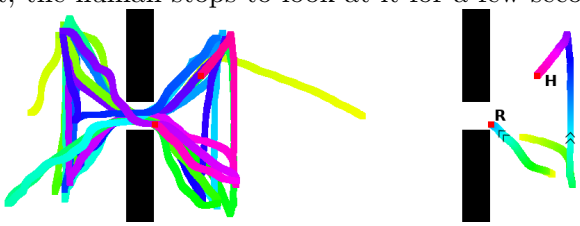


Figure 7.7: Execution of the long run scenario using the TDP robot planner and InHuS. We see the complete set of time-colored paths on the left. On the right, the same path is cut around when the robot gets stuck in the wall.

enough could be disturbed by the sudden stop of the human, which could be a situation of interest to handle.

7.3.5 Long run scenarios

The proposed system can help test the stability and robustness of the robot planner by conducting long randomized runs. Indeed, thanks to the Boss component, possibly randomized goals can be sent autonomously to the agents. This can generate unexpected situations and conflicts that can be of interest. Fig. 7.7 depicts such a test conducted with InHuS and a human-aware robot planner from Kollmitz et al. [Kollmitz 2015] here referred to as TDP. The agents were made to endlessly loop over four goal positions (each with a $1m$ radius) in reverse order to create as many conflicts as possible. After 3 minutes, the robot got stuck in the wall of the doorway, indefinitely blocking the path of the human. In addition to highlighting problematic situations where the robot does not act as expected, long runs can expose low-level issues like unexpected crashes or memory leaks.

7.4 Discussion and Limitations

Although InHuS provides an autonomous human agent, the agent can be controlled manually if needed. We do not yet provide a handy controller, but velocity commands generated by any means can be sent to the Boss component to control the human. This extends the usability of InHuS as one can use scripted trajectories or motion capture to control the

human agent in the simulator.

The proposed system interacts with an external simulator and robot controller. Since the system is mainly implemented using ROS, switching from one simulator to another is straightforward if it has a ROS interface. InHuS has specific components to abstract the simulation data format. Thus, just by slightly editing these components, we were already able to run InHuS on three different simulators: MORSE [Echeverria 2011], Stage[3] and Gazebo [Koenig 2004]. Furthermore, any robot controller using the ROS Navigation Stack can be directly used with InHuS.

Simulating intelligent human avatars is a novel field, and only a few works apart from ours have tried to address this limitation. A similar work in ROS2 was recently presented in [Pérez-Higueras 2023]. Clearly, the idea of intelligent human agents is of interest to the community, and it is necessary to test social navigation effectively. Like any other system, InHuS has limitations, too. We claim to generate only reactive and somewhat rational behavior, which is still far from natural or realistic human behavior. We currently handle scenarios with two agents only: the human and the robot. We can run scenarios with other human agents, but they will be treated like robots.

7.5 Conclusion

Human-aware social robot navigation is rapidly growing, but the community lacks good human agent simulations to test and debug their systems. The existing reactive approaches offer only limited testing. Through the InHuS system, we proposed a pertinent approach to address this issue. We showed that our system could generate conflicting situations that need resolution by making rational choices. Moreover, all the metrics and data recorded during execution and their visual plots allow us to evaluate the interaction and behavior of the robot. With such evaluation, we showed that we could compare different robot controllers. InHuS can also generate various tunable behaviors that can diversify the situations and conflicts imposed on the robot, and thus, it helps to debug and tune the system. Long runs provide additional potential ways to improve the system.

[3]https://github.com/ros-simulation/stage_ros

Chapter 8

Interactive Social Multi-Agents Simulation for Robot Navigation: IMHuS

Contents

This chapter presents the IMHuS system designed to choreograph several agents with group movements and social behaviors. This system complements the InHuS system presented in the previous chapter by generating multi-human scenarios. This system has been qualitatively evaluated in an elevator scenario.

8.1 Introduction

In chapter 7, we showed that the InHuS system effectively challenges robot navigation systems in intricate and human-populated environments. However, simulating the interactive agent is computationally demanding. This is why we limited the simulation to only a single intelligent agent. Nevertheless, it is also relevant to general intricate scenarios with several agents. Existing works simulating crowds are only based on a reactive approach and are not necessarily designed to be used to benchmark robot systems.

This is why we propose the Intelligent Multi-Human Simulator (IMHuS). This work is strongly inspired by and complementary to the InHuS framework presented in chapter 7. Together with two researchers from the University of Leon in Spain and an intern, we designed this framework based on InHuS. The implementation of this system has been majorly done by the intern. This system allows choreographing several interactive agents with group or individual movements and social behaviors. The individual agents are less complex and demanding than the InHuS one, allowing the simulation to run smoothly. This

system has been evaluated in the elevator scenario, as defined for the SciRoc competition 2019.

We begin with a comparison between InHuS and this additional work while briefly describing it. After, a more formalized presentation of IMHuS is provided, detailing the information given in the prior comparison. Eventually, the elevator use case evaluation is presented.

8.2 Comparison InHuS vs. IMHuS

8.2.1 Similarities

Let's first mention the similarities between InHuS and IMHuS work. Like InHuS, this work aims to replicate scenarios involving humans to help social robotics research. Similarly, the simulated interactive agents can be choreographed in a step-based manner to create high-level social behaviors such as waiting for an elevator, getting in and out of it, or standing in front of a store window and moving to the next one. The agents navigate in the environment while avoiding static obstacles and being reactive to moving obstacles to prevent collisions. The agents can also wait for a defined amount of time or turn to look in a direction. Like in InHuS, *Attitudes* can be activated to generate specific behavior, such as harassing the robot. This work also analyzes the execution to compute metrics evaluating the robot's performance in defined social situations. The framework architecture is close to the InHuS one and can also be used with different robotic simulators. Here, it has been implemented with Gazebo.

8.2.2 Differences

However, IMHuS differs from InHuS in several ways. The primary reason is that it manages several interactive agents instead of a unique one in InHuS. Moreover, the agents can exhibit social behavior using explicit social groups. For instance, they can move together to another location or talk in pairs. This requires the new *grouping* actions which create or dismantle groups of humans. An interesting addition is the implementation of asynchronous actions. They differ from synchronous actions, such as navigation/turning/grouping actions, which the agents accomplish during one specific step. Asynchronous actions are not associated with a specific step and correspond to a system of *request* and *response*. Thanks to them, one agent can request another to perform a specific task. For instance, the robot can request a human agent to call the elevator, or if supported, a human agent can request the robot to go somewhere. Not every agent can respond to the robot's requests. For instance, if the robot asks for someone to press the button and no one is around, the request will be dropped, counting as a failure.

In order to handle several human agents, some simplifications were made compared to the InHuS framework. First, when navigating, IMHuS agents are reactive to other agents. However, their movements are less smooth than the InHuS agent. This is because InHuS couples a frequent global path replanning, taking into account static and moving obstacles and an elaborated local planner to follow the planned path. The local planner used is the Human-Aware robot navigation planner CoHAN, which was presented in the previous chapter. It also allows the agent to have proactive avoidance movements. Also, InHuS uses frequent global path replanning to be even more reactive and to identify sudden path blockage due to other agents. Hence, instead of blindly following the global path it can identify when its optimal path is blocked and switch into a conflicted mode where it reasons

on its goal to adapt it potentially. Currently, the agent approaches the blocked spot before stopping to wait for the path to be cleared. On the other hand, IMHuS agents do not use any local planner and simply move at a defined speed along the updated global path. This induces human agents to sometimes move abruptly and avoid obstacles at the last moment. In addition, IMHuS agents' actions are dictated by the given choreography and do not have individual goal reasoning processes like in InHuS. Hence, IMHuS agents are currently incapable of identifying path blockage situations and may behave erratically in such cases.

However, this new scheme is still very interesting because of the multitude of human agents generated and the social behaviors that can be choreographed. It can generate relevant and challenging situations for social robotics to handle.

8.3 Rational

The goal of the tool described in this paper is to provide a means to generate scenarios in which predefined social interactions of groups of reactive humans can be used to test the social performance of a robot behavior under evaluation. In the rest of the paper, we will refer to it as *tested robot*. The aim is to use the system to benchmark human-robot interaction behaviors.

The goal of IMHuS, the tool described in this paper, is to provide an open-source toolkit for defining high-level reactive simulated humans with the ability to show the behavior of social groups but using realistic standard robotics simulators that allow researchers to use models of their real robots, both for debugging their algorithms and for benchmarking and repeatability.

8.4 Design of a step-based social simulation

The goal of IMHuS is to standardize the validation of autonomous robot behavior in the presence of people, allowing researchers to design repeatable human social situations. For example, we can define a set of waypoints where different simulated people arrive, meet, and move as a group to a different position, or two people facing each other and then moving together to a different location.

Our proposal considers that both individuals and groups must be included in the simulated environment and that each simulated person should exhibit adaptive behavior in both cases. To achieve this goal, we assume that global path planning for the whole set of agents is more suitable for defining fixed social behaviors than the individual path planning approach. This assumption also serves the purpose of humans exhibiting a typical social behavior that the robot must be able to detect in order not to cross, for example, through the middle of a social group.

A central process can have perfect knowledge of the simulated environment, accessing the real and accurate positions of all the elements of the simulations (obstacles, robots, etc.). It is not designed to emulate an embedded agent since the robot simulation process has to obtain the information through the noise-simulated sensor readings provided by the simulator. This process can use the API of the simulator to get all the information directly, without noise and identification problems (it will know which types of elements are in the simulation and their state). For instance, it does not need to identify if something is “a door" or if it is opened. The door state is obtained directly from the simulator.

8.4.1 Choreography oriented simulation

The constraints and assumptions made for IMHuS are:

1. The set of persons $P = \{p_1, p_2, ...p_n\}$ in the environment is defined *a priori*, and each person will be unequivocally identified by its ID (p_i) during each execution.
2. The number of locations $L = \{l_1, l_2..., l_m\}$ for these people is also known *a priori*.
3. The number of locations will be greater than the number of people $|L| \geq |P|$.
4. In a given time-step, only one person can stay at an individual location l_i.
5. Group locations L_i^n will have a maximum number n of individual positions $l_1, l_2, ..., l_n$. For instance, an elevator with four positions will be named L^4.
6. The number of actions is also finite and known.

The tool aims to give researchers a high-level definition of a "choreography" of people moving in a social way. For instance, let's consider an example: first, a set of people (p_1 to p_4) is defined. In the first step, p_1, p_2 and p_3 must move from their initial positions to a "group location" (L^3). This joint navigation is represented in figure 8.1 as a continuous top-opening box grouping the three people at time-step t_0 and a double arrow labeled with the goal destination. In the same way, p_4 will remain in its position but will face a particular direction (30 degrees), indicated by the circle with an arrow.

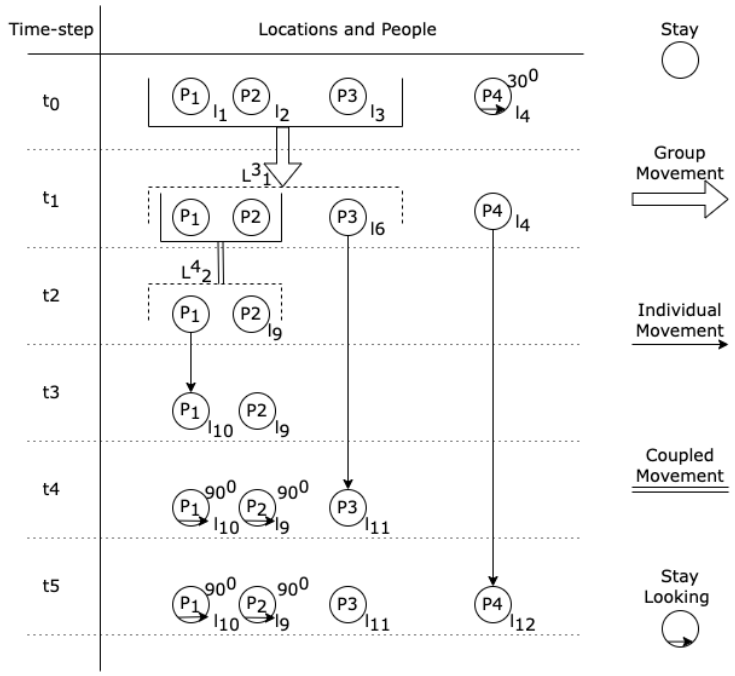


Figure 8.1: Schematic representation of the definition of a social navigation choreography

At t_1, p_1 and p_2 engage in a pair-move to L^4, while p_3 and p_4 initiate individual moves, indicated by a single arrow. A pair-move is a specific type of group move included in the design because it is usually the most common and can be considered the smallest group unit. At t_2, the two persons moving as a pair would have reached their destination, and

p_1 will initiate its movement towards l_{10}, while p_2 remains in its current position with no particular orientation (indicated by a circle with no arrow), and p_3 and p_4 continue to navigate to their targets.

p_1 at t_2 begins to move individually towards l_{10}, which reaches at t_3 while p_3 and p_4 continue to execute their solo movement. p_3 reaches l_{11} at t_4, when p_1 and p_2 begin to face in the same direction. Finally, at t_5, p_4 reaches its destination (l_{12}) and the choreography ends.

Time-step (t_i) in figure 8.1 means "*choreography steps*", that is, significant moments for the definition of the simulation. It does not refer to a magnitude measured by a clock. We will refer to them as *steps*, which is an increased ordered sequence of discrete points in time at which a given set of events has to occur. For example, $step_0$ usually specifies the initial state of all the elements of the simulation, i.e., the position of the robot, human', and the other elements. A typical step specifies a set of actions that are initiated at that instant, a navigation task for one of the humans, for instance, at $step_1$ (t_1 in figure 8.1).

This is the type of simulation that the tool should be able to generate. Figure 8.2 shows the general architectural framework. This tool receives the definitions of the social scenarios in a definition file (an XML in its current version). It then manages the simulation, obtains the simulation data, uses existing libraries and other tools (such as *move_base* to calculate trajectories in ROS), and finally generates the log data for evaluation.

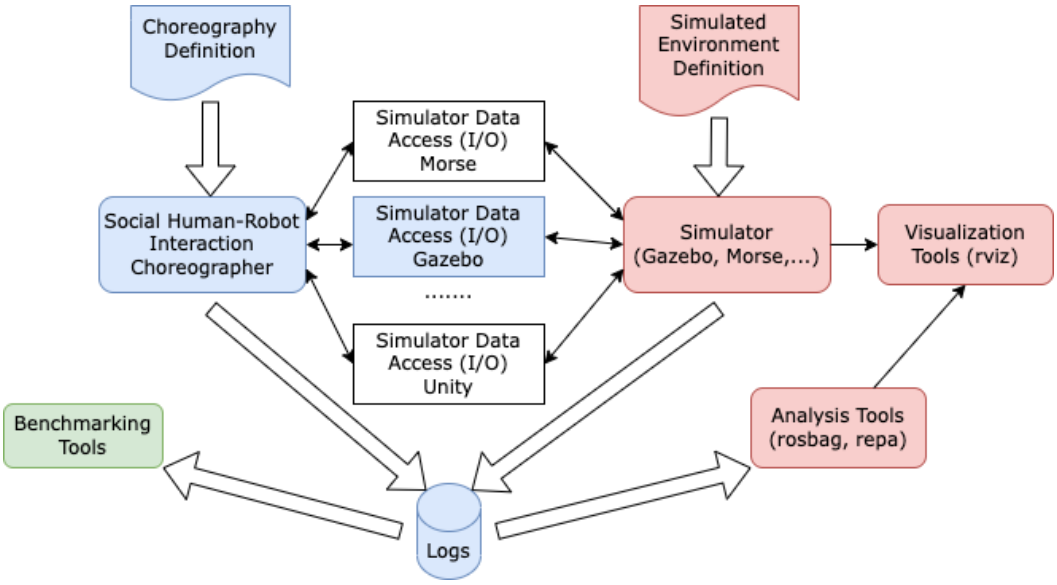


Figure 8.2: Global simulation architecture. Red signifies existing tools. Blue components are those described in this paper. Green signifies ongoing development. Uncolored boxes are alternative simulators under consideration.

The definition of the choreography shown in the upper left of the figure 8.2 is defined in XML. The design allows five types of *basic actions* that the *agents* (simulated robots or simulated people) can perform:

Navigation actions : Those actions modify the global pose of the agents in the environment. Typical actions in this group are *GoToPose* and *Wait*.

Turning actions : Actions related to the facing of the agents, such as *LookAt* and *Turn*, depending on the way the action is specified.

Grouping actions : These actions manage the creation and dismantling of groups.

Attitude actions : Actions modeling the high-level behavior of the agent. For instance, a simulated human can be ordered to *harass* the robot.

Synchronous actions : Actions related to the environment to be accomplished by an agent during one specific step of the simulation. They have been standardized as *publish* and *subscribe*.

Asynchronous actions : Actions related to the environment and not associated with a specific step in the timeline of the simulation. They have been standardized as *request* and *respond*.

These actions can be applied to a single agent or a group. Basic actions can be integrated into a *compound_task*. For defining these actions, the following components are used:

- *map*: corresponds to the “world" where the simulation will happen. Inside the map, a set of *objects* can be defined, specifying their individual ID.
- *poses*: correspond to the “location" used in figure 8.1. They comprise the x, y position and orientation θ and a *radius* of tolerance for the motion planner to consider that the goal has been reached.
- *agents*: that can be either *robots* or *humans*, each of them identified by an unique ID. They are given an initial pose where they appear in the world.
- *groups*: they can be created and dismantled during the simulation

The evolution of the simulation is based on steps, as described in the previous section. The regular steps are synchronous, but there is an asynchronous step for interacting with the tested robot:

- A scenario is composed of a set of regular steps and a singular asynchronous step.
- Steps are made up of different actions like navigation, grouping, etc.
- Every action of a step is executed *at the same time*, meaning that the actions are executed in simulated parallelism.
- One step ends when all its actions have finished.
- The asynchronous step runs in parallel to the execution of the synchronous steps.
- The scenario ends when the last regular step ends.

8.5 Implementation

In the current version, choreographies are defined in an XML file that includes four main sections:

- Map = poses and objects. Example: poses definition.

```
<poses> ::= <pose> (<pose>)* ;
    <pose> ::= <poseID> <x> <y> <theta> <radius> ;
```

- Agents = humans/choreographed robots with their initial poses and groups with their composition. Example: group definition.

```
<groups> ::= (<group>)* ;
    <group> ::= <groupID> <pair> (<human> (<human>)*) ;
        <pair> ::= <pairID> <human> <human> ;
```

- Tasks = generic actions. Example: compound tasks and look-at action definition.

```
<compound_tasks> ::= <compound_task>* ;
    <compound_task> ::= <compound_taskID> <action> (<action>)*;
<lookAt_action> ::= <actionID> (<humanID>|<robotID>|<objectID>);
```

- Scenarios = step elements of synchronous steps and asynchronous steps. Example: scenario definition.

```
<scenario> ::= <name> <step> (<step>)* (async_step);
    <step> ::= <stepID> <stepElement>;
        <stepElement> ::= <agentsID> <pose>|<compound_task>;
        <agentsID> ::= <humanID>|<robotID>|<pairID>|<groupID>;
    <async_step> ::= <respond_event_action>;
```

The prototype has been implemented as a new version of the InHuS tool ([Favier 2023]), which connects to the Gazebo simulator to obtain information about the world. It uses the *move_base* ROS to generate the global plan for each agent and updates the positions of each agent in the next step of the simulation accordingly. The new version is capable of handling the navigation of multiple agents (humans) in parallel while managing conflicts and completing individual tasks as in the previous version. It also provides an updated graphical user interface for repeatedly selecting and executing scenarios defined in the XML file.

