A NEW FORMULATION AND RESOLUTION SCHEMES FOR PLANNING COLLABORATIVE HUMAN-ROBOT TASKS

Jules Waldhart

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ FÉDÉRALE TOULOUSE MIDI-PYRÉNÉES

Délivré par :

l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *12/10/2018* **par :**

JULES WALDHART

A NEW FORMULATION AND RESOLUTION SCHEMES FOR PLANNING COLLABORATIVE HUMAN-ROBOT TASKS

JURY

SYLVIE PESTY	Professeur des Universités	Président du Jury
FRANÇOIS CHARPILLET	Directeur de Recherche	Rapporteur
PETER DOMINEY	Directeur de Recherche	Rapporteur
RACHID ALAMI	Directeur de Recherche	Directeur de Thèse
JUAN CORTÉS	Directeur de Recherche	Co-directeur de Thèse

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle
Unité de Recherche :
Laboratoire d'Analyse et d'Architecture des Systèmes
Directeur(s) de Thèse :
Rachid ALAMI et *Juan CORTÉS*
Rapporteurs :
François CHARPILLET et *Peter DOMINEY*

Acknowledgments

I address my first thanks to those who supported me during my years of PhD studentship, starting with my beloved Chloé, and all my friends and familly. I also thank Rachid for taking me under his supervision, and for his help, advices and impossible – challenging – ideas; and also for all the effort put in making our work as enjoyable as possible. Many thanks are deserved to colleagues for the constructive interactions, and are acknowledgeable for their help in the work presented here: Juan Cortès, Aurélie Clodic, Matthieu Herrb, Thierry Siméon, Félix Ingrand, Malik Ghallab, Mamoun Gharbi, Renaud Viry, Arthur Bit-Monnot, Alexandre Bœuf, Raphaël Lallement, Benjamin Vadant, Sandra Devin, Yoan Sallami, Guilhem Buisan, Guillaume Sarthou, Kathleen Belhassein, Théo Ollivier, Jean-François Erdelyi [1]. I acknowledge the partners of the MuMMER project for the very constructive interactions we had.

For the good time spent with them, I want to thank all the people cited above, plus all the members of the RIS team and colleagues from LAAS and INSA with who I shared quite a lot of breaks.

Finally, I want to thank all the staff from LAAS who is doing a great job in making it a great place to work and study at.

1. names are sorted by a complex function mixing order of appearance, seniority, intensity and duration of the collaboration, and some bias due to memory

Contents

Chapter 1

Introduction

Contents

1.1 Context And Tasks

Human-Robot collaboration is a field that expose lots of various challenges and as much ways to approach them. During my thesis preparation, I focused on some tasks that require specific approaches to be treated. These tasks where related to the European projects funding my work:

— ARCAS (Aerial Robotics Cooperative Assembly System [1]);
— SAPHARI (Safe and Autonomous Physical Human-Aware Robot [2]);
— MuMMER (MultiMedia Mall Entertainment Robot [3]).

The two main tasks studied are the Multi-Agent Transport Problem and a task related to providing route directions. The former is about handing over an object through several cooperating agents (*i.e.* robots and humans) to make it reach a position or a specific agent. The later aims at providing route directions to humans using speech and gestures, considering that the task can be improved by guiding the persons to a place where the route directions will be easier, thanks to a perspective that allows pointing at some pertinent landmark. Standard manipulation tasks in presence of humans or in collaboration with them were also studied; that is mostly pick and place actions with a robotic arm or humanoid robot. These tasks were acting like motors and test cases for our main contributions:

— MHO (Multiple HandOver) is a framework for solving the Multi-Agent Transport Problem with consideration for human preferences;
— SVP (Shared Visual Perspective) Planner addresses part of the guiding task, and is integrated in a system developed in our team that plans and executes such task;
— development of move4d, a software for generating motions that solve human-robot collaborative tasks;
— contribution to GTP (Geometric Task Planner), which is part of move4d;

Their common point is that they aim at solving and executing human-aware collaborative tasks better, in terms of quality or computational cost.

1.2 Preliminary Knowledge

1.3 Human-Robot Collaborative Tasks Design

A robot interacting with humans in a collaborative task has to expose a variety of capacities including perception, high level activity planning and supervision, mental state estimation and motion. For the robot to be of any interest for its human partners, it must prove useful to ensure engagement of the partners in the task. Key aspects of the usefulness are efficiency and fitness to the needs.

Nowadays, and probably for some time, robotic platforms have very limited capacities, compared with (most) humans, but may outperform us in some specific

1. http://www.arcas-project.eu/
2. http://www.saphari.eu/
3. http://www.mummer-project.eu/

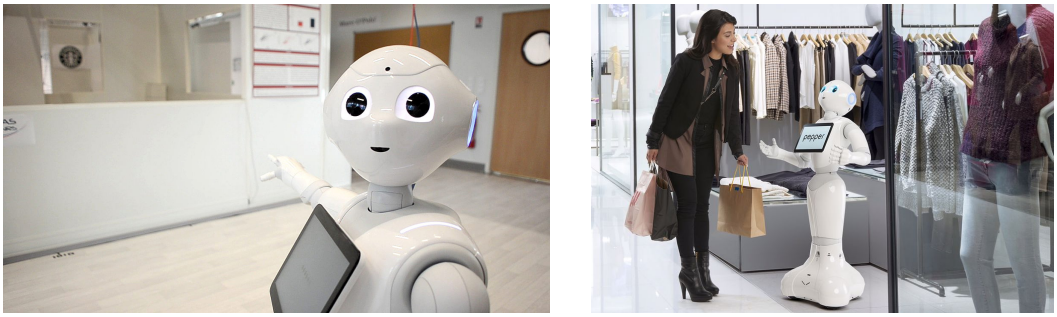


Figure 1.1: Pepper robot from SoftBank Robotics is an humanoid the size of an height-year old child (1.21 m), with a wheeled base. (Right-hand image is an advertising image from SoftBank Robotics)

domains (precision, strength, memory, compliance). To accomplish usefulness of such platforms, designers and developers must match their behaviour with their capacities, and expected capacities with real ones. Not doing so may lead to human misunderstanding, delusion, or impatience, if the robot is not capable of doing the task in a satisfactory manner while claiming (explicitly or not) that it is able to perform them.