The IMHuS tool shown in Figure 8.3 has been structured in three layers: the IMHuS layer (in blue), its configuration in the application layer (in yellow), its communication with the simulator and ROS (in red). The robot whose behavior would be tested in the tool has also been included (in violet).

Application layer This layer includes the XML configuration file needed to use IMHuS by defining its main components: map, agents, tasks, and scenarios. The map includes all the locations of the agents to be used during the choreography and those of static objects. These locations are represented as poses, and a name is assigned to each. The description of the agents includes the name, and initial position of all humans and robots choreographed, as well as the name and composition of the groups that will appear at any step. Generic tasks are described with no specific subject to perform them so that several agents can reuse them. Lastly, the scenarios describe the choreography steps. Each step includes a set of tasks assigned to a particular agent or a group. They are the step elements.

IMHuS layer The tool interprets the information contained in the XML configuration file to represent the tasks as actions and the humans/robots as agents. When the scenarios are run, a command combines actions with agents for every step element. The commands concerning each agent are executed in a separate thread inside a step. The step finishes when all the threads are done. This is the way simultaneous movements of agents are achieved. This is the behavior of the tool for the choreography, but considering that the tested robot will be present, there must be a way for it to communicate with the simulation agents, just as it would happen in the real world. The asynchronous step is responsible for this task. At any point, the tested robot can

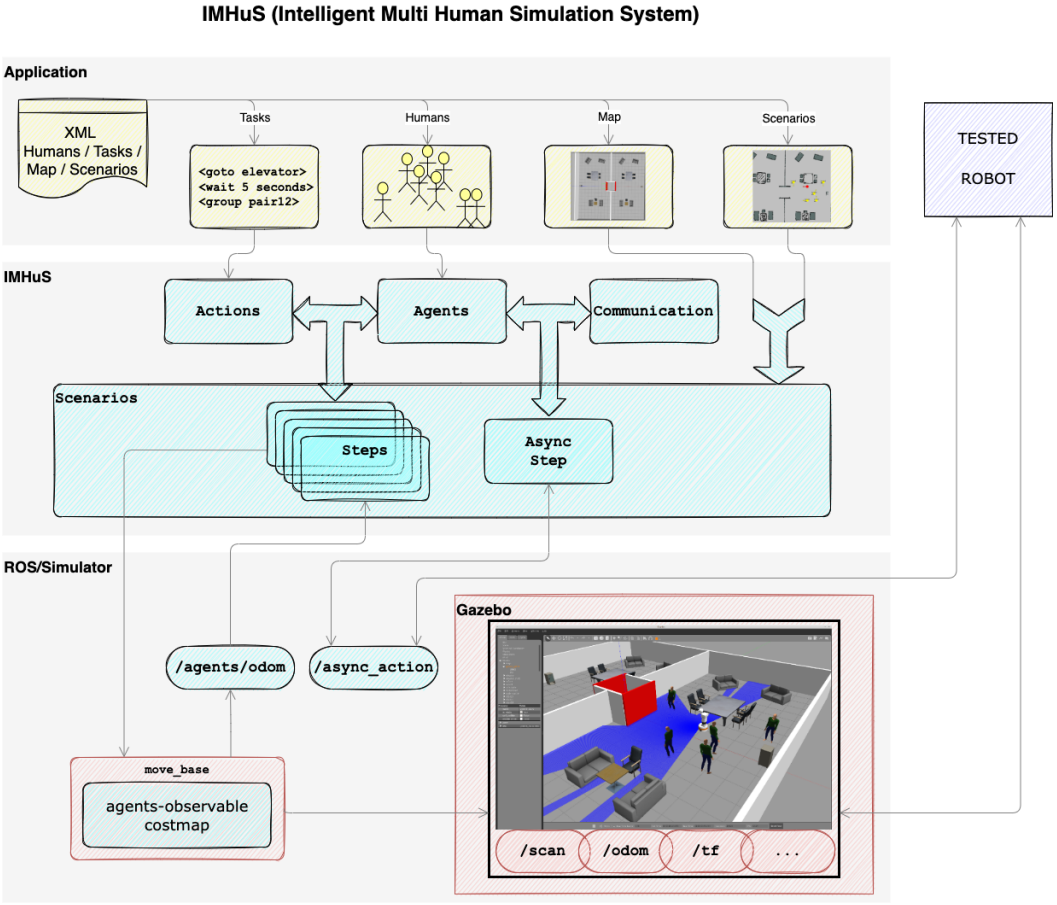


Figure 8.3: Structure of the new IMHuS tool.

ask something, for example, to press the elevator button. This action would be done through the communication module and answered by the agents in the simulation in this asynchronous step by a response action. Not every agent can respond to a request from the tested robot. For instance, if the robot asks for someone to press the button and no one is around, the request will be dropped, counting as a failure.

ROS/Simulator layer IMHuS uses this layer to place the agents in the simulation, ask for the trajectories to move them around, and communicate with the tested robot. The agents are placed in the world as obstacles so that they are avoided when `move_base` is asked for a new path. The costmap obstacle layer has been modified to obtain what we call the *agents-observable costmap* that IMHuS needs.

Interface of the tested robot The way to include the software of a robot in the simulator for its social behavior to be tested would be through the simulator, here Gazebo. Outside the simulator, the only communication would be through asynchronous actions. They allow the robot to send a request action to be answered by a response action from one agent of the simulation.

8.6 Use cases

This module is available as Open Source, and it has been evaluated in the elevator scenario, as defined for SciRoc competition (IMHuS repository link).

An elevator scenario will be used to show a running example of the behavior of the IMHuS system with a tested robot and also to explain the communication between the *tested robot* and the agents of IMHuS (video available at this link). The proposed scenario includes five human agents and the *tested robot*, whose goal is to go to the second floor. In order to do that, it has to ask a human agent to push the elevator button. Figure 8.4 shows the initial situation where the *tested robot* is approaching the elevator, human_2 is walking, and the rest of the humans are idle.

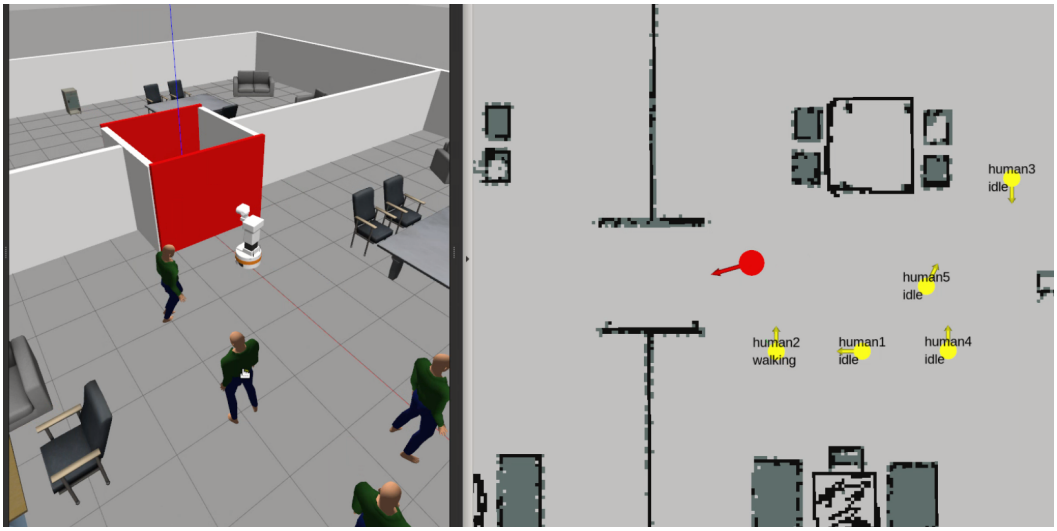


Figure 8.4: Initial situation.

Once the *tested robot* gets to the elevator door, it requests that one of the humans perform an action. In order for a *tested robot* to trigger an action in IMHuS, it has to publish a specific message type on the `async_action` topic (see Figure 8.3). Doing so triggers an asynchronous action response from IMHuS. This message should contain the time of emission, the name of the agent requesting it, the pose of this agent, and the name of the task it is requesting. If the scenario includes in its configuration an asynchronous action response corresponding to the requested action, that action is triggered on the human's side.

At this point, two different things can happen. If there is no human close enough to the *tested robot*, or there is a human but it is not in an idle state, no one will respond to the request, and it will be dropped, as Figure 8.5 shows.

The *tested robot* repeats the request every five seconds until human_2 enters an idle state and responds to the request. Figure 8.6 shows this moment of the simulation. When an asynchronous action request is triggered, IMHuS periodically checks if one human is able to respond. The requirements for a human agent to respond are to be in the idle state and within a $3m$ radius from the *tested robot* position when it requested the action. If several human agents can accept the task, the task will be assigned to the closest human.

Figure 8.7 shows the final situation where the *tested robot* has reached the second floor, simulated in the scenario as the room on the other side of the elevator.

Figure 8.5: Request dropped.

Figure 8.6: Request accepted by human2.

8.7 Conclusions and Future Work

The proposed tool, IMHuS, offers the possibility to create a realistic and challenging simulated environment in which groups of humans can be choreographed to evaluate the behavior of a *tested robot*. Different scenarios can be easily created using an XML configuration file in which social situations can be defined to measure the behavior of *tested robots* in a replicable environment. Furthermore, human agents are programmed to respond to interactions related to the particular situation of each scenario and their communication with the *tested robot*.

IMHuS's code is available as Open Source in the IMHuS repository. The current version has been implemented for the Gazebo simulator, but the design presented in section 8.4 can be easily migrated to other simulators, such as MORSE or Unity. The tool could be used to benchmark competitions such as SciRock or RoboCup as a previous step for the teams

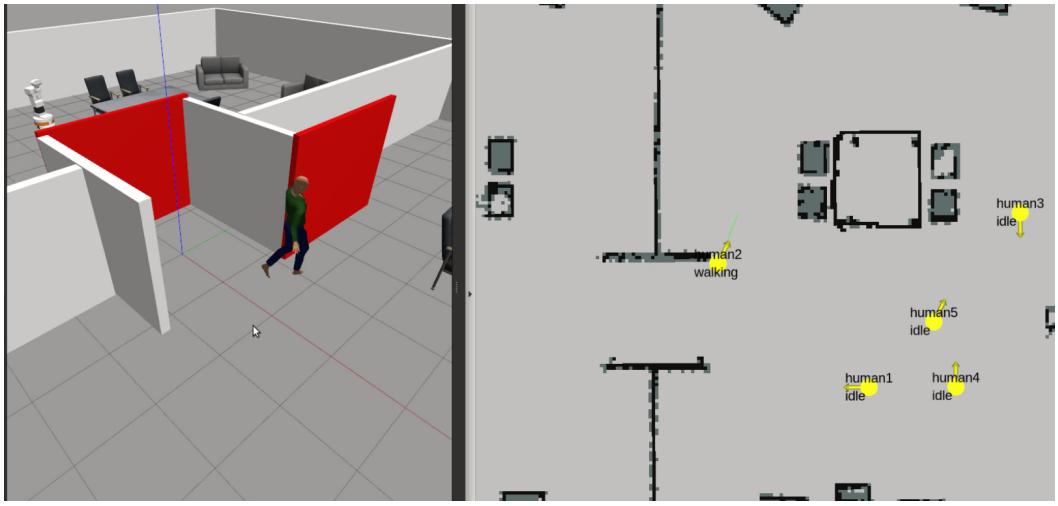


Figure 8.7: Final situation. *Tested robot* in second floor.

before getting to the physical robot challenges. To create a new scenario with IMHuS, all that is needed is a map and the configuration file to choreograph the human agents. The software has been designed to easily support the addition of both new tasks to be performed by the humans and new interactions between them and the *tested robot*.

One of the ongoing developments is the automatic generation of simulation metrics such as the distance between the robot and the agents, time-to-collision, etc. Another line of work is the extension of basic actions, especially towards the social behavior of groups of agents. Last, IMHuS is being tested with CoHAN [Singamaneni 2021], a human-aware robot navigation planner, to challenge CoHAN under several human-robot interaction settings. From this, we plan to identify the areas of improvement for human-aware navigation planners and provide a benchmark for testing these planners.

Conclusion

Contributions

In this thesis, we addressed the challenge of enhancing the decision-making of robots to ensure seamless collaboration with a human partner. Several contributions have been proposed, contributing to robotic human-aware task planning on the one hand and to robot navigation by simulating social interactive agents on the other.

To better capture my contributions, we presented in Chapter 1 the specificities and the multidisciplinarity of the Human-Robot Interaction field and the Human-Robot Collaboration subfield. Eventually, we presented state-of-the-art approaches in HRC task planning and robot navigation.

Task Planning Conclusions

Task planning for human-robot collaboration is a research topic of high interest to which numerous works have contributed. However, most of these works do not consider distinct agent models and, thus, do not consider different beliefs, action models, or goals. Moreover, works from the literature tend to produce a joint plan that must be shared and accepted by humans, assuming that humans are controllable and must follow the produced plan, which is an approach close to multi-robot planning. Finally, existing approaches usually assume the agents have already established a shared goal. However, exploring scenarios where the robot should decide 'when' and 'how' to start the collaboration is also relevant. Hence, there is a lack of task planning approaches that preserve the human latitude of online decisions while collaborating efficiently to solve both shared and individual tasks.

In Chapter 2, we presented the HATP/EHDA human-aware task planner designed to address the highlighted gap in the literature. After participating in its development, this planning scheme became a laboratory to explore relevant human-aware task planning challenges, leading to two main contributions.

In Chapter 3, we proposed models and algorithms to integrate concepts of Theory of Mind in the planning process of HATP/EHDA and plan the robot's actions appropriately. The main idea is that agents only update their beliefs from observable facts in their surroundings or by observing a co-present agent acting. More precisely, we modeled observability as follows. First, each fact describing the state of the world is associated with a place, which can be symbolic. The agents are situated and can move from one place to another. When situated in the same place, two agents are said to be co-present, and an agent and a fact are said to be co-located. Second, each fact is associated with an observability type, stating whether the fact is observable or not. For instance, the color of an object is observable and can be learned if situated near the object. However, the presence of salt in water or the temperature of an object is not observable, but it can be inferred in certain situations. Based on these models, we propose two rules to update agent beliefs. First, an agent learns from observing a co-located observable fact. These updates are done through a systematic Situation Assessment process inserted in the planning process of HATP/EHDA and executed after each planned action. Hence, the agent will automatically observe and learn from its surroundings. For instance, when moving to another room, the agent's beliefs will be updated with (only) observable facts located in the other room. There is no need to script this assessment in the '*move*' action's effects. On the other hand, an agent learns from

observing a co-present agent performing an action (including the agent itself). In this case, the agent's beliefs are directly updated with all action effects, including effects affecting non-observable facts. This rule models the inference that humans internally perform when observing their surroundings. For instance, if a human observes the robot adding some salt to the water, they can naturally infer that some salt is now in the water, although it is not observable. Using those two rules, we can maintain human beliefs more precisely to predict their behavior better and detect false beliefs that may be detrimental to collaboration. Finally, we solve *relevant* false human beliefs as follows. First, from the erroneous facts in human beliefs, we identify which ones must be corrected to solve the task. Then, a communication action correcting the minimal number of false beliefs is inserted into the robot's plan, potentially leaving some non-detrimental erroneous facts in the human beliefs. As a second possible solution, we check whether the false beliefs are due to a non-observed robot action. If so, we create another possible plan where the robot delays the relevant action until the human can observe its execution, avoiding the creation of a false belief. We evaluated this approach empirically using three domains, including shared tasks, several places, and non-observable facts. We showed that the proposed models and algorithms allow us to solve a broader class of problems than the original HATP/EHDA planner. Moreover, our relevant false beliefs detection and minimal communication already avoid systematic communication, and considering delaying robot actions reduces the communication rate of the solution plan produced even more.

In Chapter 4, we proposed another contribution to bringing planning and execution closer. We proposed a step-based model of concurrent and compliant joint action execution. This model describes several possible coordination of the agents, including the four following cases: 1) the human decides to perform any desired action and the robot complies by executing its best non-conflicting action in parallel; 2) the human decides to be passive and let the robot act alone; 3) the human is acting alone while the robot remains passive; 4) the human decides to let the robot decide and start acting purposely, then the human accompany the robot with a concurrent non-conflicting action. We use an abstracted version of this model to guide our search and explore further concurrent courses of action, allowing the robot to anticipate possible execution conflicts. The exhaustive exploration produces a directed acyclic graph where each path from the root to a leaf is a sequence of concurrent human-robot actions leading to the goal. From this graph produced offline, the robot's behavioral policy is extracted using an estimation of the human preferences regarding the joint task. These preferences can optimize various objectives, for instance, minimizing human efforts, finishing the task as soon as possible, or freeing the human as soon as possible. This policy indicates the best concurrent robot action to satisfy the estimated human preferences in every state and for each possible human action. This allows the robot to comply optimally with any human online decision. Moreover, this extraction is light and can be done online. As a result, as the execution progresses, the estimated human preferences can be reevaluated, and the robot policy can be updated accordingly on the fly. To evaluate our approach, we used BlocksWorld scenarios where the human and the robot had to collaborate to stack colored cubes to match a given goal pattern. As a first step, we evaluate the approach by symbolically simulating possible executions, following the abstracted model of execution proposed. To highlight the compliance endowed to the robot, we simulated erroneous estimations of the human preferences, making the robot's behavior more or less adversarial to the human preferences. However, despite contributing to the shared task, the robot may not necessarily achieve the task as the human would have preferred. The results showed that despite erroneously estimated human preferences, the actual human preferences are correctly satisfied overall because the robot constantly adapts

to and follows human decisions. To further validate the approach, we presented the user study we conducted in Chapter 6. This study is based on an interactive simulator created explicitly for this purpose presented in Chapter 5. This simulator includes a refined and implemented version of the joint action model used as an execution controller to supervise the robot's policy execution. This simulator allowed human participants to collaborate with a simulated Tiago robot through mouse control. We compared our approach with a contrasting baseline where the robot always imposes its decisions on humans. By recording execution data and requesting participants to answer a questionnaire, we showed that our approach performed and was appreciated significantly better than the baseline. Over the different scenarios, our approach satisfied the human preferences significantly better and the baseline. Moreover, the most significant differences using our approach are that the participant perceived the Interaction as more Positive, the Collaboration as more Adaptive and Efficient, and the Robot's Decisions as more Appropriate and Accommodating.

Social Navigating Agents Simulation Conclusions

On the other hand, navigation is a fundamental skill for a robot. State-of-the-art methods are efficient for static, dynamic, and unknown environments. However, navigating in human-populate environments is still challenging. Various approaches address the human-aware navigation subject, but only a few recent works propose tools to challenge, test, debug, and evaluate robot human-aware navigation systems. Without such simulation tools, preliminary experimentation and debugging must be achieved using a real robot, which is slow and burdensome. Most existing simulated interactive agents are based only on reactive approaches, making them suitable for crowded scenarios but limited and unrealistic in intricate ones. Thus, there is a lack of simulated interactive agents endowed with some decision-making processes allowing them to resolve more realistically intricate scenarios, like path blockage or narrow corridor crossing.

In Chapter 7, we proposed a complete system to simulate agents endowed with decision-making capabilities. The InHuS architecture has been designed to be generic, but we implemented it as a first step for the navigation use case. This system aims to challenge and evaluate robot navigation in intricate human-populated scenarios. It simulates an environment including a robot and a human avatar. The robot is controlled by a navigation system that challenges and evaluates. The InHuS system controls the human agent. Scenarios, including start position and a goal for each agent, can easily be defined and reproduced. The avatar can be given compound goals corresponding to sequences of navigating and waiting actions to simulate complex human activities. The avatar uses the CoHAN human-aware navigation planner to navigate, producing proactive and legible trajectories. Additionally, the avatar detects when the robot blocks the shortest path to its goal. Instead of following another path, the system switches to a conflict mode, makes the avatar approach, and eventually stops near the blocking spot to show its intention to cross. Once the path is cleared, the avatar proceeds with nominal navigation. To diversify the challenging situations, *Attitudes* can be defined in the system. They are modes that can be activated anytime, influencing the avatar's goals and reactions to the robot. We can simulate curious, distracted, and non-cooperative humans through these modes. There is also an *Endless* mode, making the agents repeatedly move to defined agent-specific positions. This is useful for generating unexpected situations and testing the robustness of the robot system over time. Another interesting feature is the ability to randomize all position goals. The final goal position is randomly sampled around the original within a given radius. To make the avatar's behavior more realistic, the system computes the visibility of the avatar and updates the robot's known position only when it is observable. Finally, the system

records execution logs, such as agents' positions, speeds, and additional computed metrics. Among these metrics are the Time To Collision, the estimated time until the agents collide, and the Surprised metrics. The latter helps identify sudden close robot appearances from behind the avatar. All these data are plotted on an interactive visual interface, providing real time feedback while running the system. We evaluated this system using two robot navigation systems. One is CoHAN, with human-aware properties. The other is close to the default ROS robot navigation stack, without human-aware properties. After running both systems in a few challenging scenarios, the data gathered with InHuS indicates that, as expected, CoHAN exhibited significantly better human-aware properties than the default system. This suggests that InHuS can effectively challenge and evaluate the human-aware properties of a robot navigation system.

Eventually, in Chapter 8, we proposed the IMHuS system, a complementary approach to the InHuS system. Indeed, InHuS is limited to simulating a single avatar. IMHuS permits the choreographing of several agents and the production of group and social behaviors. The IMHuS agents do not have individual decision-making capabilities. However, this framework allows producing intricate scenarios with multiple human agents to challenge robot navigation systems. Hence, it is an interesting solution between intricate single-agent scenarios and wide crowded scenarios. We showed the effectiveness of the approach in an elevator scenario.

Limitations and future works

In this section, we discuss the limitations of the contributions presented in this thesis and the possible future works they suggest.

Theory of Mind in Human-Aware Task Planning

There are some limitations to the approach presented in Chapter 3 that introduce Theory of Mind concepts in the HATP/EHDA planner. First, we do not consider uncertainties in the robot's knowledge about the world. Despite being linked to agent knowledge, and thus chapter 3, this limitation is common to everything linked to HAPT/EHDA. Indeed, since the planner is part of the robot, there is no choice but to rely on the robot's knowledge about the world, hoping it is correct. An interesting future work could be to abstract the perception layer of the robot and associate each state variable to a confidence level. This confidence level would indicate how much the corresponding fact is sure. Then, the robot could rely only on the high-confidence ones.

Secondly, we do not consider uncertainties in human beliefs. We assume to be in the worst-case scenario where humans do not know about facts they did not see. However, considering the pasta cooking example of Chapter 3, the human could assume that the robot effectively did its work while being away fetching the pasta. Thus, the human could believe that there is salt in the post even without seeing the robot's action. Keeping track of such different possible worlds is an approach used in epistemic task planning. It would be an interesting future work that we already started investigating in our lab.

Another promising future work is to add more reasoning and inference processes to the planning process. We have already introduced the situation assessment process, updating agents' beliefs concerning their observable surroundings. However, adding logical reasoning and deduction would be interesting, like in the Dynamic Epistemic Logic in [Bolander 2017]. For instance, if it is known that A or B is true and the agent knows that B is false, then it

can be deduced that A is true. However, it would undoubtedly require using a knowledge representation different from state variables.

Inference processes based on causality could also be promising. For instance, if action *a2* can only be performed after *a1* and an agent notices that *a2* has been performed, then the agent can infer that *a1* has also been performed.

Finally, we can identify the minimal relevant information to communicate in the proposed approach. However, it would be an interesting future work to explore 'when' to communicate. Currently, the robot communicates at the last moment, just before the estimated false human beliefs have an effect. However, it could be wiser to plan the communication earlier, maybe even before the human leaves.

Task Planning for Concurrent and Compliant Joint Action

There are also a few limitations in my second contribution, each that can be relevant to address in future work. First, we can mention the "passive action pairs". Indeed, when planning the robot's policy, the cases where both the robot and the human are passive are explicitly modeled but not explored because, without any action, there is no modification to the state and, thus, no need to create a new state. These situations could happen during execution but are not considered in the current plan evaluation to generate the robot's policy. Nevertheless, in some rare situations, it can be relevant for the robot to remain passive even if the human is passive. This would insist on and non-verbally communicate that it is better for humans to act than robots.

Secondly, exploring and adding the possibility for the robot to take the initiative in the execution model would be interesting. A step starts with the robot waiting for a human signal before proceeding. The only exception is when the human cannot act, the robot directly starts acting. Nevertheless, when the robot's action does not depend on the human decision, it could be relevant to skip the initial synchronization and make the robot act since there is no possible conflict.

Moreover, the model's step-based aspect can be considered a limitation. It benefits the search but constrains the execution with many synchronizations. Investigating a proper execution controller based on the model that can supervise the robot's policy execution in a flexible manner, avoiding systematic synchronizations, would be interesting.

Eventually, we conducted a user study to validate the approach using an interactive simulator. Simulation and mouse control could bias the results obtained. Hence, a pertinent future work is to make the simulation more immersive and natural, e.g., using Virtual Reality (VR). Implementing the scheme on a real robot is also interesting but would require a significant amount of engineering work to avoid bias created by a slow-moving robot or perception errors.

Performances of the planning approach

The overall planning approach proposed in this thesis is based on an exhaustive, expensive search, which does not scale. Collaboration scenarios are not usually very long, making our approach sufficient and adequate for this context. Nevertheless, despite the short scenarios, their complexity can be significantly high. In Chapter 4, we proposed switching from the AND/OR tree search space to a Directed Acyclic Graph, significantly improving performance. Still, the approach does not scale. As a result, it would be relevant to identify pertinent heuristics to avoid exhaustive exploration. One idea is to investigate the risk estimation of subtasks. It would allow starting the execution of a non-fully refined plan. As humans, we tend to apply this reasoning. For instance, consider a collaborative scenario

where the task is to set the table. Due to the numerous types and number of objects to set, the possible orderings are considerable, making the exhaustive search infeasible. However, knowing that there are no risks of deadends or failure in this context, one could start with the closest objects without precisely knowing from the beginning the future sequence of actions to perform.

Simulating Social Navigating Agents

The InHuS was initially designed as a generic intelligent agent simulation architecture to solve and benchmark any collaborative task. However, implementing the entire generic architecture is very challenging, and as a first step, we limited the scheme to navigation. Hence, future work could be to integrate a complex task planner like HATP/EHDA inside InHuS as well as manipulation controllers.

The major limitation of InHuS is that it simulates only a single agent. This limitation has been addressed with the IMHuS system, which allows the designing and reproducing of challenging navigation scenarios with several humans. However, in IMHuS, agents lack individual decision-making processes like in InHuS. Another approach to extending InHuS to multi-human scenarios could be switching agent control dynamically. We could run two instances of InHuS to control the two closest humans to the robot, and we would control the other agents using reactive approaches. This would require reasonable computational power but necessitate carefully keeping track, pausing, and resuming each agent's goal.

Finally, this work was done at the beginning of my PhD. Since then, other recent works have proposed stimulating social interactive navigating agents to benchmark robot navigation. An interesting future work is to look deeper into these recent approaches and compare them with InHuS to potentially combine them and propose a more refined and efficient version of InHuS.

Unexplored features of HATP/EHDA

When working on my contributions based on HATP/EHDA, I reimplemented the original planner several times to address specific challenges. As a result, I sometimes simplified some aspects of the planner and even removed some features irrelevant to my focus. However, it is worth indicating that all original HATP/EHDA features are pertinent to collaborative scenarios and could be integrated back into my contributions' planner versions. I provide a few details on two aspects not present in my thesis examples, but they would be worth investigating further.

First, I would like to mention the "*Trigger*" mechanism implemented in HATP/EHDA. *Triggers* model agent's reactions to particular situations rather than goal-oriented actions. For instance, whatever the human is doing, when handed over an object or asked a question, the human is likely to interrupt their activity to grab the object or answer the question. *Triggers* are helpful to describe complex human action models, later used to estimate the next actions the human partner is likely to perform in a given state. For simplicity reasons, I did not implement *Triggers* in my experimental version of the planner, but this feature does not conflict with my contributions. Adding this feature would only require a small amount of additional work.

Additionally, the use cases of my contributions always considered that a shared goal had been established priorly between the agents. However, it is worth mentioning that HATP/EHDA can manage scenarios without shared goals. Their creation is handled by the given agent action models, where *Triggers* help model questions and answers. Scenarios without an initial shared goal were less illustrative for my examples, but the proposed

approaches are not limited to this case. Moreover, designing and integrating reasoning processes indicating, in a principled way, 'when' and 'how' to create a shared goal with a human partner is an interesting work in the future.

Part III

Appendix

Appendix A

User Study results

This appendix provides the raw results obtained through the User Study described and discussed in chapter 6.

A.1 Participants information

Anonymized information on the participants is shared in the tables A.1 and A.2. It includes the date of each participation, the age and gender of the participant, their opinion about robotics (1=negative, 5=positive), if they are from my laboratory (LAAS = yes, EXT = no), if they are familiar with robotics and task planning, and if they already interacted with a robot, if so which ones. The comments are given in the participants' language, hence, mostly in French, except for participants 10 and 16.

A.2 Scenario Ordering per Participant

The ordering in which each participant encountered each scenario is shown in table A.3. These orderings have been randomized to avoid order effect and follow a uniformed distribution.

A.3 Questionnaire answers

The participants' answers are provided in the 3 tables A.4, A.5, A.6. For each participant, the answer on a Likert scale from 1 to 7. Be careful, by default, 1 corresponds to the worst answer and 7 to the best. However, some questions have inverted scales in the questionnaire. The inverted questions are : Q2, Q5, Q6, Q8, and Q9. In the rest of the manuscript, especially in chapter 6, the inverted scales are inverted to have a uniform and more legible representation.

A.4 Execution metrics extracted

The extracted execution metrics are abbreviated corresponding to the table A.7. The metrics values are shown in ten tables from A.8 to A.17.

A.5 Participants comments

The participants' comments about the experiment are shown in the tables A.18, A.19, and A.20. Since these comments were given verbally they are here described as note-taking, in French which is the language spoken by a large majority of participants, and using as much as possible the participant's own words.

A.6 Scenario preference

The results obtained after asking participants which scenario they preferred the most and the least are shown in the table A.21.

A.7 PeRDITA questionnaire

The PeRDITA questionnaire filled by the participant after each scenario can be found in the next two following pages. Both the original French version and an English translation done by myself and based on other already translated versions.

PeRDITA Questionnaire FR

N° Participant : N° Scénario :

Afin d'étudier votre évaluation personnelle de chaque comportement du robot, vous allez répondre à un questionnaire. Vous allez devoir vous situer entre deux adjectifs en plaçant une croix dans la case qui se rapporte le plus à votre impression.

- Lisez attentivement les énoncés en gras avant de répondre.
- Vous pourrez modifier vos réponses plus tard.
- Il n'y a pas de bonne ou de mauvaise réponse. Répondez le plus sincèrement possible.

Rappelez les propriétés du scénario en cochant les cases correspondantes:

Régime d'exécution

Human-First ☐ Robot-First ☐

Objectif

Finir la tâche au plus vite ☐ Être libéré au plus vite ☐

Selon vous, le robot est plutôt :

Apathique	☐	☐	☐	☐	☐	☐	☐	Réactif
*Compétent	☐	☐	☐	☐	☐	☐	☐	Incompétent
Inintelligent	☐	☐	☐	☐	☐	☐	☐	Intelligent

Selon vous, globalement, l'interaction avec le robot a été :

Négative	☐	☐	☐	☐	☐	☐	☐	Positive
*Simple	☐	☐	☐	☐	☐	☐	☐	Compliquée
*Claire	☐	☐	☐	☐	☐	☐	☐	Ambiguë

Selon vous, la collaboration avec le robot pour réaliser la tâche a été :

Contraignante	☐	☐	☐	☐	☐	☐	☐	Adaptative
*Utile	☐	☐	☐	☐	☐	☐	☐	Inutile
Inefficace	☐	☐	☐	☐	☐	☐	☐	Efficace

Selon vous, le robot a choisi d'agir de manière :

*Adéquate	☐	☐	☐	☐	☐	☐	☐	Inadéquate
Gênante	☐	☐	☐	☐	☐	☐	☐	Accommodante
Imprévisible	☐	☐	☐	☐	☐	☐	☐	Prévisible

---

*Les items précédés d'un astérisque sont des items inversés.

PeRDITA Questionnaire EN

Participant N° : Scenario N° :

In order to study your personal evaluation of each robot behavior, you're going to answer a questionnaire. You'll be asked to place a cross between two adjectives in the box that most closely matches your impression.

- Read carefully the bold statements before answering.
- You will be able to modify your answers later.
- There are no right or wrong answers. Answer as truthfully as possible.

Recall the properties of the scenario by ticking the appropriate boxes:

Execution Regime		Your objective	
Human-First ☐	Robot-First ☐	Finish the task as soon as possible ☐	Be freed as soon as possible ☐

In your opinion, the robot is rather:

Apathetic	☐	☐	☐	☐	☐	☐	☐	Responsive
*Competent	☐	☐	☐	☐	☐	☐	☐	Incompetent
Unintelligent	☐	☐	☐	☐	☐	☐	☐	Intelligent

In your opinion, generally, the interaction with the robot was:

Negative	☐	☐	☐	☐	☐	☐	☐	Positive
*Simple	☐	☐	☐	☐	☐	☐	☐	Complicated
*Clear	☐	☐	☐	☐	☐	☐	☐	Ambiguous

In your opinion, the collaboration with the robot to perform the task was:

Restrictive	☐	☐	☐	☐	☐	☐	☐	Adaptive
*Useful	☐	☐	☐	☐	☐	☐	☐	Useless
Inefficient	☐	☐	☐	☐	☐	☐	☐	Efficient

In your opinion, the robot choices of action were:

*Appropriate	☐	☐	☐	☐	☐	☐	☐	Inappropriate
Annoying	☐	☐	☐	☐	☐	☐	☐	Accommodating
Unpredictable	☐	☐	☐	☐	☐	☐	☐	Predictable

---

*Items preceded by an asterisk are reversed items.

N° Participant	Date	Age	Genre	Qu'elle est votre vision de la robotique ?	Affectation	Etes vous familier avec la robotique et la plannification de tâche ?	Avez vous déjà interagi avec un robot ? Si oui quel genre ? (Drone, Robot aspirateur, Humanoïde, Jouet)
1	17/12/2023 14:55:45	21	Male	4	EXT	Non	Tous ceux cités
2	17/12/2023 15:48:14	23	Woman	5	EXT	Non	pas vraiment
3	18/12/2023 10:02:21	26	Male	4	LAAS	Oui	PR2, Pepper
4	18/12/2023 10:45:39	25	Male	4	LAAS	Oui	thermomix
5	19/12/2023 10:40:40	28	Male	3	LAAS	Oui	Humanoïde
6	19/12/2023 11:42:17	26	Male	4	LAAS	Non	Drone
7	19/12/2023 14:42:18	26	Male	3	LAAS	Non	Pepper
8	19/12/2023 15:31:04	24	Woman	4	LAAS	Non	robot aspirateur
9	19/12/2023 19:06:39	26	Male	5	EXT	Non	Robot aspirateur, jouets
10	08/01/2024 10:28:39	30	Woman	4	LAAS	Oui	no
11	09/01/2024 16:02:02	24	Male	4	LAAS	Non	Pepper, PR2, jouets
12	09/01/2024 16:32:36	27	Male	5	LAAS	Non	Oui, tous
13	10/01/2024 16:25:59	27	Woman	4	LAAS	Non	électroménager + chatbox

Table A.1: Information on the participants. (part 1/2)

N° Participant	Date	Age	Genre	Qu'elle est votre vision de la robotique ?	Affectation	Etes vous familier avec la robotique et la plannification de tâche ?	Avez vous déjà interagi avec un robot ? Si oui quel genre ? (Drone, Robot aspirateur, Humanoïde, Jouet)
14	11/01/2024 16:00:19	25	Male	5	LAAS	Non	brat manipulateur
15	12/01/2024 10:11:42	23	Woman	5	LAAS	Non	oui (drone, jouet)
16	12/01/2024 10:56:29	30	Male	4	LAAS	Non	Humanoid
17	17/01/2024 09:27:52	61	Male	4	LAAS	Non	Oui
18	17/01/2024 10:37:37	26	Male	5	LAAS	Oui	Drone, Pepper, PR2, Chien robot, Aspirateur,tondeuse, ...
19	17/01/2024 17:23:21	27	Woman	4	EXT	Non	Non pas vraiment
20	17/01/2024 17:58:32	29	Male	5	EXT	Non	Drone, robot aspirateur
21	23/01/2024 10:26:45	25	Male	4	LAAS	Oui	Humanoïde, Robot mobile
22	23/01/2024 18:03:08	27	Woman	4	EXT	Non	Jouet
23	25/01/2024 12:15:45	47	Woman	4	LAAS	Oui	PR2, robot mobile, bras manipulateurs
24	25/01/2024 13:47:39	26	Woman	4	LAAS	Non	PR2 + Jouets
25	26/01/2024 14:20:54	62	Male	5	LAAS	Oui	Bras interactif, humanoïdes, drones

Table A.2: Information on the participants. (part 2/2)

Participant N°	S1 pose	S2 pose	S3 pose	S4 pose	S5 pose	S6 pose
1	1	2	3	6	5	4
2	3	1	2	6	5	4
3	5	3	6	1	2	4
4	2	5	4	6	3	1
5	1	5	2	6	4	3
6	6	3	4	2	1	5
7	3	5	6	4	1	2
8	4	5	3	6	2	1
9	4	2	1	5	6	3
10	1	5	3	2	4	6
11	5	4	6	2	3	1
12	3	1	5	2	6	4
13	1	2	5	6	4	3
14	4	2	3	1	5	6
15	4	1	6	5	3	2
16	2	6	3	1	4	5
17	2	3	1	5	4	6
18	5	1	3	6	2	4
19	6	1	5	2	4	3
20	4	5	6	3	1	2
21	6	2	4	3	1	5
22	1	4	6	3	2	5
23	2	4	1	3	6	5
24	5	1	4	3	2	6
25	2	1	4	6	3	5

Table A.3: Ordering in which each participant encountered each scenario.