For illustration, consider the robot Pepper, from SoftBank Robotics, illustrated in Figure 1.1. Its design resembles an eight years old child with a mobile base and a tablet on the torso. Such appearance induces an unconscious expectation for the human, who may feel deluded during the interaction when the robot proves not as strong, adroit, or sensitive as expected. Plus, if the developer programs the robot for a collaborative task it is inefficient for, human partners will expect the robot to be useful in the interaction they engaged in, and will again be disappointed, leading to a possible disengagement from the task.

When designing a robot for a commercial use, this is of course an important issue concerning success. But in HRI research, fully engaged humans are needed too, since user studies or experiments involving collaborative robots should not be disturbed by a delusion of the human. With this in mind, we will build and program robot so that they prove useful, efficiently sharing the work according to the capacities of all partners.

When building a robot for an applicative use, designers should understand the human motivations and needs and address them with the robotic solutions (when designing a user study, they may want to focus on motivations and expectations, as volunteers do not usually come with a need).

We tried to keep this in mind, so that developed solutions could adapt to degrees of robot capabilities, human abilities and needs. The Multi-Handover task solver can adapt to different robot speeds, human availability or commitment to the task, while leaving humans at their own activities when they are not needed.

The pointing task adapts to robot navigation capabilities by limiting the distance it runs to show the target. In some cases, the system may decide not to go

pointing at it, but rather providing instructions to get to it. This way, the human will not be asked to engage in a collaborative task they may find the robot unsuitable for (navigating in a crowded environment while guiding a human, in the context of the MuMMER Project, may be a difficult task for the Pepper robot).

1.4 Planning for Human and Robot Teams

The collaborative tasks addressed in this thesis are considered as joint actions, hence we formulate the hypothesis of a common goal for the team of robots and humans. This hypothesis allows us to formulate another one: that human partners will tend to do what they are required to. Then we can approach the collaborative tasks in a way similar to multi-agent task planning, where the planner computes actions to perform for the robotic agents as for the human ones. In most cases we don't need to plan as precisely for humans, since they cannot be controlled like a robot; but actions can be dispatched to human, eventually after checking their feasibility.

When robots are at the service of humans (hopefully they always are) they interact with, it is important that all planners, when relevant, be aware of the human objective. Unlike many decoupled approaches of plan refinement, we think that better robot behaviour are achieved when all levels of planning share the information on what the task is, and hence on what is the global objective.

Let us consider a robot which is serving in a hospital [4] to assist personnel by carrying objects (devices, medicines, food,...) from places to places. When required to bring a specific object to someone, the robot can either handover the object to the person or place it nearby. In each case, it needs to decide where to go to handover or place the object, plan how to get there, actually get there, plan the arm motions and execute them while controlling when to release the object (particularly important when performing a handover). Here we simply want the reader to get the intuition that every of these six features have to be aware of some information about the global task, and not only about their own subset of the problem. Consider the following scenarios:

— In the operating room, the surgeon request the scalpel (environment with obstacles but only accustomed personnel and an unconscious patient under the sheets, dangerous object, probable urgency of the task);
— The robot is serving lunches to patients in their rooms (environment with persons inattentive and unusual to the presence of the robot, hot dishes, sleepy and clumsy patients, low urgency,...);
— The robot is carrying blood for a urgent transfusion, an employee guides it and open the path in the hospital corridors (crowded environment with surprised persons, some of them being fragile, one of them is in charge of choosing the path to take and warn people, the personnel will snatch the

4. Hospital scenario are particularly interesting to me because there are many people with different or opposite roles and capabilities

bags from the robot on arrival)

We can see from these (caricatured) examples that a decision can be taken correctly only if each component of the system is aware of the global task, otherwise surgeon hands would be cut, unwary visitors get pushed, weak patients get burnt and blood bags get torn apart.

This is obvious to most of us, roboticists implement systems that can safely handover a warm bowl of soup to an weakened person or navigate in crowded environments. But when it comes to create a versatile system such as the imaginary hospital robot, we must guarantee that the robot uses the right method at the right time, and this implies sharing information between really different components that usually operate on very distinct types of information and poorly communicate, or even do not communicate at all.

1.5 Summary of Work

We started working on the Multiple Agent Handover task (Chapter 3), related to many topics of robotics research: human-robot collaboration at the task and motion level, handover, human-aware navigation. There the planner takes into account many information about the scenario and partners to find a plan that complies to users. The planning is actually split into a few levels with different search spaces: task level, navigation, manipulation, motion-planning. While the overall approach is satisfying and gives good results, the motion algorithms needed improvements. We focused for a while on optimal motion planning with a focus on the improvement of the robot motion dynamic, and adaptation of the motions, both off-line and in real-time. All this with the objective of making robots motion more acceptable to humans (Chapter 7). Back to task and motion planning, I contributed to a framework for a better connection of symbolic planners and geometric reasoners (including motion-planners), still with the objective of improving robot acceptability and efficiency in human-robot collaborative tasks (Chapter 6). In the same spirit as the Multiple Agent Handover task, we got interest in a guiding task where actions for both human and robot are planned, and motion are generated taking into account the global task (Chapter 4 & 5).

1.5.1 List of Publications

1.5.1.1 Published

— Jules Waldhart, Mamoun Gharbi, and Rachid Alami. Planning Handovers Involving Humans and Robots in Constrained Environment. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6473–6478, Hamburg, Germany, September 2015. IEEE.

— Jules Waldhart, Mamoun Gharbi, and Rachid Alami. A Novel Software Combining Task and Motion Planning for Human-Robot Interaction. In *AAAI Fall Symposia*, Washington D.C., 2016.

1.5.1.2 Submitted

— Jules Waldhart, Aurélie Clodic and Rachid Alami. *The SVP Planner: Planning Human-Robot Shared Visual Perspective to Provide Route Direction.* Submitted to HAI '18 the 6th International Conference on Human Agent Interaction. ACM, 2018.

1.5.1.3 Accepted

— Jules Waldhart, Aurélie Clodic and Rachid Alami. *Planning Human and Robot Placements for Shared Visual Perspective.* In Robotic Co-workers 4.0: Human Safety and Comfort in Human-Robot Interactive Social Environments Workshop at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018

Chapter 2

State of the Art

Contents

We will present an overview of work related to the planning and execution of collaborative tasks involving humans and robots. We will first review contributions in the domains of planning: motion-planning, task-planning and their merging. Work specific to human-robot interaction will then be presented, leading us to a third part where an overview of planning techniques for human-robot collaborative tasks will be provided.