N°	S1 Q1	S1 Q2	S1 Q3	S1 Q4	S1 Q5	S1 Q6	S1 Q7	S1 Q8	S1 Q9	S1 Q10	S1 Q11	S1 Q12	S2 Q1	S2 Q2	S2 Q3	S2 Q4	S2 Q5	S2 Q6	S2 Q7	S2 Q8	S2 Q9	S2 Q10	S2 Q11	S2 Q12
1	6	2	5	6	2	2	6	2	6	3	6	6	6	3	3	6	2	2	5	2	5	5	2	2
2	6	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	1	7	1	7	1	7	7
3	6	2	5	6	2	2	7	2	7	1	7	7	4	1	6	7	1	1	7	1	7	1	7	3
4	7	1	1	7	2	1	7	2	7	1	7	7	6	1	1	7	1	3	7	1	7	1	7	5
5	6	3	3	6	3	3	6	3	6	2	5	6	6	2	5	6	3	3	6	3	6	3	5	6
6	4	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	2	7	1	7	1	7	6
7	6	2	2	6	2	2	5	3	5	2	5	5	6	2	2	6	2	2	6	2	6	2	4	4
8	7	1	7	7	1	1	7	1	7	1	7	7	7	2	7	7	3	2	6	3	6	2	6	5
9	7	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	2	1	6	1	7	1	7	7
10	6	2	5	6	2	3	6	2	5	2	6	6	6	2	6	6	2	2	6	2	6	2	6	6
11	7	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	1	7	1	7	1	7	7
12	6	2	6	5	2	2	6	3	6	2	6	6	3	3	3	4	5	6	3	3	2	5	3	4
13	5,5	6	4	6	1	1	6	3	5	2	7	7	7	1	4	7	1	1	7	2	7	1	7	6
14	6	2	3	5	3	2	4	3	6	2	6	4	6	2	6	6	3	3	6	2	6	3	6	5
15	7	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	1	6	1	7	1	7	7
16	6	3	6	6	2	2	7	6	6	2	6	6	6	2	6	6	2	2	7	2	7	2	5	5
17	6	2	5	6	3	2	6	6	6	2	5	5	5	2	5	6	2	3	4	1	6	2	4	6
18	7	2	6	6	2	2	7	1	6	1	7	6	6	2	5	6	2	2	6	2	6	1	6	7
19	6	1	7	6	1	1	7	1	7	1	7	7	7	1	6	6	4	4	6	2	6	1	7	7
20	7	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	1	7	1	7	1	7	6
21	6	2	6	6	2	2	6	2	6	2	6	6	7	2	6	6	2	1	6	2	6	2	7	6
22	7	1	4	7	1	1	7	1	7	1	7	7	7	2	4	6	2	2	5	1	7	1	7	6
23	6	2	5	6	2	2	6	2	6	2	6	6	6	2	5	6	2	2	5	3	5	2	6	6
24	7	1	7	7	1	1	7	1	7	1	7	7	7	1	7	7	1	1	7	1	7	7	7	4
25	6	2	6	5	2	2	6	2	6	2	6	6	6	2	6	5	2	2	3	2	5	2	6	2

Table A.4: Participants' answers to the twelve questions of the questionnaire (from Q1 to Q12), for each of the six scenarios (from S1 to S6), on a - Part 1

N°	S3 Q1	S3 Q2	S3 Q3	S3 Q4	S3 Q5	S3 Q6	S3 Q7	S3 Q8	S3 Q9	S3 Q10	S3 Q11	S3 Q12	S4 Q1	S4 Q2	S4 Q3	S4 Q4	S4 Q5	S4 Q6	S4 Q7	S4 Q8	S4 Q9	S4 Q10	S4 Q11	S4 Q12
1	6	3	3	6	2	2	5	2	5	5	2	2	6	2	6	6	2	2	6	1	6	2	6	5
2	6	1	7	7	1	1	7	1	7	1	7	7	7	6	3	6	6	7	2	3	3	6	3	6
3	6	2	6	5	5	2	5	2	6	1	6	7	4	7	2	2	1	2	1	6	1	7	2	1
4	7	1	1	6	1	1	7	1	7	2	7	7	7	7	1	3	1	1	3	2	4	6	3	7
5	6	2	6	5	2	2	6	4	7	2	7	7	4	6	1	1	6	7	1	7	1	7	1	1
6	7	1	6	7	2	2	7	1	6	1	7	7	7	6	2	1	7	2	1	4	2	6	1	2
7	4	2	6	6	2	2	4	5	6	2	6	6	6	2	6	2	6	6	2	6	2	6	2	6
8	7	1	7	7	1	1	7	1	7	1	7	7	5	3	5	5	2	5	3	5	3	2	5	3
9	7	1	7	7	2	1	6	1	7	1	6	7	7	6	2	5	1	1	3	4	2	6	2	1
10	5	3	5	6	2	3	5	3	5	5	4	4	3	5	3	3	2	4	3	4	2	6	2	4
11	7	1	7	7	1	1	7	1	7	1	7	7	7	1	6	6	1	1	5	1	5	2	7	6
12	6	3	5	5	1	2	4	3	5	1	6	7	2	3	4	2	3	5	3	6	3	5	1	1
13	7	1	6	7	1	1	7	2	7	1	7	7	7	2	3	6	1	1	3	4	5	5	6	3
14	5	3	3	6	2	3	5	3	6	3	6	5	5	5	5	5	2	3	5	3	4	4	4	6
15	7	2	6	5	3	4	7	1	7	2	7	7	7	4	6	3	4	4	2	1	5	2	3	6
16	6	3	6	6	2	2	7	6	6	2	6	6	5	5	5	5	3	3	4	4	3	5	3	3
17	5	6	3	3	5	4	3	3	1	6	3	4	5	6	2	3	5	5	3	5	6	6	3	2
18	7	2	6	7	1	2	6	1	6	2	6	6	2	6	2	3	6	7	1	7	2	7	1	1
19	7	1	7	7	1	1	7	1	7	1	7	7	6	7	1	1	7	5	1	7	2	7	1	1
20	7	1	7	7	1	1	7	1	7	1	7	7	4	5	3	3	5	5	3	3	4	3	5	4
21	6	2	6	6	2	2	6	2	6	2	6	6	5	4	3	3	5	5	2	5	2	5	3	3
22	7	1	4	7	1	1	7	1	7	1	7	7	7	6	3	1	6	7	2	6	3	7	1	1
23	6	2	6	6	2	2	6	2	6	2	6	6	6	3	5	5	3	3	6	2	6	2	5	6
24	7	1	6	7	1	1	7	1	7	1	7	6	7	3	2	6	5	1	5	4	4	5	3	4
25	6	2	2	7	2	2	6	2	6	2	7	7	6	6	2	2	3	4	2	3	2	6	2	3

Table A.5: Participants' answers to the 12 questions (Q) of the questionnaire, for each of the 6 scenarios (S) - Part 2

N°	S5 Q1	S5 Q2	S5 Q3	S5 Q4	S5 Q5	S5 Q6	S5 Q7	S5 Q8	S5 Q9	S5 Q10	S5 Q11	S5 Q12	S6 Q1	S6 Q2	S6 Q3	S6 Q4	S6 Q5	S6 Q6	S6 Q7	S6 Q8	S6 Q9	S6 Q10	S6 Q11	S6 Q12
1	5	6	2	4	2	2	3	5	3	7	3	3	6	3	2	5	2	2	3	6	3	6	1	1
2	7	1	7	1	1	1	7	1	7	1	7	7	7	5	4	2	7	7	4	2	5	5	3	5
3	6	2	4	2	2	1	5	3	7	1	7	7	4	5	3	5	4	2	1	4	1	5	3	5
4	7	1	1	6	1	2	7	1	6	1	7	7	7	5	1	5	1	2	3	4	6	4	6	7
5	4	3	5	4	1	2	4	4	4	3	4	6	4	6	3	2	4	5	1	6	2	6	1	2
6	7	1	7	7	1	2	6	1	6	3	5	5	7	6	4	2	6	4	1	3	4	6	1	4
7	6	2	6	5	4	5	4	3	6	5	4	4	6	2	6	2	6	6	3	3	5	5	2	2
8	7	1	6	7	1	1	7	1	7	2	7	6	6	3	5	5	5	6	3	3	2	6	3	6
9	7	1	6	7	1	2	7	1	6	2	7	6	5	2	3	6	2	5	7	1	3	5	3	3
10	6	1	7	6	2	2	6	1	6	3	5	5	4	5	2	3	2	4	2	5	2	5	2	4
11	7	1	7	7	1	1	7	1	7	2	7	6	7	1	6	7	1	1	3	2	3	5	3	5
12	5	3	2	3	1	5	7	2	5	4	3	5	5	6	3	1	2	6	1	6	2	7	2	3
13	5	2	4	6	1	1	6	1	6	2	7	5	5	4	4	5	1	1	5	4	5	5	4	5
14	6	2	2	6	2	2	6	2	6	2	6	5	5	5	5	4	2	2	5	5	3	4	4	4
15	7	2	5	6	3	4	7	2	6	3	7	4	7	3	3	3	2	4	2	5	2	5	2	3
16	7	2	7	6	2	2	5	2	5	2	6	6	4	4	4	5	3	4	4	5	2	5	3	5
17	5	4	5	4	4	5	4	2	3	5	4	5	5	3	5	2	6	4	4	3	2	6	2	3
18	6	3	6	7	3	3	6	1	6	3	5	6	6	2	5	6	2	2	5	2	2	6	2	6
19	6	4	5	5	3	3	4	2	5	6	4	4	5	2	5	4	4	5	5	2	6	4	2	6
20	7	1	7	7	1	1	7	1	6	1	7	6	6	2	6	4	3	2	6	2	6	3	6	6
21	6	2	6	6	2	2	6	2	6	2	6	6	6	2	6	4	3	2	5	3	4	3	3	4
22	7	1	4	7	1	1	6	1	7	2	7	6	7	2	6	5	2	2	5	3	4	3	4	1
23	6	3	5	6	3	3	4	2	5	3	7	5	5	3	5	5	3	4	5	2	5	3	6	5
24	7	2	5	7	1	1	7	1	6	2	7	6	7	2	2	5	2	5	1	7	1	7	1	4
25	6	2	2	7	2	2	7	1	7	2	6	6	6	2	2	4	4	3	2	3	2	6	2	2

Table A.6: Participants' answers to the 12 questions (Q) of the questionnaire, for each of the 6 scenarios (S) - Part 3

M1	task_completion_time	M17	h_action_time_average
M2	number_steps	M18	h_action_time_sd
M3	nb_h_optimal_action	M19	h_action_time_max
M4	ratio_h_optimal_action	M20	h_action_time_min
M5	decision_time_total	M21	r_action_nb
M6	decision_time_average	M22	r_action_time_total
M7	decision_time_sd	M23	r_action_time_average
M8	decision_time_max	M24	r_action_time_sd
M9	decision_time_min	M25	r_action_time_max
M10	wait_ns_total	M26	r_action_time_min
M11	wait_ns_average	M27	time_human_free
M12	wait_ns_sd	M28	plan_mvt_total
M13	wait_ns_max	M29	plan_mvt_min
M14	wait_ns_min	M30	plan_mvt_max
M15	h_action_nb	M31	plan_mvt_average
M16	h_action_time_total	M32	plan_mvt_sd

Table A.7: Execution metrics abbreviations.

N°	S1 M1	S1 M2	S1 M3	S1 M4	S1 M5	S1 M6	S1 M7	S1 M8	S1 M9	S1 M10	S1 M11	S1 M12	S1 M13	S1 M14	S1 M15	S1 M16	S1 M17	S1 M18	S1 M19
1	56,6	10,0	10,0	100,0	8,2	1,0	0,5	1,8	0,3	12,5	1,6	0,9	3,2	0,6	8,0	26,2	3,3	0,6	4,4
2	58,7	10,0	10,0	100,0	9,7	1,2	0,5	2,0	0,6	14,3	1,8	1,2	3,9	0,7	8,0	25,6	3,2	0,6	4,2
3	61,6	10,0	9,0	90,0	10,3	1,0	0,8	2,9	0,4	14,9	1,9	1,2	4,0	0,7	8,0	26,6	3,3	0,6	4,3
4	56,7	10,0	9,0	90,0	3,8	0,4	0,4	1,2	0,0	17,6	2,2	1,5	5,3	0,6	8,0	26,1	3,3	0,6	4,3
5	55,3	10,0	10,0	100,0	2,9	0,4	0,5	1,2	0,0	17,7	2,2	1,4	4,3	0,6	8,0	26,3	3,3	0,6	4,3
6	52,6	10,0	10,0	100,0	3,6	0,4	0,5	1,7	0,0	14,4	1,8	1,0	3,8	0,7	8,0	25,9	3,2	0,6	4,3
7	83,5	14,0	10,0	71,4	12,0	0,9	0,7	2,4	0,1	11,8	2,0	1,1	4,0	0,6	6,0	18,7	3,1	0,5	4,2
8	60,6	10,0	9,0	90,0	2,8	0,3	0,6	1,9	0,0	14,5	1,8	1,2	4,2	0,7	8,0	26,4	3,3	0,6	4,3
9	61,3	10,0	10,0	100,0	7,3	0,9	0,5	2,0	0,6	18,3	2,3	1,3	4,3	0,7	8,0	25,9	3,2	0,6	4,3
10	62,4	10,0	10,0	100,0	14,1	1,8	2,3	6,8	0,0	13,9	1,7	1,0	3,6	0,7	8,0	26,0	3,3	0,6	4,3
11	58,6	10,0	10,0	100,0	4,6	0,5	0,5	1,8	0,1	20,2	2,5	1,8	6,5	0,6	8,0	25,8	3,2	0,6	4,3
12	58,3	10,0	10,0	100,0	3,6	0,5	1,0	3,2	0,0	18,9	2,4	1,7	5,7	0,7	8,0	26,2	3,3	0,6	4,3
13	61,4	12,0	10,0	83,3	1,6	0,1	0,2	0,4	0,0	17,9	1,8	1,1	4,1	0,7	10,0	32,7	3,3	0,6	4,3
14	57,5	10,0	10,0	100,0	3,5	0,4	0,6	1,7	0,0	20,7	2,6	1,8	6,8	0,6	8,0	26,0	3,2	0,6	4,3
15	62,3	10,0	9,0	90,0	2,7	0,3	0,6	1,9	0,0	18,3	2,3	1,5	4,8	0,6	8,0	26,4	3,3	0,6	4,3
16	60,2	10,0	10,0	100,0	4,5	0,6	1,0	3,1	0,0	19,4	2,4	1,5	4,7	0,6	8,0	25,9	3,2	0,6	4,3
17	59,3	10,0	10,0	100,0	10,2	1,3	1,8	5,8	0,0	13,0	1,6	0,9	3,0	0,6	8,0	25,8	3,2	0,6	4,2
18	56,5	10,0	10,0	100,0	2,7	0,3	0,2	0,8	0,0	18,9	2,4	1,7	5,6	0,6	8,0	26,1	3,3	0,6	4,3
19	59,7	10,0	9,0	90,0	4,9	0,5	1,0	3,0	0,0	14,2	1,8	1,5	5,1	0,6	8,0	26,6	3,3	0,6	4,3
20	54,8	10,0	10,0	100,0	2,8	0,3	0,2	0,7	0,0	17,2	2,2	1,7	5,1	0,6	8,0	26,2	3,3	0,6	4,4
21	55,2	10,0	10,0	100,0	1,6	0,2	0,2	0,5	0,0	18,6	2,3	1,4	4,9	0,6	8,0	25,8	3,2	0,6	4,3
22	60,3	10,0	10,0	100,0	8,7	1,1	0,5	1,9	0,5	16,1	2,0	1,0	4,0	0,6	8,0	26,1	3,3	0,6	4,3
23	67,4	12,0	10,0	83,3	6,8	0,7	1,1	3,5	0,0	19,5	2,0	1,4	4,6	0,7	10,0	32,6	3,3	0,6	4,3
24	56,5	10,0	10,0	100,0	9,8	1,1	0,5	2,0	0,4	11,6	1,5	0,7	2,5	0,7	8,0	26,0	3,3	0,6	4,3
25	58,5	10,0	10,0	100,0	2,7	0,3	0,5	1,8	0,0	18,9	2,4	1,4	5,0	0,6	8,0	25,8	3,2	0,6	4,2

Table A.8: Execution metrics - Part 1

N°	S1 M20	S1 M21	S1 M22	S1 M23	S1 M24	S1 M25	S1 M26	S1 M27	S1 M28	S1 M29	S1 M30	S1 M31	S1 M32	S2 M1	S2 M2	S2 M3	S2 M4	S2 M5	S2 M6
1	2,6	8,0	28,2	3,5	0,6	4,5	2,6	22,1	5,7	0,0	1,2	0,6	0,3	59,5	10,0	10,0	100,0	18,3	2,3
2	2,5	8,0	31,3	3,9	0,9	5,8	2,7	22,7	3,2	0,0	0,7	0,3	0,2	54,7	10,0	10,0	100,0	7,8	1,0
3	2,6	8,0	29,8	3,7	0,9	6,1	3,0	61,6	5,8	0,1	1,8	0,7	0,4	53,9	10,0	10,0	100,0	11,0	1,4
4	2,5	8,0	33,4	4,2	1,2	7,0	2,9	56,7	4,1	0,2	0,7	0,5	0,2	50,0	10,0	10,0	100,0	3,8	0,4
5	2,7	8,0	34,0	4,2	1,4	6,7	3,1	18,8	3,9	0,0	0,7	0,4	0,3	51,1	10,0	10,0	100,0	1,6	0,2
6	2,7	8,0	28,8	3,6	0,8	5,2	2,8	18,2	5,5	0,0	0,8	0,6	0,3	50,0	10,0	10,0	100,0	2,0	0,3
7	2,6	12,0	53,2	4,4	1,5	7,9	3,1	21,0	6,1	0,0	0,7	0,4	0,3	54,4	10,0	10,0	100,0	6,5	0,8
8	2,6	8,0	32,5	4,1	1,2	5,8	2,7	60,6	5,0	0,3	1,0	0,6	0,2	59,4	11,0	10,0	90,9	6,2	0,8
9	2,6	8,0	35,4	4,4	1,0	5,9	3,0	22,9	4,2	0,0	0,8	0,4	0,3	52,7	10,0	10,0	100,0	3,8	0,5
10	2,6	8,0	30,6	3,8	0,8	5,0	2,6	25,2	4,5	0,0	0,7	0,5	0,3	64,0	12,0	10,0	83,3	17,1	2,1
11	2,6	8,0	34,6	4,3	1,3	6,9	3,1	22,5	5,4	0,0	1,1	0,5	0,3	53,1	10,0	10,0	100,0	9,7	1,2
12	2,6	8,0	35,3	4,4	1,2	6,2	3,0	21,0	4,9	0,0	1,2	0,5	0,4	54,5	10,0	10,0	100,0	12,3	1,5
13	2,6	8,0	32,2	4,0	0,8	5,6	3,3	17,5	4,9	0,0	0,8	0,4	0,3	57,0	10,0	10,0	100,0	8,2	1,0
14	2,7	8,0	34,4	4,3	1,2	6,2	3,0	24,4	5,0	0,0	2,3	0,5	0,6	63,0	11,0	10,0	90,9	12,7	1,6
15	2,7	8,0	33,9	4,2	1,2	6,5	3,0	62,3	5,2	0,5	0,8	0,6	0,1	58,3	10,0	10,0	100,0	12,0	1,5
16	2,6	8,0	36,2	4,5	1,2	6,2	2,8	19,8	5,0	0,0	0,7	0,5	0,3	60,9	11,0	10,0	90,9	11,3	1,4
17	2,6	8,0	30,6	3,8	1,1	5,5	2,3	19,2	3,7	0,0	0,7	0,4	0,3	75,8	14,0	10,0	71,4	5,4	0,9
18	2,6	8,0	34,3	4,3	1,5	7,4	3,1	18,4	5,0	0,0	0,8	0,5	0,3	52,3	10,0	10,0	100,0	5,8	0,6
19	2,6	8,0	29,9	3,7	1,2	6,7	2,9	59,7	4,3	0,1	0,7	0,5	0,2	54,0	11,0	10,0	90,9	0,9	0,1
20	2,7	8,0	33,0	4,1	1,3	6,7	3,1	20,3	4,4	0,0	0,8	0,4	0,3	56,7	10,0	10,0	100,0	10,9	1,4
21	2,6	8,0	33,9	4,2	1,0	6,0	2,7	20,5	5,0	0,0	0,9	0,5	0,3	44,7	10,0	10,0	100,0	1,7	0,2
22	2,7	8,0	31,3	3,9	0,9	5,8	2,8	23,2	5,7	0,0	0,9	0,6	0,3	51,8	10,0	10,0	100,0	9,0	1,1
23	2,6	8,0	32,2	4,0	0,9	5,4	3,1	24,5	5,3	0,0	0,7	0,4	0,3	64,6	12,0	10,0	83,3	6,9	1,0
24	2,6	8,0	28,3	3,5	0,4	4,1	2,8	20,9	4,1	0,0	0,7	0,4	0,3	54,2	10,0	10,0	100,0	10,2	1,3
25	2,6	8,0	36,5	4,6	1,3	6,6	2,7	19,9	4,6	0,0	0,9	0,5	0,3	65,2	12,0	10,0	83,3	11,5	1,4

Table A.9: Execution metrics - Part 2

N°	S2 M7	S2 M8	S2 M9	S2 M10	S2 M11	S2 M12	S2 M13	S2 M14	S2 M15	S2 M16	S2 M17	S2 M18	S2 M19	S2 M20	S2 M21	S2 M22	S2 M23	S2 M24	S2 M25
1	1,4	5,4	0,9	5,9	0,7	0,1	1,1	0,6	8,0	26,0	3,3	0,6	4,3	2,6	8,0	31,2	3,9	0,9	5,9
2	0,4	1,9	0,4	8,3	1,0	0,4	1,6	0,6	8,0	26,1	3,3	0,6	4,3	2,6	8,0	31,2	3,9	0,6	4,9
3	0,8	2,8	0,2	8,3	1,0	0,5	2,0	0,6	8,0	26,2	3,3	0,6	4,3	2,6	8,0	30,3	3,8	0,8	5,3
4	0,5	1,5	0,0	11,7	1,5	1,1	3,6	0,6	8,0	26,2	3,3	0,6	4,2	2,6	8,0	30,1	3,8	1,2	6,7
5	0,2	0,6	0,0	14,8	1,8	1,5	5,3	0,6	8,0	26,3	3,3	0,6	4,3	2,7	8,0	32,1	4,0	1,2	6,9
6	0,2	0,6	0,0	12,9	1,6	1,2	4,3	0,7	8,0	26,1	3,3	0,6	4,3	2,6	8,0	32,6	4,1	0,9	5,7
7	1,5	4,7	0,0	10,8	1,4	1,0	3,8	0,6	8,0	26,3	3,3	0,6	4,3	2,6	8,0	31,9	4,0	0,9	5,8
8	1,2	3,8	0,0	12,4	1,6	1,5	4,8	0,6	8,0	26,3	3,3	0,6	4,3	2,7	8,0	35,4	4,4	1,2	6,4
9	0,5	1,6	0,0	11,1	1,4	0,8	2,5	0,6	8,0	26,2	3,3	0,6	4,3	2,6	8,0	33,0	4,1	1,2	6,3
10	1,9	5,9	0,0	8,2	1,0	0,6	2,4	0,6	8,0	26,1	3,3	0,6	4,3	2,6	8,0	30,2	3,8	1,1	5,9
11	0,9	2,9	0,0	9,7	1,2	0,8	2,6	0,6	8,0	25,9	3,2	0,6	4,3	2,6	8,0	30,0	3,8	1,1	6,3
12	1,1	3,9	0,0	6,2	0,8	0,2	1,2	0,6	8,0	26,1	3,3	0,7	4,4	2,6	8,0	30,9	3,9	0,8	5,4
13	1,2	3,7	0,0	14,4	1,8	1,3	4,7	0,6	8,0	26,0	3,3	0,6	4,3	2,6	8,0	34,8	4,3	2,0	8,7
14	1,9	4,8	0,0	9,7	1,2	0,6	2,4	0,6	8,0	26,0	3,3	0,6	4,4	2,6	8,0	31,4	3,9	0,6	4,8
15	1,5	4,7	0,0	12,2	1,5	1,6	5,6	0,6	8,0	26,0	3,2	0,6	4,3	2,7	8,0	34,5	4,3	1,2	6,3
16	1,3	3,5	0,0	7,6	0,9	0,7	2,8	0,6	8,0	26,3	3,3	0,7	4,4	2,7	8,0	34,3	4,3	1,1	6,2
17	1,8	5,0	0,0	13,7	2,3	1,0	3,6	0,7	6,0	19,7	3,3	0,7	4,3	2,6	12,0	48,6	4,1	1,0	6,0
18	0,8	2,5	0,0	11,5	1,4	0,9	3,3	0,6	8,0	26,5	3,3	0,6	4,4	2,6	8,0	31,1	3,9	0,7	4,7
19	0,2	0,6	0,0	8,9	1,1	0,6	2,2	0,6	8,0	26,3	3,3	0,6	4,3	2,6	8,0	28,1	3,5	0,3	3,9
20	1,4	4,7	0,0	10,6	1,3	1,4	5,0	0,6	8,0	26,4	3,3	0,6	4,3	2,7	8,0	33,1	4,1	0,9	5,8
21	0,3	0,9	0,0	9,0	1,1	0,5	1,9	0,6	8,0	26,1	3,3	0,6	4,3	2,6	8,0	26,4	3,3	0,3	3,7
22	1,4	4,2	0,0	8,0	1,0	0,6	2,4	0,6	8,0	26,3	3,3	0,6	4,3	2,7	8,0	32,1	4,0	0,8	5,5
23	1,0	2,9	0,0	6,6	1,1	0,6	2,1	0,6	6,0	20,4	3,4	0,7	4,4	2,6	10,0	40,5	4,1	0,6	4,8
24	0,9	2,9	0,4	8,1	1,0	0,8	3,2	0,6	8,0	26,3	3,3	0,6	4,3	2,6	8,0	31,4	3,9	0,9	5,4
25	1,6	4,7	0,0	8,5	1,1	0,8	3,1	0,6	8,0	26,0	3,3	0,6	4,4	2,6	8,0	36,5	4,6	1,3	7,2

Table A.10: Execution metrics - Part 3

N°	S2 M26	S2 M27	S2 M28	S2 M29	S2 M30	S2 M31	S2 M32	S3 M1	S3 M2	S3 M3	S3 M4	S3 M5	S3 M6	S3 M7	S3 M8	S3 M9	S3 M10	S3 M11	S3 M12
1	2,8	26,0	5,1	0,2	1,1	0,6	0,3	54,8	10,0	10,0	100,0	4,9	0,6	0,2	1,0	0,4	15,4	1,9	1,4
2	2,9	54,7	5,2	0,5	0,7	0,6	0,1	55,0	10,0	10,0	100,0	6,3	0,8	0,5	2,0	0,0	14,6	1,8	0,9
3	2,8	22,9	4,7	0,2	0,7	0,6	0,2	88,1	12,0	10,0	83,3	9,0	1,1	0,6	2,1	0,1	16,6	2,1	1,6
4	2,7	50,0	4,1	0,3	0,7	0,5	0,2	60,7	10,0	9,0	90,0	4,6	0,5	0,7	2,5	0,0	20,3	2,5	1,7
5	3,0	18,8	5,1	0,3	1,2	0,6	0,3	60,4	12,0	10,0	83,3	0,7	0,1	0,1	0,5	0,0	19,2	1,9	1,1
6	3,0	19,9	4,7	0,2	1,0	0,6	0,2	83,3	12,0	10,0	83,3	6,4	0,7	0,6	2,2	0,0	19,5	2,4	1,6
7	3,0	54,4	5,8	0,2	1,5	0,7	0,3	84,3	12,0	10,0	83,3	20,5	1,9	3,0	10,8	0,1	22,4	2,2	1,6
8	2,9	23,0	4,9	0,3	0,9	0,6	0,2	63,9	12,0	10,0	83,3	1,9	0,2	0,4	1,2	0,0	21,1	2,1	1,2
9	2,6	19,1	5,2	0,4	0,8	0,7	0,1	63,5	10,0	10,0	100,0	6,8	0,9	0,5	2,0	0,4	22,2	2,8	1,8
10	2,7	24,2	4,5	0,4	0,7	0,6	0,1	80,1	14,0	11,0	78,6	2,4	0,3	0,3	1,0	0,0	17,5	2,2	2,0
11	2,8	20,6	5,3	0,3	1,0	0,7	0,2	58,6	10,0	10,0	100,0	3,0	0,3	0,2	0,5	0,0	21,5	2,7	1,9
12	3,1	19,6	4,8	0,2	0,7	0,6	0,2	62,8	12,0	10,0	83,3	2,9	0,3	0,4	1,2	0,0	17,1	1,7	1,0
13	2,9	21,4	5,8	0,3	1,4	0,7	0,3	71,9	12,0	11,0	91,7	1,2	0,1	0,2	0,5	0,0	20,2	2,5	1,7
14	2,8	25,6	5,1	0,4	0,9	0,6	0,1	71,1	10,0	10,0	100,0	15,2	1,9	1,4	3,8	0,1	21,6	2,7	1,8
15	2,9	25,5	5,4	0,1	1,3	0,7	0,4	74,1	12,0	11,0	91,7	3,8	0,6	0,8	1,7	0,0	14,8	2,5	1,9
16	2,7	21,6	6,4	0,2	1,5	0,8	0,4	67,8	12,0	10,0	83,3	5,1	0,6	1,1	3,1	0,0	12,8	1,6	1,4
17	2,7	32,0	8,3	0,3	1,3	0,7	0,2	116,2	17,0	14,0	82,4	0,8	0,1	0,1	0,4	0,0	11,0	1,8	1,2
18	3,0	52,3	4,7	0,2	0,7	0,6	0,2	53,5	10,0	10,0	100,0	1,7	0,2	0,3	0,9	0,0	18,1	2,3	1,1
19	3,0	54,0	4,3	0,2	0,7	0,5	0,2	55,5	10,0	10,0	100,0	1,3	0,2	0,2	0,5	0,0	17,9	2,2	1,5
20	2,9	23,5	5,2	0,2	1,2	0,7	0,3	55,5	10,0	10,0	100,0	2,6	0,3	0,4	1,3	0,0	17,7	2,2	1,7
21	2,7	16,6	4,7	0,3	0,7	0,6	0,1	51,7	10,0	10,0	100,0	1,6	0,2	0,2	0,6	0,0	15,2	1,9	1,0
22	2,9	18,1	3,8	0,2	0,8	0,5	0,2	51,8	10,0	10,0	100,0	2,2	0,3	0,2	0,8	0,0	16,1	2,0	1,4
23	2,8	64,6	6,2	0,1	1,4	0,6	0,3	61,1	12,0	10,0	83,3	0,4	0,0	0,0	0,1	0,0	19,5	2,0	1,2
24	2,8	23,0	4,1	0,2	0,7	0,5	0,2	65,7	10,0	10,0	100,0	8,7	1,1	1,1	3,7	0,2	19,7	2,5	1,4
25	3,0	20,8	4,3	0,3	0,8	0,5	0,2	53,7	10,0	10,0	100,0	1,7	0,2	0,4	1,3	0,0	16,8	2,1	1,1

Table A.11: Execution metrics - Part 4

N°	S3 M13	S3 M14	S3 M15	S3 M16	S3 M17	S3 M18	S3 M19	S3 M20	S3 M21	S3 M22	S3 M23	S3 M24	S3 M25	S3 M26	S3 M27	S3 M28	S3 M29	S3 M30	S3 M31
1	4,8	0,6	8,0	25,9	3,2	0,6	4,3	2,6	8,0	30,1	3,8	0,9	5,8	3,1	21,6	4,3	0,0	0,7	0,4
2	3,1	0,6	8,0	25,8	3,2	0,6	4,3	2,6	8,0	27,6	3,4	0,6	4,4	2,6	21,2	5,8	0,0	1,1	0,6
3	5,6	0,7	8,0	26,0	3,3	0,5	4,2	2,6	10,0	40,5	4,1	1,2	6,5	2,2	88,1	5,0	0,2	0,7	0,5
4	6,2	0,7	8,0	26,4	3,3	0,6	4,3	2,6	8,0	33,4	4,2	1,4	7,1	3,0	60,7	6,2	0,3	1,9	0,8
5	3,6	0,6	10,0	32,8	3,3	0,6	4,3	2,6	8,0	31,7	4,0	0,7	4,9	2,8	19,2	4,0	0,0	0,7	0,3
6	6,2	0,7	8,0	26,1	3,3	0,5	4,2	2,6	10,0	41,8	4,2	1,2	7,6	3,2	83,3	5,5	0,2	0,7	0,6
7	5,0	0,6	10,0	32,5	3,3	0,6	4,2	2,7	10,0	45,7	4,6	1,3	7,5	2,9	32,9	5,8	0,0	0,8	0,5
8	3,5	0,7	10,0	32,9	3,3	0,6	4,3	2,6	8,0	33,2	4,2	0,8	5,5	3,0	21,2	4,5	0,0	0,7	0,4
9	5,8	0,7	8,0	26,1	3,3	0,6	4,3	2,6	8,0	36,3	4,5	1,5	7,4	3,1	25,4	4,7	0,0	1,1	0,5
10	6,5	0,6	8,0	25,8	3,2	0,5	4,2	2,7	10,0	44,2	4,4	1,5	8,3	2,8	32,8	5,3	0,0	1,0	0,4
11	6,1	0,6	8,0	25,9	3,2	0,6	4,3	2,6	8,0	36,4	4,5	1,5	7,7	3,0	21,6	4,0	0,0	1,0	0,4
12	3,4	0,6	10,0	32,9	3,3	0,6	4,3	2,6	8,0	31,4	3,9	0,4	4,6	3,3	19,2	4,4	0,0	0,7	0,4
13	6,1	0,7	8,0	25,7	3,2	0,5	4,2	2,7	10,0	42,8	4,3	1,5	7,4	2,6	20,2	6,1	0,0	0,7	0,5
14	6,3	0,7	8,0	26,0	3,2	0,6	4,3	2,6	8,0	35,0	4,4	1,2	7,0	3,1	25,8	4,9	0,0	1,0	0,5
15	6,2	0,7	6,0	19,2	3,2	0,5	4,3	2,7	10,0	45,2	4,5	1,4	7,5	2,9	36,0	5,3	0,0	0,7	0,4
16	5,0	0,6	8,0	26,6	3,3	0,6	4,3	2,6	8,0	31,1	3,9	1,0	6,1	2,9	21,4	4,1	0,0	0,7	0,3
17	3,7	0,6	6,0	19,8	3,3	0,7	4,3	2,6	14,0	61,0	4,4	1,2	7,1	2,7	31,5	9,2	0,0	1,2	0,5
18	3,9	0,6	8,0	26,1	3,3	0,6	4,2	2,6	8,0	31,5	3,9	0,8	5,2	2,7	21,1	4,8	0,0	0,8	0,5
19	5,2	0,6	8,0	26,2	3,3	0,6	4,3	2,6	8,0	34,0	4,2	1,4	6,6	2,8	19,6	4,3	0,0	0,7	0,4
20	6,1	0,7	8,0	26,5	3,3	0,6	4,4	2,7	8,0	32,3	4,0	1,3	7,2	2,9	19,6	5,0	0,0	0,7	0,5
21	3,7	0,6	8,0	26,1	3,3	0,6	4,3	2,6	8,0	29,1	3,6	0,5	4,7	3,2	19,0	5,0	0,0	0,7	0,5
22	5,2	0,6	8,0	26,2	3,3	0,6	4,3	2,6	8,0	28,9	3,6	1,2	6,6	2,6	18,1	4,9	0,0	0,8	0,5
23	4,1	0,6	10,0	33,0	3,3	0,6	4,3	2,6	8,0	31,8	4,0	0,9	6,0	3,0	18,6	4,7	0,0	0,7	0,4
24	4,5	0,7	8,0	26,2	3,3	0,6	4,4	2,6	8,0	36,0	4,5	1,0	5,5	2,9	27,7	4,9	0,0	1,2	0,5
25	4,3	0,7	8,0	25,8	3,2	0,6	4,3	2,6	8,0	31,2	3,9	0,5	5,3	3,5	18,4	4,8	0,0	0,7	0,5

Table A.12: Execution metrics - Part 5

N°	S3 M32	S4 M1	S4 M2	S4 M3	S4 M4	S4 M5	S4 M6	S4 M7	S4 M8	S4 M9	S4 M10	S4 M11	S4 M12	S4 M13	S4 M14	S4 M15	S4 M16	S4 M17	S4 M18
1	0,2	91,7	18,0	16,0	88,9	0,6	0,3	0,0	0,3	0,3	1,3	0,7	0,0	0,7	0,6	2,0	6,0	3,0	0,3
2	0,3	80,7	16,0	16,0	100,0	0,7	0,2	0,1	0,3	0,0	6,0	1,5	1,4	4,0	0,7	4,0	12,9	3,2	0,7
3	0,2	82,4	16,0	16,0	100,0	3,7	0,9	0,7	2,1	0,1	7,2	1,8	2,0	5,2	0,6	4,0	13,1	3,3	0,6
4	0,4	78,1	16,0	16,0	100,0	1,8	0,4	0,4	0,9	0,0	4,4	1,1	0,7	2,4	0,7	4,0	13,4	3,3	0,7
5	0,3	78,2	16,0	16,0	100,0	2,9	0,7	0,7	1,8	0,0	5,2	1,3	1,1	3,1	0,6	4,0	13,0	3,3	0,6
6	0,2	79,0	16,0	16,0	100,0	0,4	0,1	0,1	0,3	0,0	5,6	1,4	1,3	3,6	0,6	4,0	13,0	3,3	0,7
7	0,3	86,3	16,0	16,0	100,0	4,0	1,0	1,2	2,9	0,0	3,4	0,8	0,3	1,4	0,6	4,0	13,0	3,3	0,7
8	0,3	77,8	16,0	16,0	100,0	2,9	0,7	1,1	2,6	0,0	8,8	2,2	2,2	6,0	0,6	4,0	13,1	3,3	0,6
9	0,3	88,0	16,0	16,0	100,0	1,6	0,4	0,7	1,6	0,0	8,0	2,0	2,2	5,7	0,6	4,0	13,2	3,3	0,7
10	0,3	79,7	14,0	13,0	92,9	12,7	2,1	0,7	3,3	1,1	5,9	1,0	0,7	2,6	0,6	6,0	19,6	3,3	0,6
11	0,3	87,2	16,0	13,0	81,3	4,5	1,1	1,1	2,6	0,0	5,7	1,4	1,3	3,6	0,7	4,0	12,7	3,2	0,4
12	0,3	82,6	16,0	15,0	93,8	1,4	0,3	0,3	0,8	0,0	6,3	1,6	1,3	3,8	0,6	4,0	13,0	3,3	0,7
13	0,3	77,4	16,0	16,0	100,0	0,1	0,0	0,0	0,0	0,0	7,2	1,8	1,8	4,9	0,6	4,0	13,0	3,2	0,7
14	0,4	86,3	17,0	16,0	94,1	5,2	1,3	1,2	2,7	0,0	3,2	0,8	0,2	1,2	0,7	4,0	13,2	3,3	0,7
15	0,3	85,7	16,0	16,0	100,0	0,1	0,0	0,0	0,0	0,0	6,7	1,7	1,2	3,7	0,6	4,0	13,1	3,3	0,6
16	0,3	88,1	16,0	16,0	100,0	1,0	0,3	0,4	1,0	0,0	5,4	1,4	1,1	3,3	0,6	4,0	13,1	3,3	0,7
17	0,4	86,9	16,0	16,0	100,0	0,8	0,2	0,3	0,6	0,0	6,7	1,7	1,2	3,5	0,6	4,0	13,0	3,3	0,7
18	0,3	76,5	16,0	16,0	100,0	0,7	0,2	0,3	0,6	0,0	4,7	1,2	0,8	2,6	0,6	4,0	12,9	3,2	0,7
19	0,2	80,6	16,0	16,0	100,0	1,3	0,3	0,5	1,3	0,0	4,4	1,1	0,6	2,1	0,6	4,0	13,2	3,3	0,7
20	0,3	86,0	16,0	16,0	100,0	5,9	1,5	1,3	3,0	0,0	4,0	1,0	0,6	2,0	0,6	4,0	13,2	3,3	0,6
21	0,3	78,6	16,0	16,0	100,0	0,8	0,2	0,3	0,8	0,0	4,2	1,1	0,7	2,3	0,6	4,0	13,1	3,3	0,6
22	0,3	84,7	16,0	16,0	100,0	5,9	1,5	0,8	2,4	0,1	6,7	1,7	1,8	4,8	0,6	4,0	13,1	3,3	0,7
23	0,3	83,9	16,0	16,0	100,0	0,1	0,0	0,0	0,0	0,0	7,0	1,8	1,1	3,2	0,6	4,0	13,1	3,3	0,6
24	0,4	76,4	16,0	16,0	100,0	3,3	0,8	0,6	1,8	0,0	3,6	0,9	0,3	1,5	0,7	4,0	13,0	3,3	0,6
25	0,3	87,4	16,0	16,0	100,0	3,5	0,9	1,5	3,4	0,0	9,0	2,2	1,8	5,0	0,7	4,0	13,1	3,3	0,7