2.1 Task And Motion Planning

Robotic systems integrated in our society as service robots or assistants are required to deal with the complexity and variability of their environment, while performing their tasks safely and robustly. These robots need to plan and reason at a high level to decide actions to take according to their objective and environments. But they also need to check the feasibility of their plans regarding collisions, stability, dynamics, deadlines, and so on. Usually, high-level planning is solved by searching a sequence of actions, which prerequisites and effects are already known, that transform the environment into a state that satisfies an objective [Ghallab 2016]. On the other hand, physical level planning, where feasible motions are generated, usually link two fully defined physical states with a motion [LaValle 2006]. This leaves a gap between the symbolic actions found at the task level (high level), like "*grap the screwdriver*" or "*exit the room*", and the requests to communicate to motion planning, which would be "*move gripper from A to B, then close gripper*" or "*navigate from* (x, y) *to* (x', y')". This problem of finding a motion for a high level action specification is called *refining*. The problem often called of "Task And Motion Planning" (TMP) addresses the generic issue of finding a feasible motion plan that satisfies a symbolic objective.

The integration of different levels of reasoning is crucial to deal with increasingly complex systems, environments and tasks. The recent increase of publications addressing this particular issue shows that it is getting a high interest. This problem have been given a number of names and solved by a multitude of methods, [Erdem 2016] and [Gharbi 2015a] present an overview of the literature in this domain and propose classifications of such methods. We will briefly overview some of the techniques that have been proposed to solve the TMP problem. Some approaches refines the symbolic plan down to the geometric level in a depth-first manner [Dornhege 2012]. Other approaches consist in first searching for multiple high level plans and then check them against geometric constraints to finally produce a feasible motion plan for one of the high level plan [Şucan 2012, Lagriffoul 2014]. A similar approach is to first search for a symbolic plan, then refine it to the geometric level, and when actions are infeasible, use that information to guide the search at the symbolic level to produce a new plan [Lozano-Pérez 2014, Srivastava 2013, Caldiran 2009]. In some work, including [Zickler 2009, Nedunuri 2014, Garrett 2014, Plaku 2010, Barry 2013], the search for a motion plan directly occurs at the geometric level, using symbolic information or

plan outline to guide the search. Finally, [Hauser 2009, Cambon 2009] propose solutions where the search is done in both levels simultaneously.

Our work accounts for these issues and we inspired from some techniques used to address the problem of joining task and motion planning. Indeed we aimed at developing "*task-solvers*" that would be able to refine a symbolic request to a physical, feasible solution, which is a core issue of Task and Motion Planning. But in our case we focused on specific tasks and propose specialized solvers (MHO solver and SVP planner, respectively presented in chapters 3 and 4), that could however be integrated in more generic TMP frameworks.

2.2 Human-Robot Interaction

Besides planning for acting in complex environments and for complex tasks, future robot developed to share human environment must behave in an acceptable way. The field of Human-Robot Interaction (HRI) have addresses many topics related to how a robot interacting with humans should move, take decisions, communicate, and so on. HRI is highly related to human psychology as we need to understand humans behaviours and expectations to create engaging robot that are efficient in social interactions. We will first present in this sections work studying the mechanisms humans use to achieve common goals as collaborators: Joint Action Theory. Secondly, we will focus on work related to generating robot motion for such collaborative tasks with humans.

Some more specific aspects of the literature will be treated chapter-wise: human-robot handover is addressed in 3.2, and work related to guiding and providing route directions in 4.3.

2.2.1 Human Joint Action Theory

Understanding how humans perform joint action can help us to endow robot with the ability to perform such collaborative interaction with humans. We will use the definition of Joint Action proposed by [Sebanz 2006]:

> Joint action can be regarded 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.

To achieve joint action, the individuals need to agree on what "change in the environment" they want to bring, how they will do it and on which conditions they remain engaged in the task. This relates to commitment that we will develop in Section 2.2.1.1. As stated in the above definition, the agents need to coordinate with each other, which in turn requires they perceive and predict their teammates actions. We will develop this in Section 2.2.1.2.

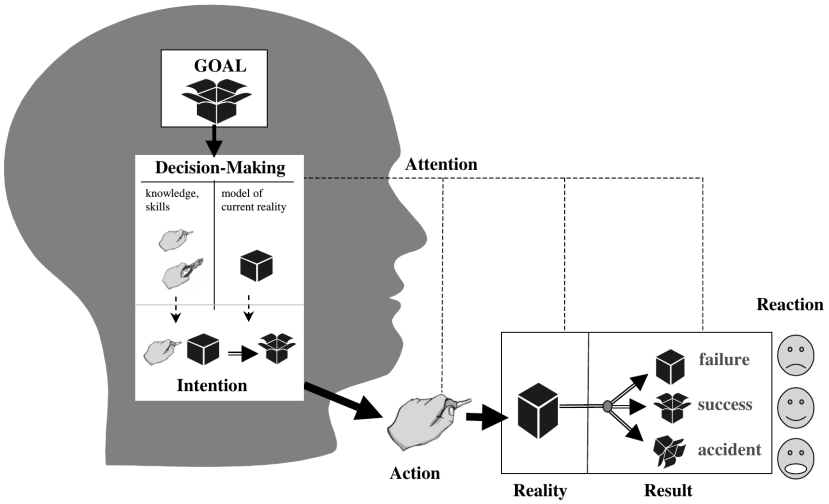


Figure 2.1: Example from [Tomasello 2005]. The person forms an intention from her goal, knowledge and skills.

2.2.1.1 Commitment

For an action to be carried out by an agent, she has to have a *goal*, *i.e.* a representation of the desired state of the world, and an *intention* to achieve it. For a single individual, [Tomasello 2005] presents a task where a person has to open a box. The *goal* is the open box, and the *intention* is to use a key to open the box (Figure 2.1).

The definition of intention in [Cohen 1991] differs from the one of Tomasello *et al.*: they define it as the commitment of the individual to achieving the goal. However in their definition, intention still relates the goal.

We can now extend these definitions from individual action to joint action. A definition of *Joint intention* is proposed by [Bratman 1989]:

> We intend to J if and only if:
>
> 1. (a) I intend that we J and (b) you intend that we J.
> 2. I intend that we J in accordance with and because of 1a, 1b, and meshing subplans of 1a and 1b; you intend that we J in accordance with and because of 1a, 1b, and meshing subplans of 1a and 1b.
> 3. 1 and 2 are common knowledge between us.

[Tomasello 2005] extend their example of the box opening to this definition (Figure 2.2). There one agent holds the box with a clamp while the other open it with a cutter.