Table A.13: Execution metrics - Part 6

N°	S4 M19	S4 M20	S4 M21	S4 M22	S4 M23	S4 M24	S4 M25	S4 M26	S4 M27	S4 M28	S4 M29	S4 M30	S4 M31	S4 M32	S5 M1	S5 M2	S5 M3	S5 M4	S5 M5
1	3,3	2,7	16,0	67,0	4,2	1,1	6,6	2,6	29,0	10,8	0,2	1,6	0,7	0,4	67,9	12,0	9,0	75,0	6,7
2	4,3	2,6	14,0	58,4	4,2	1,3	7,6	2,3	31,1	8,3	0,2	1,4	0,6	0,3	89,1	17,0	17,0	100,0	8,5
3	4,3	2,7	14,0	57,6	4,1	1,3	7,6	2,7	29,1	9,0	0,3	1,0	0,6	0,2	88,9	17,0	17,0	100,0	11,6
4	4,4	2,7	14,0	52,8	3,8	0,8	5,6	2,5	27,0	9,8	0,3	1,7	0,7	0,4	95,6	17,0	17,0	100,0	8,0
5	4,3	2,7	14,0	56,0	4,0	0,9	5,7	2,5	29,8	7,2	0,2	0,8	0,5	0,2	93,0	17,0	17,0	100,0	9,1
6	4,3	2,7	14,0	57,2	4,1	1,0	5,7	2,7	27,4	8,3	0,2	0,7	0,6	0,2	90,3	17,0	17,0	100,0	8,1
7	4,3	2,7	14,0	63,9	4,6	1,3	7,2	3,0	29,5	7,3	0,1	0,7	0,5	0,2	65,0	12,0	9,0	75,0	3,3
8	4,3	2,7	14,0	54,2	3,9	1,2	7,7	2,3	28,3	7,7	0,1	0,7	0,6	0,2	89,9	17,0	17,0	100,0	10,2
9	4,4	2,7	14,0	64,5	4,6	1,3	7,2	3,0	33,3	8,6	0,1	1,9	0,6	0,4	97,7	17,0	17,0	100,0	10,2
10	4,2	2,6	12,0	52,5	4,4	1,4	6,4	2,6	34,7	7,9	0,2	1,9	0,7	0,4	87,0	17,0	17,0	100,0	7,0
11	3,7	2,7	14,0	61,6	4,4	1,0	6,3	3,3	34,1	9,3	0,2	1,4	0,7	0,3	88,2	17,0	17,0	100,0	10,2
12	4,4	2,6	14,0	59,3	4,2	1,3	6,7	2,8	28,4	9,5	0,1	1,9	0,7	0,4	90,6	17,0	17,0	100,0	8,3
13	4,3	2,6	14,0	57,2	4,1	1,0	6,3	2,6	25,9	7,0	0,1	0,7	0,5	0,2	97,9	18,0	18,0	100,0	11,7
14	4,4	2,7	14,0	59,0	4,2	1,2	6,9	2,7	31,8	8,2	0,2	1,1	0,6	0,2	80,9	15,0	14,0	93,3	9,9
15	4,3	2,7	14,0	64,3	4,6	1,0	7,4	3,0	29,0	8,4	0,2	0,9	0,6	0,2	104,0	17,0	17,0	100,0	7,0
16	4,4	2,6	14,0	61,9	4,4	1,3	7,6	2,8	34,0	12,2	0,2	5,4	0,9	1,3	88,7	17,0	17,0	100,0	10,2
17	4,3	2,6	14,0	61,6	4,4	1,5	7,2	2,8	32,3	11,3	0,1	4,3	0,8	1,0	100,4	17,0	17,0	100,0	10,1
18	4,3	2,6	14,0	53,6	3,8	1,1	6,7	2,0	29,8	9,0	0,2	1,7	0,6	0,3	92,0	17,0	17,0	100,0	9,5
19	4,3	2,7	14,0	59,3	4,2	1,0	6,1	2,6	29,5	7,1	0,1	0,8	0,5	0,2	81,7	15,0	14,0	93,3	9,3
20	4,3	2,7	14,0	62,0	4,4	1,4	7,7	3,2	28,3	7,0	0,1	0,7	0,5	0,2	95,9	17,0	17,0	100,0	9,1
21	4,3	2,7	14,0	54,4	3,9	0,7	5,6	2,9	28,7	10,0	0,2	2,1	0,7	0,5	65,4	11,0	9,0	81,8	7,4
22	4,3	2,7	14,0	59,9	4,3	1,4	7,9	2,6	28,8	6,9	0,1	0,8	0,5	0,2	89,0	17,0	17,0	100,0	8,8
23	4,3	2,7	14,0	61,9	4,4	1,3	6,6	2,4	30,7	8,5	0,2	1,1	0,6	0,2	90,0	17,0	17,0	100,0	9,0
24	4,3	2,7	14,0	53,0	3,8	0,7	5,8	3,0	27,2	8,3	0,2	1,1	0,6	0,2	92,2	17,0	17,0	100,0	10,2
25	4,4	2,7	14,0	61,7	4,4	1,2	6,5	2,6	32,5	9,0	0,2	1,3	0,6	0,2	91,5	17,0	17,0	100,0	8,1

Table A.14: Execution metrics - Part 7

N°	S5 M6	S5 M7	S5 M8	S5 M9	S5 M10	S5 M11	S5 M12	S5 M13	S5 M14	S5 M15	S5 M16	S5 M17	S5 M18	S5 M19	S5 M20	S5 M21	S5 M22	S5 M23	S5 M24
1	0,7	0,5	2,0	0,1	19,1	1,9	1,4	5,1	0,6	10,0	32,7	3,3	0,6	4,3	2,6	10,0	40,1	4,0	1,0
2	0,5	0,2	1,1	0,4	7,6	1,9	1,2	3,8	0,6	4,0	12,6	3,2	0,6	4,1	2,6	16,0	61,6	3,8	0,7
3	0,7	0,6	2,4	0,4	7,7	1,9	1,4	4,3	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	61,8	3,9	0,8
4	0,5	0,5	2,4	0,0	9,2	2,3	1,4	4,3	0,6	4,0	12,9	3,2	0,7	4,2	2,6	16,0	69,3	4,3	1,4
5	0,5	0,6	3,0	0,0	8,7	2,2	0,9	3,0	0,7	4,0	12,9	3,2	0,6	4,2	2,6	16,0	68,6	4,3	1,1
6	0,5	0,2	1,2	0,4	8,6	2,2	1,8	5,1	0,7	4,0	12,9	3,2	0,6	4,2	2,6	16,0	66,6	4,2	1,5
7	0,3	0,7	2,5	0,0	19,0	1,9	1,1	3,6	0,7	10,0	32,8	3,3	0,6	4,4	2,6	10,0	39,7	4,0	0,9
8	0,6	0,8	3,8	0,0	6,7	1,7	0,7	2,4	0,7	4,0	12,9	3,2	0,7	4,3	2,6	16,0	63,0	3,9	0,8
9	0,6	0,5	2,5	0,2	8,4	2,1	1,2	4,1	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	71,2	4,4	1,4
10	0,4	0,3	1,5	0,0	10,2	2,6	2,0	5,1	0,6	4,0	12,7	3,2	0,6	4,1	2,6	16,0	62,6	3,9	1,0
11	0,6	0,6	3,1	0,1	8,0	2,0	1,7	4,9	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	61,7	3,9	1,0
12	0,5	0,6	2,9	0,0	6,8	1,7	0,8	2,6	0,7	4,0	12,9	3,2	0,7	4,2	2,6	16,0	66,8	4,2	1,1
13	0,7	0,8	3,2	0,0	6,3	1,6	1,2	3,5	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	66,3	4,1	1,2
14	0,7	0,8	2,9	0,0	11,3	1,9	1,0	3,2	0,6	6,0	19,1	3,2	0,5	4,2	2,7	14,0	52,3	3,7	0,9
15	0,4	0,5	2,4	0,0	10,1	2,5	1,7	5,2	0,7	4,0	12,9	3,2	0,6	4,2	2,7	16,0	75,1	4,7	1,8
16	0,6	0,8	3,6	0,0	7,4	1,8	1,5	4,4	0,6	4,0	12,7	3,2	0,7	4,2	2,6	16,0	63,0	3,9	1,3
17	0,7	1,2	5,2	0,0	9,3	2,3	1,2	3,7	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	64,8	4,1	1,4
18	0,6	0,7	3,2	0,0	9,8	2,4	1,7	5,2	0,6	4,0	12,9	3,2	0,6	4,2	2,6	16,0	66,7	4,2	1,2
19	0,6	0,7	2,5	0,0	15,2	2,5	1,9	6,2	0,7	6,0	19,1	3,2	0,5	4,2	2,7	14,0	54,7	3,9	1,3
20	0,5	0,3	1,4	0,1	9,5	2,4	1,2	3,9	0,7	4,0	13,0	3,2	0,6	4,2	2,7	16,0	70,0	4,4	1,3
21	0,8	1,1	3,1	0,0	16,6	2,1	1,4	5,4	0,6	8,0	26,0	3,3	0,6	4,3	2,6	10,0	39,2	3,9	1,0
22	0,5	0,5	2,3	0,0	10,2	2,6	1,4	4,4	0,7	4,0	12,9	3,2	0,6	4,3	2,6	16,0	63,6	4,0	0,9
23	0,5	0,8	3,8	0,0	6,8	1,7	1,1	3,1	0,7	4,0	12,8	3,2	0,6	4,2	2,6	16,0	65,2	4,1	1,3
24	0,6	0,4	1,7	0,4	7,8	1,9	2,0	5,3	0,6	4,0	12,9	3,2	0,7	4,3	2,6	16,0	65,6	4,1	1,2
25	0,5	0,4	1,6	0,0	5,1	1,3	0,5	1,9	0,6	4,0	12,8	3,2	0,7	4,2	2,6	16,0	68,6	4,3	1,9

Table A.15: Execution metrics - Part 8

N°	S5 M25	S5 M26	S5 M27	S5 M28	S5 M29	S5 M30	S5 M31	S5 M32	S6 M1	S6 M2	S6 M3	S6 M4	S6 M5	S6 M6	S6 M7	S6 M8	S6 M9	S6 M10	S6 M11	S6 M12
1	6,1	2,9	21,9	5,7	0,0	0,7	0,5	0,2	53,0	10,0	9,0	90,0	7,2	0,9	0,9	2,9	0,0	9,8	1,2	0,7
2	5,0	2,7	23,1	8,9	0,0	0,8	0,5	0,2	59,2	11,0	11,0	100,0	11,8	1,2	1,0	2,7	0,0	11,3	1,1	0,7
3	5,5	2,9	24,0	9,1	0,0	0,7	0,5	0,2	65,9	11,0	11,0	100,0	20,5	2,1	2,0	5,7	0,0	7,8	0,8	0,2
4	8,2	3,0	21,8	11,8	0,0	1,3	0,7	0,3	62,8	11,0	11,0	100,0	11,8	1,2	2,3	7,7	0,0	11,7	1,2	1,0
5	6,9	2,9	21,8	8,8	0,0	0,7	0,5	0,2	54,8	11,0	11,0	100,0	2,3	0,2	0,2	0,7	0,0	15,1	1,5	1,2
6	8,1	2,6	22,4	8,5	0,0	0,7	0,5	0,2	67,4	13,0	9,0	69,2	8,5	0,9	1,1	3,5	0,0	8,1	1,0	0,8
7	5,8	2,8	17,6	6,7	0,0	1,5	0,6	0,4	60,5	11,0	11,0	100,0	11,2	1,1	1,6	4,4	0,0	11,0	1,1	0,7
8	5,6	2,7	20,3	10,2	0,0	1,3	0,6	0,3	85,9	17,0	11,0	64,7	6,3	0,9	1,3	3,7	0,0	7,3	1,2	0,8
9	7,8	2,8	24,2	9,6	0,0	1,4	0,6	0,4	54,5	11,0	11,0	100,0	6,6	0,7	0,4	1,2	0,0	10,4	1,0	0,7
10	6,2	2,6	22,7	10,9	0,0	1,7	0,6	0,4	64,9	13,0	10,0	76,9	2,5	0,3	0,6	1,7	0,0	12,2	1,5	1,0
11	6,0	2,8	22,1	9,7	0,0	1,3	0,6	0,3	65,9	12,0	11,0	91,7	8,8	0,9	1,0	3,3	0,0	15,9	1,6	1,6
12	6,4	2,7	19,2	9,2	0,0	1,0	0,5	0,2	59,2	11,0	11,0	100,0	9,4	0,9	1,1	3,3	0,0	10,6	1,1	0,7
13	6,7	2,7	26,3	9,2	0,0	1,0	0,5	0,3	76,0	13,0	12,0	92,3	6,3	0,6	1,2	3,4	0,0	23,3	2,3	1,8
14	5,8	2,4	19,8	8,5	0,0	1,6	0,6	0,3	70,0	14,0	11,0	78,6	13,4	1,7	2,1	5,1	0,0	8,2	1,0	0,7
15	9,1	2,4	22,5	9,7	0,0	1,3	0,6	0,3	58,2	11,0	11,0	100,0	5,4	0,5	0,8	2,5	0,0	15,4	1,5	1,1
16	6,7	2,5	20,9	9,0	0,0	0,7	0,5	0,2	64,3	12,0	9,0	75,0	7,6	0,9	1,2	3,7	0,0	9,1	1,1	0,9
17	8,0	2,6	26,6	9,1	0,0	0,9	0,5	0,2	66,3	13,0	8,0	61,5	0,5	0,1	0,2	0,4	0,0	7,8	1,3	0,6
18	7,0	2,7	23,1	9,5	0,0	1,4	0,6	0,3	54,0	11,0	11,0	100,0	3,7	0,4	0,7	2,3	0,0	13,4	1,3	0,7
19	7,7	2,6	20,6	8,3	0,0	0,7	0,6	0,2	75,4	13,0	11,0	84,6	16,2	1,6	2,9	8,2	0,0	11,4	1,1	0,6
20	7,6	2,8	24,7	10,1	0,0	1,4	0,6	0,3	66,7	13,0	11,0	84,6	9,5	1,0	1,3	3,8	0,0	8,9	0,9	0,5
21	6,7	3,0	19,4	5,7	0,0	0,7	0,5	0,2	69,3	13,0	10,0	76,9	7,6	0,9	1,1	3,2	0,0	12,1	1,5	1,6
22	5,5	2,7	24,0	10,0	0,0	0,8	0,6	0,2	69,1	13,0	10,0	76,9	8,1	1,0	1,0	2,4	0,0	10,3	1,3	1,1
23	7,1	2,3	19,1	9,0	0,0	0,7	0,5	0,2	62,9	12,0	11,0	91,7	5,2	0,5	1,0	3,4	0,0	13,4	1,3	1,0
24	5,9	2,1	23,5	9,6	0,0	1,3	0,6	0,3	60,4	11,0	11,0	100,0	13,9	1,4	1,1	3,4	0,0	8,6	0,9	0,6
25	8,1	2,2	18,4	8,2	0,0	0,8	0,5	0,2	89,5	16,0	10,0	62,5	4,0	0,7	0,8	1,9	0,0	8,6	1,4	0,8