The *shared goal* is defined as the representation of a desired state plus the fact that it will be done in collaboration with other person(s) and a *joint intention* is defined as a collaborative plan the agents commit to in order to achieve the shared goal and which takes into account both agents individual plans.

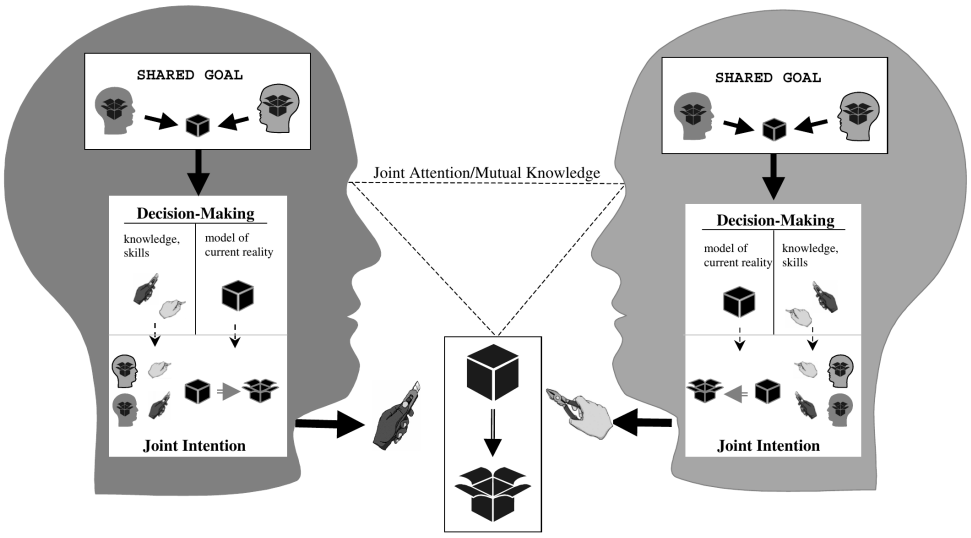


Figure 2.2: Extension of the example of Figure 2.1 [Tomasello 2005] for a collaborative action.

For [Cohen 1991], the *joint intention* is achieved when members mutually know that they all want to reach the goal. Their definition do not cover *how* agents will achieve the goal, but interestingly they state that agents must be engaged in informing each-other about the status of the goal (achieved or not, or irrelevant to the task).

Concerning the way to achieve the *shared goal*, the notion of *shared plan* is introduced by [Grosz 1988, Grosz 1999]. In their definition, there exist a plan to reach the shared goal, but no individual necessarily know the whole shared plan, each individual need to know only its part of the shared plan.

2.2.1.2 Coordination

Psychologists study mechanisms and signals supporting the coordination, [Vesper 2017] is a recent review of the work in that field.

We presented that co-actors of a joint action need to share a common goal and the representation of each-other tasks. They also show a need for monitoring other's actions. The co-actors can make use of several sensorimotor information that increase the overall performance. Co-actors are influenced by each-other gaze, knowing their attentional states can provide help to predict their actions and infer if they are aware of some elements of the scene [Pierno 2006, Emery 2000]. Also, [Lallée 2013] present an experiment where robot successfully communicates shared plans to a naïve human partner using gaze only. [Boucher 2012] shows how a human partner actions can be improved when she perceives the observer gaze (human or robot) and that she can deduce the observer is looking at her performing the action. Sensorimotor prediction is also a key to coordinate the actions of a joint ac-

tion [Blakemore 2001], and co-actors can adjust their motion to make them easier to predict [Vesper 2014]. Information is also conveyed via the tactile channel (haptic coupling); multiple senses can be redundantly used; co-actors express and understand emotions [Vesper 2017]. Co-actors can perceive other's affordances, that is the possible actions available to them with respect to the environment [Gibson 1977].

Concerning our work, joint action theory can help us (1) design robot behavior that satisfies the need of information from the co-actors, (2) monitor and understand human signals, (3) design effective systems that make profit of human's high joint action capabilities to increase task efficiency and user satisfaction and comfort.

2.2.2 Generating Motions for Acting Around Humans

Generating robotic motion is a complex task, even when only functional motions are needed: [LaValle 2006] provides a good overview of the difficulties and existing approaches of the motion-planning problem. Difficulties include high dimensional space, collisions, torque limits, dynamics constraints [Boeuf 2014, Li 2015] and often constrained end-effector pose [Stilman 2007, Koga 1994, Ahuactzin 1999, Yao 2005]. However recent improvements of CPUs and optimization algorithms now allow to generate motions that optimize costs [Jaillet 2008, Devaurs 2013, Zucker 2013]. Motion cost in our case would reflect properties required to generate motion in line with what humans expect from an acting robot.

There are a number of contributions for navigation actions [Kruse 2013, Rios-Martinez 2015]. Concerning manipulation, the contribution are often limited to a specific action such as the hand-over [Cakmak 2011b, Huang 2015, Kupcsik 2016] or to a few properties of the manipulation motion such as safety, comfort [Sisbot 2012], legibility [Dragan 2013a], visibility [Sisbot 2007b], timing [Zhou 2017]. Let us overview some of the important criteria of human-aware motion generation and how they are addressed in the robotic literature.

2.2.2.1 Safety

Safety is the primary concern when designing robots that work into human environment. Common approaches include respecting some safety distance with human, eventually predicting their motion to plan accordingly [Kruse 2013, Rios-Martinez 2015]; the distance also relates to the *proxemics* theory [Hall 1969], which can be used to increase the comfort of humans. Most of the human-aware motion and navigation planners are still built with a proxemics cost as a basis. But recent planners are exploring the other important aspects of collaborative motions.

2.2.2.2 Legibility and Predictability of Robot Motion

Motions in a collaborative task can convey information, even if their primary objective is not communication. [Dragan 2015] defines three attributes of motion: functionality, predictability and legibility.

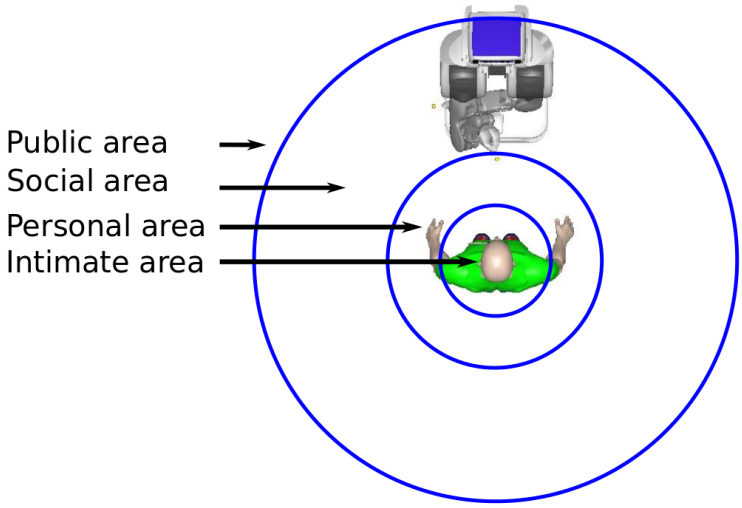
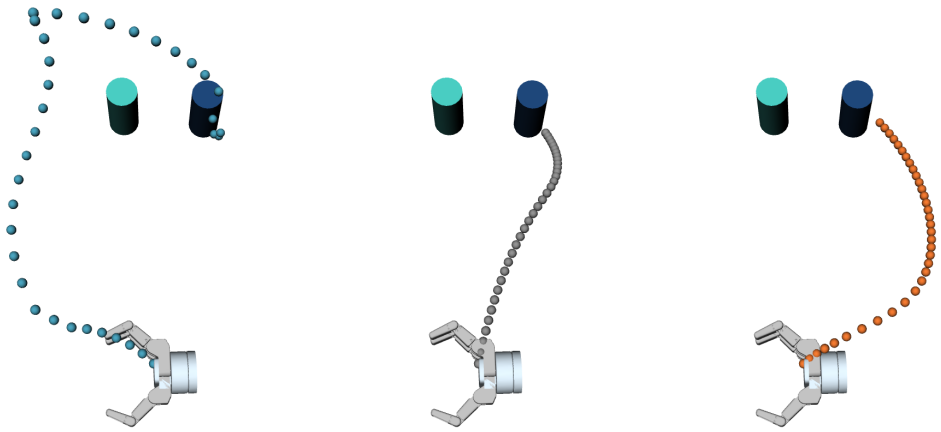


Figure 2.3: The four *proxemics* area around the human as defined by Hall [Hall 1969]. A robot interacting with a human should stay in the "Social Area".



Functional Predictable Legible

Figure 2.4: Images from [Dragan 2015] highlighting the differences between functional, predictable and legible motion.

Definition 2.1 *Functional motion is motion that reaches the goal and avoids collisions.*

Such *functional* motion may not even be the most efficient one, but it gets the job done. But when acting in collaboration, as discussed above, partners create inferences about the motion: they infer *goals from motions* and *motions from goal*. The human has an expectation of the motion *knowing the goal*, and a motion matching this expectation is *predictable*:

Definition 2.2 *Predictable motion is functional motion that matches what the collaborator would expect, given the known goal.*

The partner can infer the goal of the ongoing motion, a motion making this easy is *legible*:

Definition 2.3 *Legible motion is functional motion that enables the collaborator to quickly and confidently infer the goal.*

Figure 2.4 shows three motion where a robot has to grasp the object on the right-hand side, each motion is represented by the points on their path, and each of them exhibit one of the above attributes. In the functional motion, the gripper makes a few curves in the workspace before reaching the goal; the predictable motion is rather straight towards the goal; and the legible motion exaggerates its curve so the observer can discriminate between the two target early during the motion execution.

The same publication presents results from a user study showing that motion planned to be *legible*, that clearly express the robot's objective, makes the collaboration more fluent and improves the perception of the task by the partner. They use a motion planner that can explicitly take into account motion legibility for their experiment, which is from the same authors [Dragan 2013b].

In [Kruse 2012] they develop a navigation planner inspired on how humans cross each-other to reach a better robotic motion legibility in similar situations.

2.2.2.3 Visibility of the Moving Robot

While moving, a robot, or robot part, can become visible or hidden to its partners. The visibility status is important to consider while moving, but the change of visibility status has an important impact on the acceptability of the motion. [Koay 2007] is a study showing that a robot should avoid to get close to human while not visible by them, like when approaching from behind them or when appearing suddenly from behind an obstacle close to the human. [Sisbot 2007b] implements a navigation planner explicitly taking into account the visibility of the robot from the human partner perspective.

2.2.2.4 Human-Robot Shared Attention

Coordination in a joint action is partly supported by joint attention. Systems to enable robot to share attention point with humans have been developed and studied. For example in [Marin-Urias 2009] they can compute what the human is looking at to make the robot look at it, hence creating a shared attention initiated by the human.

Robots directing human partner attention thanks to gaze hints were successfully implemented, like in [Mwangi 2018].

2.3 Plans Involving the Human Partners

Few recent studies now focus on planners able to generate plans for the team, meaning that the robot creates a plan where humans are assigned with specific tasks needed to reach the common goal.

The idea of planning not only for robots but also for the humans cooperating with them is not intuitive. Indeed the system cannot control the humans. However the Joint Action theory teaches us about shared goal, joint intention and shared plan, and relating to robotics [Clodic 2017] studies what are the key features needed for human-robot collaborations, which include the reasoning about shared plans. The joint action exists because agents are willing to share the effort, meaning that human partners are acting too. When building their own part of the shared plan, agent are then trying to foresee what partners will do, thanks to knowledge on their skills, knowledge and affordances. This is done at the task level by [Alili 2009, Lallement 2014, Lallement 2016] with the Hierarchical Agent Task Planner (HATP). Combined with the Geometric Task Planner (GTP) [Gharbi 2015b], the system plans the motion of the partners for the task assigned to them by HATP. Functional systems based on HATP planner are presented in [Lemaignan 2017, Devin 2016].

On a more specific task, [Mainprice 2012] propose a planner that takes into account the effort sharing to build a plan. They search for the best place to go and handover an object to a human partner, considering how much effort she his willing or able to provide to accomplish the task. They take into account the distance run by the human and the comfort of the handover posture. Their planner can then produce very different plans depending on the task urgency or the human physical condition or desire to be assisted vs. to be proactive. Figure 2.5 shows an example of this, where we can see that in order to share effort, their planner plan a motion for the human partner, in order to evaluate its feasibility and cost.