Table A.16: Execution metrics - Part 9

N°	S6 M13	S6 M14	S6 M15	S6 M16	S6 M17	S6 M18	S6 M19	S6 M20	S6 M21	S6 M22	S6 M23	S6 M24	S6 M25	S6 M26	S6 M27	S6 M28	S6 M29	S6 M30	S6 M31	S6 M32
1	2,6	0,6	8,0	26,3	3,3	0,6	4,3	2,6	8,0	32,3	4,0	1,7	6,7	1,5	53,0	6,8	0,2	3,4	0,9	1,0
2	2,8	0,6	10,0	32,7	3,3	0,5	4,3	2,6	8,0	34,1	4,3	1,2	6,0	2,8	59,2	4,9	0,4	0,9	0,6	0,2
3	1,3	0,6	10,0	33,2	3,3	0,5	4,3	2,6	8,0	32,2	4,0	0,7	4,9	3,2	65,9	4,9	0,2	0,7	0,6	0,2
4	4,2	0,6	10,0	35,1	3,5	0,6	4,6	2,9	8,0	32,5	4,1	1,9	7,7	0,5	62,8	6,6	0,2	2,9	0,8	0,8
5	4,7	0,6	10,0	33,2	3,3	0,5	4,3	2,7	8,0	30,6	3,8	1,3	6,8	2,2	54,8	5,0	0,4	0,7	0,6	0,1
6	3,2	0,6	8,0	26,8	3,3	0,6	4,3	2,6	10,0	41,3	4,1	1,1	6,5	2,9	67,4	6,1	0,1	1,2	0,6	0,3
7	3,0	0,6	10,0	32,8	3,3	0,5	4,3	2,6	8,0	30,6	3,8	0,9	5,7	2,8	60,5	5,4	0,6	0,8	0,7	0,1
8	2,9	0,6	6,0	19,8	3,3	0,5	4,3	2,6	14,0	53,5	3,8	1,1	6,5	2,7	85,9	8,1	0,2	1,0	0,6	0,2
9	3,1	0,6	10,0	33,1	3,3	0,5	4,3	2,6	8,0	31,7	4,0	1,1	6,5	2,8	54,5	4,3	0,3	0,7	0,5	0,1
10	3,8	0,6	8,0	26,8	3,4	0,6	4,3	2,6	10,0	39,5	3,9	1,0	6,2	2,9	64,9	5,8	0,1	0,7	0,6	0,2
11	5,7	0,6	10,0	33,0	3,3	0,5	4,3	2,6	8,0	33,4	4,2	1,2	7,0	3,2	65,9	5,4	0,2	1,6	0,7	0,4
12	2,7	0,6	10,0	33,0	3,3	0,5	4,3	2,6	8,0	34,1	4,3	1,3	6,7	2,5	59,2	4,0	0,1	0,7	0,5	0,2
13	5,8	0,6	10,0	32,6	3,3	0,4	4,1	2,6	10,0	47,2	4,7	1,6	8,3	2,6	76,0	7,0	0,5	1,2	0,7	0,2
14	2,8	0,6	8,0	27,0	3,4	0,6	4,3	2,6	10,0	37,7	3,8	1,3	7,4	2,9	70,0	6,7	0,6	0,8	0,7	0,0
15	3,8	0,6	10,0	33,1	3,3	0,5	4,3	2,6	8,0	32,3	4,0	1,3	7,1	2,7	58,2	4,8	0,2	0,7	0,6	0,2
16	3,4	0,6	8,0	26,9	3,4	0,6	4,2	2,6	10,0	40,1	4,0	1,0	6,5	3,0	64,3	5,9	0,1	1,0	0,6	0,3
17	2,4	0,7	6,0	20,0	3,3	0,7	4,3	2,6	10,0	43,8	4,4	1,7	8,1	3,0	66,3	5,3	0,2	1,0	0,5	0,2
18	2,7	0,6	10,0	33,2	3,3	0,5	4,3	2,6	8,0	31,2	3,9	0,5	4,8	3,1	54,0	4,5	0,2	0,7	0,6	0,2
19	2,2	0,6	10,0	33,4	3,3	0,5	4,3	2,6	10,0	39,1	3,9	0,9	5,7	2,8	75,4	4,9	0,1	0,7	0,5	0,2
20	1,9	0,6	10,0	33,6	3,4	0,5	4,4	2,7	8,0	32,4	4,0	1,2	6,9	2,8	66,7	3,4	0,2	0,7	0,4	0,2
21	5,3	0,6	8,0	26,9	3,4	0,6	4,3	2,6	10,0	39,4	3,9	1,2	6,8	2,5	69,3	5,7	0,2	1,6	0,6	0,4
22	3,7	0,6	8,0	27,0	3,4	0,6	4,3	2,6	10,0	38,9	3,9	0,7	5,2	3,2	69,1	5,4	0,2	0,7	0,5	0,2
23	3,8	0,6	10,0	32,9	3,3	0,5	4,3	2,6	8,0	33,5	4,2	1,1	6,2	3,0	62,9	4,6	0,3	0,7	0,6	0,2
24	2,6	0,6	10,0	33,3	3,3	0,5	4,3	2,7	8,0	33,6	4,2	0,9	6,2	3,2	60,4	4,8	0,4	0,7	0,6	0,1
25	2,8	0,6	6,0	19,2	3,2	0,5	4,1	2,6	14,0	56,9	4,1	0,9	6,1	3,0	89,5	7,3	0,2	0,7	0,5	0,2

Table A.17: Execution metrics - Part 10

ID	Comments
1	Quand robot défie notre logique perturbant et frustant. Globalement bien aimé. Notion de hierarchie, en fonction de la tâche (notamment simple) on se supérieur (on sait mieux faire) et donc le robot doit nous suivre. (2) bien pour tâche au + vite car va vite mais moins bien pour être lib, dépend de la tâche, pire (4)
2	RF est bien mais les mauvais choix sont pénible, frustrant, il ne concidère pas mon objectif/preferences. Cependant il s'adapte quand même une fois l'action faite. (2) RF trop cool, va + vite pour faire la tâche mais marche pas bien pour lib. HF est plus efficace en fonction de l'objectif. pire (4)
3	Chrono TO stressant. ne pas pouvoir prendre les cubes à l'avance n'est pas naturel, pertubant et rend un peu compliqué mais devient simple une fois habitué. Son de fin de tâche. Se sent obligé de faire quelque chose à chaque étape, et donc meme en RF clique sur la main pour confirmer que je serai passif. Rappeller à la fin de tache le regime et obj. (5) et (1) HF pour finir la tâche une fois habitué va relativement vite et fluide. HF pour etre libéré aussi, apres rien a faire.
4	Bon moment, Clair preference pour HF. RF est une catastrophe. En RF on n'a pas vraiment son mot à dire. En plus, quand RF commence à faire une erreur on est plus concentré à l'empecher de continuer à faire des erreurs plutot que sur la tâche, très frustant. Meme quand RF fait bon choix on se sent obligé de l'ecouter et impuissant.(5) HF et lib, car fluide en controle et j'ai pu atteindre mon objectif. Pire (6)
5	Intuitif, marche bien, efficace. Etre libere un peu frustant car on a envie d'agir, regarder le robot faire etre penible.. Sol: Donner une tache auxiliaire à l'humain a faire uniquement quand se désengage de la tache principale ? (3) HF et tache wrong. Car H agit le plus, R agit seulement quand necessaire. De plus, le robot s'adapte. Il a anticipé si jamais je ne prend pas le bleu en prenant le vert, mais une fois qu'il m'a vu prendre le bleu il n'a pas pris le sien. pire (4)
6	Globalement positif. Aurai aimé que le robot donne plus d'indication, guide plus les actions. (2) vif, mieux pour la tache, mais moins prévisible. (1) bien aussi mais plus lent/passif. pire le (6)
7	Certain scenario vraiment contraignant et frustant. Uiliser ressource commune en 1er est vraiment un mauvaix choix. Globalement ok. 2 efficace une fois compris. Faire la dernière action est gratifiant, donc que le robot s'adapte pour le permettre s'est positif.Preféré (2), rapide et bon choix, pire 4

Table A.18: Comments from participants given after the experiment. Part 1

ID	Comments
8	Parfois frustant a cause des mauvais choix du robot. Les actions sont "rigide", pas naturel (attraper cube bleu sous vert, le faisant "tomber" le vert). Bien dans l'ensemble, le fait que chacun ai sa part rend la collaboration pertinente et utile. Je prefère que la priorité des choix et actions soit à l'humain. (3) car s'adapte à ce que je fais et c'est clair. Pire (4) un mauvais choix du robot a impliqué un mauvais choix de ma part, frustrant (1) simplement frustrant, le robot semble contre moi.
9	RF pas efficace, mais follow est plus simple, moins compliqué. HF mieux mais parfois incohérent (barre rose). (1) est le plus satisfaisant. (4) est le pire
10	ne pas pouvoir attraper en avance un peu agaçant. Le fait de devoir regarder le but à gauche + la scène + lire le prompt text un peu complex =>mieux si robot dit quoi faire.Prefère quand le robot est "passif", dans le sens follower. Que le robot attende qu'on decide puis agisses. N'aime pas quand le robot "prend des initiatives" car possible mauvais choix: vole les cubes très agaçant.. + prend cube commun en 1er. Pref 1 Pire 4
11	Bonne simu, claire. Mvt ont l'air réel tache claire et globalement se passe bien. Les steps cadense bien, pratique Meilleur (1) pire (4)
12	Temps d'attente (step) frustrant. Serait bien de pouvoir prendre avant pour indiquer intention, donner info. HF + interessant + de control, - efficace mais - frustrant, - imprevu et donc de mauvais choix.(+) 1 (-) 6 car on est obligé de reposer le cube.
13	Interaction simple, comme un jeu video. Les mauvais choix du robot sont assez frustrant. Simple et plaisant.(+) 3 car s'est bien adapté malgré erreur humaine (-) 4 car vole les cubes
14	Tres bien dans l'ensemble. Géné par l'affichage, du mal à lire. Certain scenario efficace d'autre non. Simple, sauf lecture/texte. (+) 5 fini rapidement, double satisfaction de finir sa part vitre puis voir la pile fini par le robot (-) 6 degouté, frustrant
15	Bien, a volé 2 fois les cubes, pas agréable. HF + facile.(+) 2 (-) 4, 3
16	Simple, agréable. Le manque de collision avec cube réduit le réalisme mais ok. Un peu confu/pertubant et un peu frustrant de devoir attendre que robot finisse action avant d'agir à nouveau.(+) 2, RF car rapide (-) 6

Table A.19: Comments from participants given after the experiment. Part 2

ID	Comments
17	certain scenario ok =>c'est plutot H qui a mal agit, 2 sce avec mauvais choix de R. HF/RF pas tellement choix !=(-) 6 RF etre lib (+) 2
18	pas pouvoir prendre cube en avance frustrant. En RF, R devient prévisiblement génant, on prévoit et réfléchi pour s'adapter et anticiper mauvais choix (defensif). HF mieux.(+) HF tache plus vite, plus interessant que lib rapidement, ennuyeux. (-) 4, R imprévisible, devient ennemi
19	Globalement ok reagit bien, comprend bien ce que je faisait (pink), avant dernier tres frustrant, vole les cubes.. Sinon bien passé (+) 2 (-) 4
20	Simu claire, voit bien le R et quand il agit, interaction bonne et claire, 4 mauvais mais globalement benefique interaction. Pref au quotidien laisser R faire les tache chronophage, donc bien aimé obj lib au plus vite.(+) 5 puis 2 (-) 4
21	R prévisible cool, on peut preshot et anticiper. (+) 1 3 (-) 4
22	Sympa, simu bien faite. interactif, on est pris dedans et acteur. Beaucoup d'émotion déjà (satisfaction / frustration) donc si c'était une tache plus concrete ça serait encore plus frustrant.(+) 1 3 (-) 4
23	Simu bien faite, s'imagine bien interaction. Au debut un peu confu diff entre HF/RF puis ok. Confusion cube blanc et gris(+) aucun (-) 6
24	Interessant, jamais une gène, interaction amusante, sympa de prédire RA(+) 1, 3 (-) 6
25	HF ok, qd RF on peut rien faire. Simulation moins naturelle, pas parfaitement réaliste. Notion etape/synchronisation perturbant un peu (+) 5 HF lib (-) 4

Table A.20: Comments from participants given after the experiment. Part 3

	S1	S2	S3	S4	S5	S6
Times preferred the most	***9***	***7***	***3***	*0*	***5***	*0*
Times preferred the least	*0*	*0*	*0*	***16***	*0*	***9***

Table A.21: Number of times each scenario has been respectively preferred the most and the least. HF scenarios are never disliked and RF scenarios with erroneous estimations are never preferred.

Bibliography

[Alami 1993] Rachid Alami, Raja Chatila and Bernard Espiau. *Designing an intelligent control architecture for autonomous robots.* In ICAR, volume 93, pages 435–440, 1993. (Cited in page 13.)

[Alami 1998] R. Alami, R. Chatila, S. Fleury, M. Ghallab and F. Ingrand. *An Architecture for Autonomy.* The International Journal of Robotics Research, vol. 17, no. 4, pages 315–337, April 1998. (Cited in page 13.)

[Alili 2009] Samir Alili, Rachid Alami and Vincent Montreuil. *A task planner for an autonomous social robot.* Distributed autonomous robotic systems 8, pages 335–344, 2009. (Cited in page 26.)

[Anderson 2004] John R Anderson, Daniel Bothell, Michael D Byrne, Scott Douglass, Christian Lebiere and Yulin Qin. *An integrated theory of the mind.* Psychological review, vol. 111, no. 4, page 1036, 2004. (Cited in page 13.)

[Annett 1967] John Annett and Keith D Duncan. *Task analysis and training design.* Journal of Occupational Psychology, 1967. (Cited in page 15.)

[Arntz 2022] Alexander Arntz, Carolin Straßmann, Stefanie Völker and Sabrina C. Eimler. *Collaborating eye to eye: Effects of workplace design on the perception of dominance of collaboration robots.* Frontiers in Robotics and AI, vol. 9, page 999308, September 2022. (Cited in page 49.)

[Aroor 2018] Anoop Aroor, Susan L. Epstein and Raj Korpan. *MengeROS: a Crowd Simulation Tool for Autonomous Robot Navigation.* arXiv:1801.08823 [cs], January 2018. (Cited in page 126.)

[Bartneck 2020] Christoph Bartneck, Tony Belpaeme, Friederike Eyssel, Takayuki Kanda, Merel Keijsers and Selma Šabanović. Human-robot interaction: An introduction. Cambridge University Press, 2020. (Cited in page 7.)

[Belle 2023] Vaishak Belle, Thomas Bolander, Andreas Herzig and Bernhard Nebel. *Epistemic planning: Perspectives on the special issue.* Artificial Intelligence, vol. 316, page 103842, March 2023. (Cited in page 48.)

[Bolander 2017] Thomas Bolander. *A Gentle Introduction to Epistemic Planning: The DEL Approach.* Electronic Proceedings in Theoretical Computer Science, vol. 243, pages 1–22, March 2017. (Cited in pages 48 and 152.)

[Bolander 2021] Thomas Bolander, Lasse Dissing and Nicolai Herrmann. *DEL-based Epistemic Planning for Human-Robot Collaboration: Theory and Implementation.* In Proc. of KR, 2021. (Cited in page 53.)

[Bratman 1987] Michael Bratman. Intention, plans, and practical reason. Cambridge, MA: Harvard University Press, Cambridge, 1987. (Cited in page 11.)

[Buckingham 2020] David Buckingham, Meia Chita-Tegmark and Matthias Scheutz. *Robot planning with mental models of co-present humans.* In International Conference on Social Robotics, pages 566–577. Springer, 2020. (Cited in page 29.)

[Buisan 2021] Guilhem Buisan. *Planning For Both Robot and Human: Anticipating and Accompanying Human Decisions.* PhD thesis, INSA Toulouse, France, 2021. (Cited in pages 14, 19, and 26.)

[Buisan 2022] Guilhem Buisan, Anthony Favier, Amandine Mayima and Rachid Alami. *HATP/EHDA: A Robot Task Planner Anticipating and Eliciting Human Decisions and Actions.* In Proc. of ICRA, 2022. (Cited in pages 2, 62, and 63.)

[Chakraborti 2015] Tathagata Chakraborti, Gordon Briggs, Kartik Talamadupula, Yu Zhang, Matthias Scheutz, David E. Smith and Subbarao Kambhampati. *Planning for serendipity.* In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015, pages 5300–5306. IEEE, 2015. (Cited in page 14.)

[Chakraborti 2017] Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang and Subbarao Kambhampati. *Plan explanations as model reconciliation: moving beyond explanation as soliloquy.* In Proceedings of the 26th International Joint Conference on Artificial Intelligence, pages 156–163, 2017. (Cited in page 28.)

[Chatila 1992] Raja Chatila, Rachid Alami, Bernard Degallaix and Herve Laruelle. *Integrated planning and execution control of autonomous robot actions.* In Proceedings 1992 IEEE International Conference on Robotics and Automation, pages 2689–2690. IEEE Computer Society, 1992. (Cited in page 13.)

[Chen 2018] Xu Chen, Martin Treiber, Venkatesan Kanagaraj and Haiying Li. *Social force models for pedestrian traffic–state of the art.* Transport reviews, vol. 38, no. 5, pages 625–653, 2018. Publisher: Taylor & Francis. (Cited in page 20.)

[Cherry 1957] Colin Cherry. *On human communication; a review, a survey, and a criticism.* 1957. Publisher: The Technology Press of MIT. (Cited in page 8.)

[Cirillo 2009a] Marcello Cirillo, Lars Karlsson and Alessandro Saffiotti. *A Human-Aware Robot Task Planner.* In Proc. of ICAPS 2009, 2009. (Cited in page 62.)

[Cirillo 2009b] Marcello Cirillo, Lars Karlsson and Alessandro Saffiotti. *Human-aware task planning for mobile robots.* In ICAR 2009, 2009. (Cited in page 62.)

[Clodic 2017] Aurélie Clodic, Elisabeth Pacherie, Rachid Alami and Raja Chatila. *Key Elements for Human Robot Joint Action.* In Sociality and Normativity for RobotsPhilosophical Inquiries into Human-Robot Interactions, Studies in the Philosophy of Sociality, pages 159–177. Springer, 2017. This document is an extended version of the one published in the proceedings of RoboPhilosophy conference. (Cited in pages 62 and 63.)

[Cohen 1970] Philip Cohen and Hector Levesque. *On Team Formation.* February 1970. (Cited in page 9.)

[Cohen 1988] Jacob Cohen. *The concepts of power analysis. Statistical power analysis for the behavioral sciences.* Hillsdale: Elrbaum, 1988. (Cited in page 108.)

[Cohen 1991] Philip R Cohen and Hector J Levesque. *Teamwork.* Nous, vol. 25, no. 4, pages 487–512, 1991. Publisher: JSTOR. (Cited in pages 9 and 11.)

[Crosby 2014] Matthew Crosby, Anders Jonsson and Michael Rovatsos. *A Single-Agent Approach to Multiagent Planning.* In Proc. of ECAI, 2014. (Cited in pages 58 and 63.)

[Curioni 2019] Arianna Curioni, Cordula Vesper, Günther Knoblich and Natalie Sebanz. *Reciprocal information flow and role distribution support joint action coordination.* Cognition, vol. 187, pages 21–31, 2019. (Cited in page 63.)

[Curioni 2022] Arianna Curioni, Pavel Voinov, Matthias Allritz, Thomas Wolf, Josep Call and Günther Knoblich. *Human adults prefer to cooperate even when it is costly.* Proceedings of the Royal Society B: Biological Sciences, vol. 289, 04 2022. (Cited in page 63.)

[Darvish 2021] Kourosh Darvish, Enrico Simetti, Fulvio Mastrogiovanni and Giuseppe Casalino. *A Hierarchical Architecture for Human-Robot Cooperation Processes.* IEEE Trans. Robotics, vol. 37, no. 2, pages 567–586, 2021. (Cited in page 30.)

[De Carolis 2000] Berardina De Carolis, Catherine Pelachaud and Isabella Poggi. *Verbal and nonverbal discourse planning.* In Proc. AAMAS 2000 Workshop "Achieving Human-Like Behavior in Interactive Animated Agents, 2000. (Cited in page 19.)

[De Carolis 2001] Berardina De Carolis, Catherine Pelachaud, Isabella Poggi and Fiorella de Rosis. *Behavior planning for a reflexive agent.* In International Joint Conference on Artificial Intelligence, volume 17, pages 1059–1066. LAWRENCE ERLBAUM ASSOCIATES LTD, 2001. Issue: 1. (Cited in page 19.)

[Devin 2016] Sandra Devin and Rachid Alami. *An implemented theory of mind to improve human-robot shared plans execution.* In 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 319–326, Christchurch, New Zealand, March 2016. IEEE. (Cited in pages 46 and 47.)

[Devin 2018] Sandra Devin, Camille Vrignaud, Kathleen Belhassein, Aurelie Clodic, Ophelie Carreras and Rachid Alami. *Evaluating the Pertinence of Robot Decisions in a Human-Robot Joint Action Context: The PeRDITA Questionnaire.* In 2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), pages 144–151, Nanjing, August 2018. IEEE. (Cited in pages 100 and 101.)