Concerning navigation, a robot can proactively propose a co-navigation solution to the human. In [Khambhaita 2017] they show that human-robot navigation can be viewed as a joint action in some constrained situations. They propose a planner that generates trajectories for both partners in corridor-crossing scenario. The trajectory is designed to be legible, so the human can understand the solution proposed by the robot. The pertinence of this work have been recently confirmed by a user study

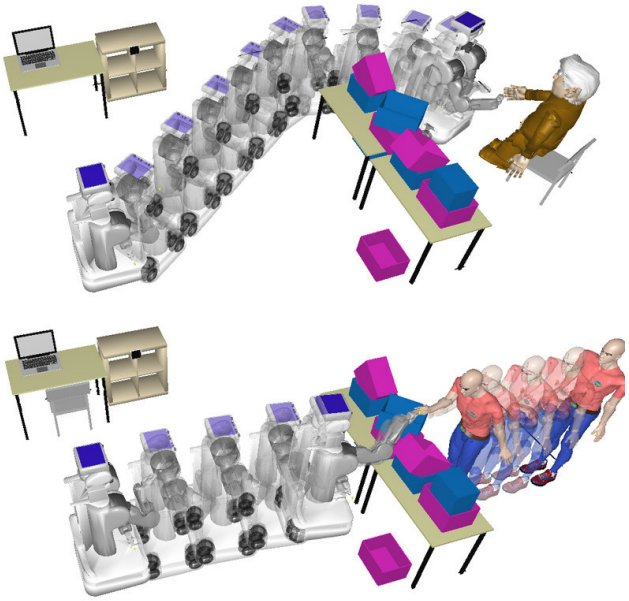


Figure 2.5: Illustration from [Mainprice 2012]: for the same task, the planner produce two very different solutions; in the first case, the person to give the object to is a seated old man, so the robot takes all the effort to bring the object to him; while in the second case, the person is already standing and is mobile, so the effort can be shared.

conducted in our team, which results will soon be published.

Chapter 3

Multiple Handover Task Solving

Planning where to perform successive handovers to solve a transport problem with robots and humans

---

Contents

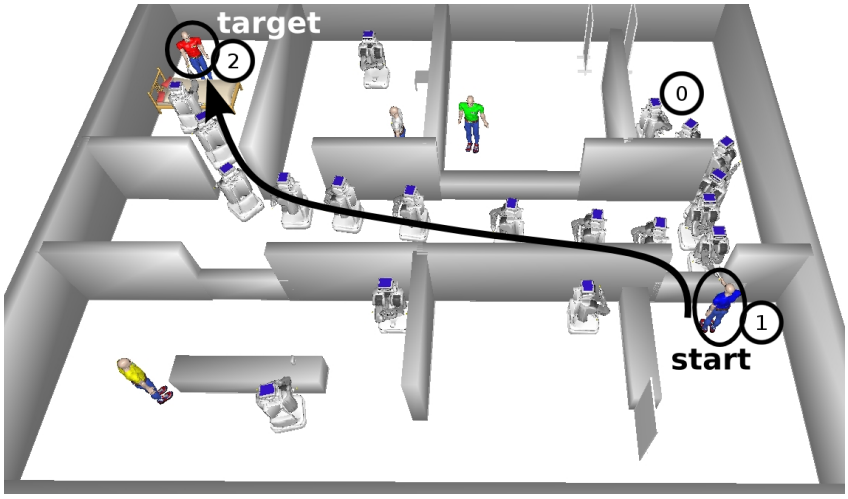


Figure 3.1: An example in an environment where agents are separated into two navigable zones linked by windows and counters. The blue human is in possession of an object needed by the red one. The solution found by our algorithm is: robot 0 navigates to the blue human, takes the object through a handover over the counter, and then navigates to the red human and gives it to him.

3.1 Introduction

Our environment is made of objects we manipulate and that are often passed from hands to hands, and we, humans, are proficient in performing so-called (physical) *handover*. Hence a human-robot collaboration key feature is the handover, particularly for robots required to perform fetch-and-carry tasks, coworker robots in workshops, service robots, and more... During the last decade, studies about when, how and where to perform a handover have been conducted ([Strabala 2013, Cakmak 2011a, Cakmak 2011c, Chan 2012, Dautenhahn 2006, Mainprice 2010, Mainprice 2012]).

We address here a related problem which is a transport task, that may involve several agents (robots or humans). The transport problem is defined as an object to be moved from an agent to an other specific one, with the help of several agents present in the environment. The solution is a series of handovers between some agents. Figure 3.1 shows an example of this problem, the agents are separated by a wall where windows and counters allow objects exchanges.

This transport problem involves several decisions such as which agents to use and where to perform the handovers. An important issue to address, as illustrated in Figure 3.1, is to preserve the humans comfort. In the figure, a handover is possible between initial and target humans, but using the robot reduces the human efforts.

This kind of problem can be solved using a combination of symbolic and geometric planning ([Karlsson 2012, Dornhege 2009, Kaelbling 2011]): these approaches will solve the problem, but would be too slow to allow on-line use of the algo-

rithm. (note that using a task planner alone will be under efficient as the problem is geometrically complex [Lagriffoul 2013]).

The main contribution here is the design of a planner interleaving strongly various models, from task level to geometric computation. This planner should be able to find efficiently high-quality solutions based on a number of parameters related to social rules and humans comfort. The system has been implemented in simulation and also using two PR2 robots and two humans in a cluttered environment.

In section 3.2 we describe the related work while in section 3.3 we define precisely the problem. Section 3.4 addresses the formalism and the models used in the resolution. In section 3.5 we present the results found in several environments and finally we conclude in section 3.6.

3.2 Related Work

To properly solve the task, we must find which agents have to move, where to move, and how to perform the handovers. We also need to develop a task solver to plan for a solution addressing these questions, and a system to execute the task.

Our multi-agent transport problem is related to a number of categories of work: handover, human aware navigation, pick-up and delivery problem, and manipulation planning.

3.2.1 Handover

Difficulties appears for handovers between human and robot, concerning human security, comfort and other social rules, to make it as natural as possible [Cakmak 2011a, Cakmak 2011c]. [Moon 2014] explore how robot-human handovers can be made more *fluent* (hence more efficient) thanks to non-verbal cues provided through gaze. The problem of detecting when a human intends to give an object to a robot have been adressed in [Pan 2017] (the reader can also refer to the thesis dissertation of M. Pan [Pan 2018], dedicated to human-robot handover. Also the problem of deciding when the object should be released by the robot have been studied, mostly using force detection during the handover [He 2015]. For a handover between two robots, synchronization is essential [Quispe 2014].