[Echeverria 2011] Gilberto Echeverria, Nicolas Lassabe, Arnaud Degroote and Séverin Lemaignan. *Modular open robots simulation engine: Morse.* In 2011 ieee international conference on robotics and automation, pages 46–51. IEEE, 2011. (Cited in page 135.)

[Echeverria 2012] G. Echeverria, S. Lemaignan and A. et Al. Degroote. *Simulating Complex Robotic Scenarios with MORSE.* In Simulation, Modeling, and Programming for Autonomous Robots, volume 7628, pages 197–208. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. (Cited in page 126.)

[Erdogan 2022] Emre Erdogan, Frank Dignum, Rineke Verbrugge and Pınar Yolum. *Abstracting Minds: Computational Theory of Mind for Human-Agent Collaboration.* In Stefan Schlobach, María Pérez-Ortiz and Myrthe Tielman, editors, Frontiers in Artificial Intelligence and Applications. IOS Press, September 2022. (Cited in page 47.)

[Erol 1996] Kutluhan Erol, James Hendler and Dana S Nau. *Complexity results for HTN planning.* Annals of Mathematics and Artificial Intelligence, vol. 18, no. 1, pages 69–93, 1996. Publisher: Springer. (Cited in page 15.)

[Favier 2023] Anthony Favier, Phani Teja Singamaneni and Rachid Alami. *Challenging Human-Aware Robot Navigation with an Intelligent Human Simulation System.* In Social Simulation Conference (SSC), Glasgow, France, September 2023. (Cited in page 143.)

[Ferrari 2022] Davide Ferrari, Federico Benzi and Cristian Secchi. *Bidirectional Communication Control for Human-Robot Collaboration*, June 2022. arXiv:2206.05202 [cs]. (Cited in page 49.)

[Fusaro 2021] Fabio Fusaro, Edoardo Lamon, Elena De Momi and Arash Ajoudani. *A Human-Aware Method to Plan Complex Cooperative and Autonomous Tasks using Behavior Trees.* In 2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids), pages 522–529, Munich, Germany, July 2021. IEEE. (Cited in page 31.)

[Galland 2022] Lucie Galland, Catherine Pelachaud and Florian Pecune. *Adapting conversational strategies to co-optimize agent's task performance and user's engagement.* In Proceedings of the 22nd ACM International Conference on Intelligent Virtual Agents, pages 1–3, Faro Portugal, September 2022. ACM. (Cited in page 49.)

[Ghallab 2016] Malik Ghallab, Dana Nau and Paolo Traverso. Automated planning and acting. Cambridge University Press, 2016. (Cited in pages 16, 17, and 32.)

[Gombolay 2015] Matthew Gombolay, Reymundo Gutierrez, Shanelle Clarke, Giancarlo Sturla and Julie Shah. *Decision-Making Authority, Team Efficiency and Human Worker Satisfaction in Mixed Human-Robot Teams.* Autonomous Robots, vol. 39, 07 2015. (Cited in page 63.)

[Gordon 2023] Jeremy Gordon, Guenther Knoblich and Giovanni Pezzulo. *Strategic Task Decomposition in Joint Action.* Cognitive Science, vol. 47, no. 7, page e13316, 2023. (Cited in page 62.)

[Grice 1975] HP Grice. *Logic and Conversation.* Syntax and Semantics, vol. 3, pages 43–58, 1975. (Cited in page 9.)

[Gurney 2022] Nikolos Gurney and David V. Pynadath. *Robots with Theory of Mind for Humans: A Survey.* In 2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 993–1000, August 2022. ISSN: 1944-9437. (Cited in page 47.)

[Helbing 1995] Dirk Helbing and Peter Molnar. *Social force model for pedestrian dynamics.* Physical review E, vol. 51, no. 5, page 4282, 1995. (Cited in pages 20 and 126.)

[Hoffman 2004] Guy Hoffman and Cynthia Breazeal. *Collaboration in human-robot teams.* In AIAA 1st intelligent systems technical conference, page 6434, 2004. (Cited in page 11.)

[Hoffman 2007] Guy Hoffman and Cynthia Breazeal. *Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team.* In Proceedings of the ACM/IEEE international conference on Human-robot interaction, pages 1–8, 2007. (Cited in page 29.)

[Hoffmann 2005] Jörg Hoffmann and Ronen Brafman. *Contingent planning via heuristic forward search with implicit belief states.* In Proc. ICAPS, volume 2005. Citeseer, 2005. (Cited in page 30.)

[Iocchi 2016] Luca Iocchi, Laurent Jeanpierre, Maria Lazaro and Abdel-Illah Mouaddib. *A practical framework for robust decision-theoretic planning and execution for service robots.* In Proceedings of the International Conference on Automated Planning and Scheduling, volume 26, pages 486–494, 2016. (Cited in page 30.)

[Izquierdo-Badiola 2022] Silvia Izquierdo-Badiola, Gerard Canal, Carlos Rizzo and Guillem Alenya. *Improved Task Planning through Failure Anticipation in Human-Robot Collaboration.* In 2022 International Conference on Robotics and Automation (ICRA), pages 7875–7880, Philadelphia, PA, USA, May 2022. IEEE. (Cited in page 31.)

[Johannsmeier 2016] Lars Johannsmeier and Sami Haddadin. *A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes.* IEEE Robotics and Automation Letters, vol. 2, no. 1, pages 41–48, 2016. (Cited in page 28.)

[Johnson 2018] Benjamin Johnson, Michael W. Floyd, Alexandra Coman, Mark A. Wilson and David W. Aha. *Goal Reasoning and Trusted Autonomy.* In Foundations of Trusted Autonomy, volume 117. Springer, 2018. (Cited in page 127.)

[Köckemann 2014] Uwe Köckemann, Federico Pecora and Lars Karlsson. *Grandpa hates robots-interaction constraints for planning in inhabited environments.* In Proceedings of the AAAI Conference on Artificial Intelligence, volume 28, 2014. (Cited in page 62.)

[Koenig 2004] Nathan Koenig and Andrew Howard. *Design and use paradigms for gazebo, an open-source multi-robot simulator.* In 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566), volume 3, pages 2149–2154. IEEE, 2004. (Cited in page 135.)

[Kollmitz 2015] Marina Kollmitz, Kaijen Hsiao, Johannes Gaa and Wolfram Burgard. *Time dependent planning on a layered social cost map for human-aware robot navigation.* In 2015 European Conference on Mobile Robots (ECMR), pages 1–6, Lincoln, United Kingdom, September 2015. IEEE. (Cited in page 134.)

[Koppula 2016] Hema S Koppula, Ashesh Jain and Ashutosh Saxena. *Anticipatory planning for human-robot teams.* In Experimental robotics, pages 453–470. Springer, 2016. (Cited in page 30.)

[Kourtis 2014] Dimitrios Kourtis, Günther Knoblich, Mateusz Woźniak and Natalie Sebanz. *Attention allocation and task representation during joint action planning.* Journal of Cognitive Neuroscience, vol. 26, no. 10, pages 2275–2286, 2014. (Cited in page 63.)

[Laird 1987] John E Laird, Allen Newell and Paul S Rosenbloom. *Soar: An architecture for general intelligence.* Artificial intelligence, vol. 33, no. 1, pages 1–64, 1987. Publisher: Elsevier. (Cited in page 13.)

[Lallement 2014] Raphaël Lallement, Lavindra De Silva and Rachid Alami. *HATP: An HTN planner for robotics.* arXiv preprint arXiv:1405.5345, 2014. (Cited in page 26.)

[Lemaignan 2017] Séverin Lemaignan, Mathieu Warnier, E. Akin Sisbot, Aurélie Clodic and Rachid Alami. *Artificial cognition for social human–robot interaction: An implementation.* Artificial Intelligence, vol. 247, pages 45–69, June 2017. (Cited in page 13.)

[Lemaignan 2018] Séverin Lemaignan, Yoan Sallami, Christopher Wallbridge, Aurélie Clodic, Tony Belpaeme and Rachid Alami. *Underworlds: Cascading situation assessment for robots.* In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7750–7757. IEEE, 2018. (Cited in page 47.)

[Levesque 1990] Hector J Levesque, Philip R Cohen and José HT Nunes. On acting together. SRI International Menlo Park, CA 94025-3493, 1990. (Cited in page 9.)

[Levine 2014] Steven James Levine and Brian Charles Williams. *Concurrent plan recognition & execution for human-robot teams.* In ICAPS, 2014. (Cited in page 63.)

[Martinie 2019] Célia Martinie, Philippe Palanque, Elodie Bouzekri, Andy Cockburn, Alexandre Canny and Eric Barboni. *Analysing and demonstrating tool-supported customizable task notations.* Proc. of the ACM on HCI, vol. 3, no. EICS, pages 1–26, 2019. (Cited in page 15.)

[McEllin 2018] Luke McEllin, Natalie Sebanz and Günther Knoblich. *Identifying others' informative intentions from movement kinematics.* Cognition, vol. 180, pages 246–258, 08 2018. (Cited in page 63.)

[McMillan 2023] Donald McMillan, Razan Jaber, Benjamin R. Cowan, Joel E. Fischer, Bahar Irfan, Ronald Cumbal, Nima Zargham and Minha Lee. *Human-Robot Conversational Interaction (HRCI).* In Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction, HRI '23, pages 923–925, New York, NY, USA, March 2023. Association for Computing Machinery. (Cited in page 48.)

[Mehrabian 1967] Albert Mehrabian and Morton Wiener. *Decoding of inconsistent communications.* Journal of personality and social psychology, vol. 6, no. 1, page 109, 1967. (Cited in page 9.)

[Michael 2016] John Michael, Natalie Sebanz and Günther Knoblich. *Observing Joint Action: Coordination Creates Commitment.* Cognition, vol. 157, pages 106–113, 09 2016. (Cited in page 63.)

[Nomura 2016] Tatsuya Nomura, Takayuki Kanda, Hiroyoshi Kidokoro, Yoshitaka Suehiro and Sachie Yamada. *Why do children abuse robots?* Interaction Studies, vol. 17, no. 3, pages 347–369, 2016. (Cited in page 129.)

[Paternò 2004] Fabio Paternò. *ConcurTaskTrees: an engineered notation for task models.* The handbook of task analysis for human-computer interaction, pages 483–503, 2004. (Cited in page 15.)

[Pecora 2012] Federico Pecora, Marcello Cirillo, Francesca Dell'Osa, Jonas Ullberg and Alessandro Saffiotti. *A constraint-based approach for proactive, context-aware human support.* Journal of Ambient Intelligence and Smart Environments, vol. 4, no. 4, pages 347–367, 2012. (Cited in page 19.)

[Pérez-Higueras 2023] Noé Pérez-Higueras, Roberto Otero, Fernando Caballero and Luis Merino. *HuNavSim: A ROS2 Human Navigation Simulator for Benchmarking Human-Aware Robot Navigation.* arXiv preprint, 2023. (Cited in page 135.)

[Perille 2020] Daniel Perille, Abigail Truong, Xuesu Xiao and Peter Stone. *Benchmarking metric ground navigation.* In 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pages 116–121. IEEE, 2020. (Cited in page 20.)

[Puig 2018] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler and Antonio Torralba. *VirtualHome: Simulating Household Activities Via Programs.* In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8494–8502, Salt Lake City, UT, June 2018. IEEE. (Cited in page 126.)

[Ramachandruni 2023] Kartik Ramachandruni, Cassandra Kent and Sonia Chernova. *UHTP: A User-Aware Hierarchical Task Planning Framework for Communication-Free, Mutually-Adaptive Human-Robot Collaboration.* ACM Transactions on Human-Robot Interaction, 2023. (Cited in page 63.)

[Romeo 2022] Marta Romeo, Peter E. McKenna, David A. Robb, Gnanathusharan Rajendran, Birthe Nesset, Angelo Cangelosi and Helen Hastie. *Exploring Theory of Mind for Human-Robot Collaboration.* In 2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 461–468, August 2022. ISSN: 1944-9437. (Cited in page 47.)

[Roncone 2017] Alessandro Roncone, Olivier Mangin and Brian Scassellati. *Transparent role assignment and task allocation in human robot collaboration.* In Proc. of ICRA, 2017. (Cited in pages 62 and 63.)

[Samarakoon 2022] SM Bhagya P Samarakoon, MA Viraj J Muthugala and AG Buddhika P Jayasekara. *A Review on Human–Robot Proxemics.* Electronics, vol. 11, no. 16, page 2490, 2022. (Cited in page 20.)

[Sanelli 2017] Valerio Sanelli, Michael Cashmore, Daniele Magazzeni and Luca Iocchi. *Short-term human-robot interaction through conditional planning and execution.* In Proceedings of the International Conference on Automated Planning and Scheduling, volume 27, pages 540–548, 2017. (Cited in page 30.)

[Scheutz 2013] Matthias Scheutz, Jack Harris and Paul Schermerhorn. *Systematic integration of cognitive and robotic architectures.* Advances in Cognitive Systems, vol. 2, pages 277–296, 2013. (Cited in page 12.)

[Schmitz 2017] Laura Schmitz, Cordula Vesper, Natalie Sebanz and Günther Knoblich. *Co-Representation of Others' Task Constraints in Joint Action.* Journal of experimental psychology. Human perception and performance, vol. 43, 04 2017. (Cited in page 63.)

[Sebanz 2006a] Natalie Sebanz, Harold Bekkering and Günther Knoblich. *Joint action: bodies and minds moving together.* Trends in cognitive sciences, vol. 10, no. 2, pages 70–76, 2006. (Cited in pages 9 and 62.)

[Sebanz 2006b] Natalie Sebanz, Harold Bekkering and Günther Knoblich. *Joint Action: Bodies and Minds Moving Together.* Trends in Cognitive Sciences, vol. 10, no. 2, pages 70–76, 2006. (Cited in page 1.)

[Sebanz 2009] Natalie Sebanz and Günther Knoblich. *Prediction in Joint Action: What, When, and Where.* Top. Cogn. Sci., vol. 1, no. 2, pages 353–367, 2009. (Cited in page 62.)

[Selvaggio 2021] Mario Selvaggio, Marco Cognetti, Stefanos Nikolaidis, Serena Ivaldi and Bruno Siciliano. *Autonomy in physical human-robot interaction: A brief survey.* IEEE Robotics Autom. Lett., 2021. (Cited in page 62.)

[Shekhar 2020] Shashank Shekhar and Ronen I. Brafman. *Representing and planning with interacting actions and privacy.* AIJ, vol. 278, 2020. (Cited in page 63.)

[Shen 2020] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martın-Martın, Linxi Fan, Guanzhi Wang, Shyamal Buch, Claudia D'Arpino, Sanjana Srivastava, Lyne P Tchapmi, Kent Vainio, Li Fei-Fei and Silvio Savarese. *iGibson 1.0: a Simulation Environment for Interactive Tasks in Large Realistic Scenes.* arXiv preprint, 2020. (Cited in page 126.)

[Shvo 2022] Maayan Shvo, Ruthrash Hari, Ziggy O'Reilly, Sophia Abolore, Sze-Yuh Nina Wang and Sheila A. McIlraith. *Proactive Robotic Assistance via Theory of Mind.* In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 9148–9155, October 2022. ISSN: 2153-0866. (Cited in page 47.)

[Singamaneni 2021] Phani Teja Singamaneni, Anthony Favier and Rachid Alami. *Human-Aware Navigation Planner for Diverse Human-Robot Ineraction Contexts.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021. (Cited in pages 20, 129, and 147.)

[Smith 1998] I.A. Smith, P.R. Cohen, J.M. Bradshaw, M. Greaves and H. Holmback. *Designing conversation policies using joint intention theory.* In Proceedings International Conference on Multi Agent Systems (Cat. No.98EX160), pages 269–276, Paris, France, 1998. IEEE Comput. Soc. (Cited in page 8.)

[Strachan 2020] James Strachan and Georgina Török. *Efficiency is prioritised over fairness when distributing joint actions.* Acta Psychologica, vol. 210, page 103158, 10 2020. (Cited in page 63.)

[Thierauf 2024] Christopher Thierauf, Theresa Law, Tyler Frasca and Matthias Scheutz. *Toward Competent Robot Apprentices: Enabling Proactive Troubleshooting in Collaborative Robots.* Machines, vol. 12, no. 1, page 73, January 2024. (Cited in page 13.)

[Thomaz 2016] Andrea Thomaz, Guy Hoffman and Maya Cakmak. *Computational Human-Robot Interaction.* Foundations and Trends in Robotics, vol. 4, no. 2-3, pages 104–223, 2016. (Cited in page 10.)

[Tsoi 2020] Nathan Tsoi, Mohamed Hussein, Jeacy Espinoza, Xavier Ruiz and Marynel Vázquez. *SEAN: Social Environment for Autonomous Navigation.* In Proceedings of the 8th International Conference on Human-Agent Interaction (HAI), November 2020. (Cited in page 126.)

[Tsoi 2022] Nathan Tsoi, Alec Xiang, Peter Yu, Samuel S Sohn, Greg Schwartz, Subashri Ramesh, Mohamed Hussein, Anjali W Gupta, Mubbasir Kapadia and Marynel Vázquez. *Sean 2.0: Formalizing and generating social situations for robot navigation.* IEEE Robotics and Automation Letters, vol. 7, pages 11047–11054, 2022. (Cited in page 126.)

[Unhelkar 2020] Vaibhav V Unhelkar, Shen Li and Julie A Shah. *Decision-making for bidirectional communication in sequential human-robot collaborative tasks.* In Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction, pages 329–341, 2020. (Cited in page 29.)

[Van Den Berg 2011] Jur Van Den Berg, Stephen J Guy, Ming Lin and Dinesh Manocha. *Reciprocal n-body collision avoidance.* In Robotics research, pages 3–19. Springer, 2011. (Cited in page 126.)

[Vattam 2013] Swaroop Vattam, Matthew Evans Klenk, Matthew Molineaux and David W. Aha. *Breadth of Approaches to Goal Reasoning: A Research Survey.* In Annual

Conference on Advances in Cognitive Systems: Workshop on Goal Reasoning, 2013. (Cited in page 127.)

[Verhagen 2022] Ruben S. Verhagen, Mark A. Neerincx and Myrthe L. Tielman. *The influence of interdependence and a transparent or explainable communication style on human-robot teamwork*. Frontiers in Robotics and AI, vol. 9, page 993997, September 2022. (Cited in page 48.)

[Yamaguchi 2019] Motonori Yamaguchi, Helen J. Wall and Bernhard Hommel. *The roles of action selection and actor selection in joint task settings*. Cognition, vol. 182, pages 184–192, 2019. (Cited in page 63.)

[Yige 2021] Liu Yige, Li Siyun, Li Chengshu, Perez-D'Arpino Claudia and Savarese Silvio. *Interactive Pedestrian Simulation in iGibson*. RSS Workshop on Social Robot Navigation, 2021. (Cited in page 126.)

[Yu 2023] Chuang Yu, Baris Serhan, Marta Romeo and Angelo Cangelosi. *Robot Theory of Mind with Reverse Psychology*. In Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction, pages 545–547, Stockholm Sweden, March 2023. ACM. (Cited in page 47.)

[Ziparo 2011] Vittorio A Ziparo, Luca Iocchi, Pedro U Lima, Daniele Nardi and Pier Francesco Palamara. *Petri net plans: A framework for collaboration and coordination in multi-robot systems*. Autonomous Agents and Multi-Agent Systems, vol. 23, pages 344–383, 2011. (Cited in page 30.)