3.2.2 Human aware navigation

The transport task in our context requires the robots to navigate in a space occupied by humans, who may be busy. In this context, the robot will need to navigate among them while creating the least disturbance possible. State of the art related to navigation was already presented in Section 2.2.2 (Chapter 2)

3.2.3 Manipulation Planning

Manipulation planning [Simeon 2004, Barry 2013], an extension of classical sampling-based motion planning [LaValle 2006] could solve the transport problem.

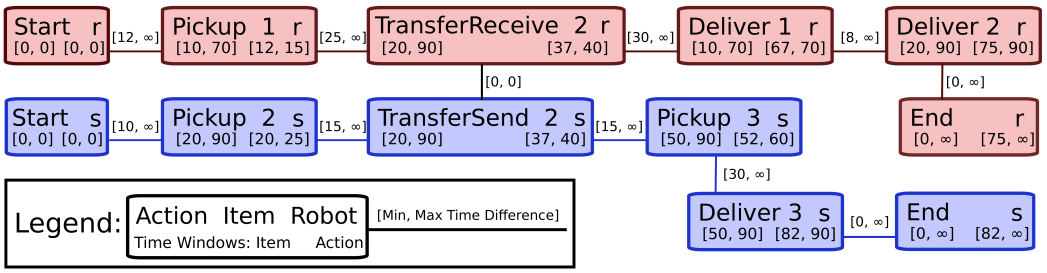


Figure 3.2: Illustration from [Coltin 2014] where they plan for multiple Pick-up and Deliveries where objects to deliver can be transfered between robots to optimize the plan. The image shows a temporal network of task with two robots, three items to deliver and one transfer.

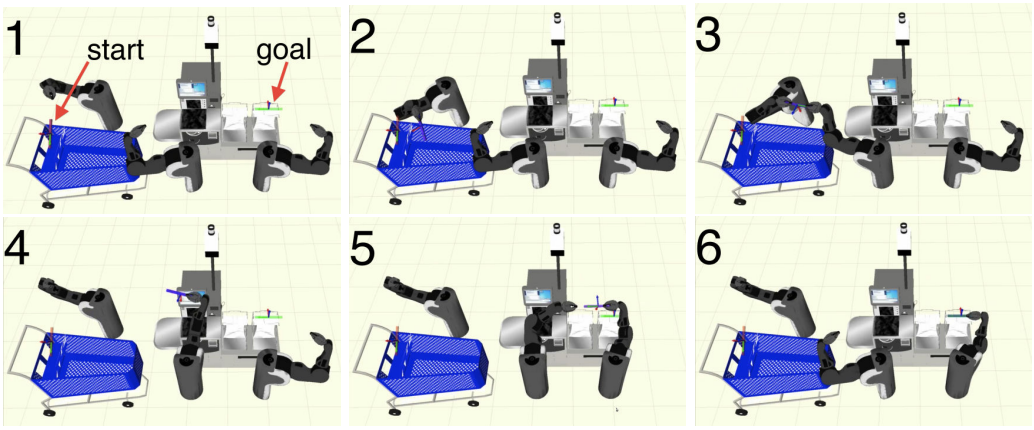


Figure 3.3: Images from [Cohen 2014]: their planner finds a sequence of handovers for n arms to bring an object to a goal position. Images above show successive steps of the solution: initial, pick, first handover, transfer, second handover, place.

Nevertheless, it would not be able to deal efficiently with the complex costs involved in the task, and the very high dimensionality of the search space (see Section 3.3).

3.2.4 Pick-up and delivery problem

The transport problem is close to the pick-up and delivery problem (PDP) which have been widely addressed in the literature. In [Savelsbergh 1995], the authors present a survey of the problem types and solution methods for PDPs. More recently, the link between PDP and object transfer has been stressed out with an algorithm where robots transfer objects to optimize a PDP plan [Coltin 2014] (an illustration from that reference is shown in Figure 3.2).

3.2.5 Multiple agent handover

Recent work related to the multi-agent handover problem uses a heuristic to guide the search through possible handovers [Cohen 2014] (see Figure 3.3). The

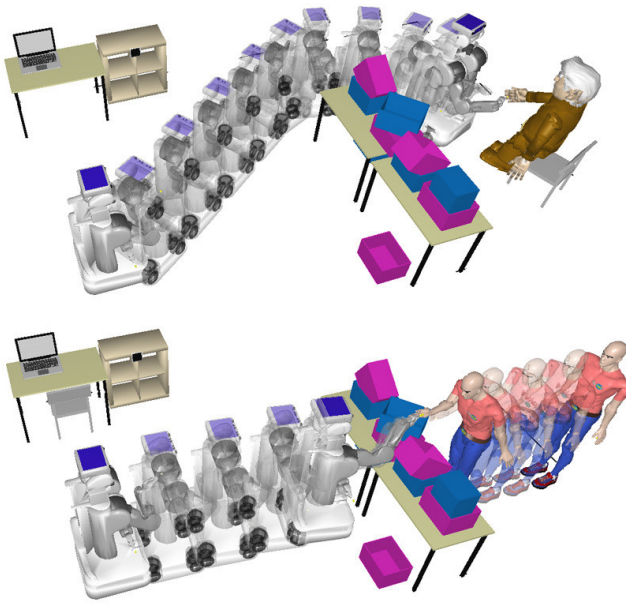


Figure 3.4: Illustration from [Mainprice 2012]: for the same task, the planner produce two very different solutions, by taking into account a parameter specific for each agent called *mobility* that reflects how much effort can be taken by the agent. In the top image, the old man has a low mobility, so he is *served* by the robot, while in the bottom image, the man has a mobility similar to the robot mobility, so effort is shared to accomplish the task.

difference with the previous approach is that handovers are required in order to perform the task, and do not result from an optimization, as robots are fixed manipulator arms. The heuristic used takes into account cost estimation of the handovers and an approximation of the number of remaining handovers that will be required. They also introduce a lazy variant of weighted A*.

In [Mainprice 2012], the authors present an algorithm able to share the efforts between a human and a robot depending on the human preferences. In some cases, the human might prefer moving toward the robot, while in other cases he will prefer waiting for the robot to come all the way to him. Even in this last case, if no solution is found, the robot might ask the human to move to a place carefully chosen in order to successfully perform the handover. This approach is limited to single handovers and doesn't scale to series of handovers. We propose a similar approach adapted to multiple agent problems where it is important to choose the agents to involve and the place to transfer the object. We use the same concept of *mobility* they introduce: mobility is a scalar factor that is used to weight how much effort we want an agent to provide with respect to each other. An agent with low mobility (near zero) will be mostly served by other agents, providing the minimal effort required for the accomplishment of the task, while an agent with high mobility will provide most of the effort (see Figure 3.4).

3.2.6 Rationale

We present in the following a tool that produces a plan solving a transport problem possibly involving multiple heterogeneous agents, including robots and humans. The plan has to observe constraints related to human safety and acceptability. The efficiency of such task is dependent on the ability of the robots to communicate with humans, especially providing non-verbal cues, so the plan shall be generated with human-centered algorithm. The plan should optimize human efforts and comfort while respecting task constraints.

This chapter presents a functional system developed with this objective. The task solver shall explicitly take into consideration human-related constraints and objectives, but is not intended to find solutions to sub-tasks like navigation and handover; it relies on external tools specific for these purpose, which are called during the search. So we present here the top level planner and its link with the other specific components mostly related to human-robot collaborative motions.

We insist on the fact that this work is not focused on algorithm to decide how to perform a handover, but rather an algorithm that selects where to perform handovers and with which agents. However it takes into account *how* the handover is performed, but thanks to external components. As we will present later, the planner also decide *when* to perform the handover.

So to sum up, our planner shall decide:
— *where* to perform successive handovers;
— *who* will perform them;
— *when* they will be performed;

— *how* handovers are performed, thanks to external tools.

3.3 Problem definition

The problem is to bring an object from an initial position to a target agent, having at our disposal several agents able to carry the object and to hand it over to other agents. The goal is to find optimal solutions in a reasonable amount of time allowing an on-line use.

The object is held and moved by a unique agent, and can be transferred to other agents through handovers.

The problem inputs are the agent list, the initial state, consisting on all agents and objects positions, and agents specific information about speed and availability. The target state is defined by the target agent (also an input) holding the object. The algorithm should also take into account the task urgency and the comfort of humans.

A solution to the problem is a scheduled sequence of actions (navigation or handovers), that bring the object from the initial state to the desired final state.

The search space is the full configuration space [LaValle 2006] of the whole problem. As it involves several agents, it can be written as the cross-product of the configuration spaces of each agent: $C = C_0 \times C_1 \times \cdots \times C_n$. As it is, the problem high dimensionality makes it nearly impossible to find a solution based on classical algorithms. E.g. in the environment of Figure 3.1, we have $card(C) \simeq 300$. Note that we do not take into consideration the object itself for navigation, we assume it is small enough to have no influence on the navigation of its holders.

3.4 Model and resolution

The full representation of the problem is not suitable for on-line solution search. We break down the main problem into problems of lower dimensionality: we distinguish two subsets of the main problem, (1) navigation between handover places, and (2) handovers themselves. However we take into consideration the influence of the decision of where to go to perform the handover on the quality of the handover itself, like pointed out by [Sisbot 2007a]; indeed we search for a solution optimizing the quality of the whole solution to the transport problem.

Navigation is simplified to path-finding in a discrete 2D grid. This 2D model is based on the input environment and agent geometries, but is built off-line and does not affect running-time. Besides, we do not take into account inter-agent collisions during the search phase. For that, we make the assumption of a large environment with sparse obstacles and few (or no) narrow passages. Thus, we can consider only one agent at a time for navigation. We explain in Sec. 3.4.4 how we deal with violations of this hypothesis.

Handovers involve two agents and we need the full dimensionality of their models and their positions to find a solution. The planner can treat situations where the

object has to be handed over a table, through a window or even a narrower passage. This computation is expensive, specifically when the handover is not possible. Thus, we limit the number of calls to the handover planner to the minimum, and try to detect impossible situations with faster techniques.

In addition to that, the problem has a higher level discrete component: finding a sequence of agents to perform successive handovers which guides the search toward relevant handovers, limiting calls to time-consuming evaluations.

In this section, we present formally the models we use, and the algorithms involved in the solution.

3.4.1 Representation

We solve the problem as a path finding problem in a *main graph* G. We consider nodes being simplified states containing information only about the agent holding the object at that time, while others are considered to be at their initial positions. A node is then a pair formed of an agent identifier and a discrete 2D position: $[a_{id}, \{x, y\}]$. We define two actions that cause the current state to change –*i.e.* to go from a node to another: an elementary navigation action and a handover action.

— *Navigation*: an agent A holding the object can travel to a neighbour position: $[A, \{x, y\}] \rightarrow [A, \{x + \delta_x, y + \delta_y\}]$;

— *Handover*: the holding agent A gives the object to another agent B: $[A, \{x, y\}] \rightarrow [B, \{x', y'\}]$. This requires B to move from its initial position to its new one ($\{x', y'\}$), and causes A to go back to its own initial position.

An edge between two nodes of G exists when there is an action leading from a state to another. Edges are weighted with the costs of the actions. The cost of a handover action is high compared to a navigation action, as it involves two auxiliary navigation tasks along with the handover itself.

This search algorithm relies on other models: high level representation, 2D model for navigation, and full geometric representation for handover posture search and check, collision checking and motion planning.

High-level graph This graph guides the search through all possible handovers. We will later refer to it as the *agent graph* G_A (different from the *main graph* G). In this graph, the nodes are the agents, all linked with edges representing the possibility of a handover between the two linked agents. The model is initialized with all the handovers set as possible (edges between every pair of nodes), and each edge is weighted with an optimistic estimation of the cost, based on time needed to perform the handover and the optimal human-related cost expected, independently of the environment. While the search will be pursued, the costs are adjusted and the edges may be removed in case of inability to perform a handover.

2D navigation grid To plan the navigation tasks. It is used for navigation steps that are not directly explored in G, these are the navigation relative to a handover,

to estimate the cost for the giver to go back to initial position, and for the receiver to come to the handover position.

Geometric environment model We use geometric algorithms (e.g. collision checking, inverse-kinematics, motion planning) to find valid handover positions and their cost related to comfort, acceptability, legibility and so on. A valid handover is a collision-free position for both agents, where the object can be transferred [Mainprice 2012]. All the process of finding and evaluating a handover will be referred as the handover search tool.

Model relations Figure 3.5 shows the main components of the architecture and their interactions. The main planner (searching in G) is linked with the three other models, and their respective planners, but there are also interactions between every planner. Each model constantly evolves on the basis of lower level model evolutions. For example, edges in the agent graph G_A are removed when the geometric tools have tested these handovers and found none is possible.

Cost The cost function defined to evaluate a solution is as follows, with several f being factors to stress different parameters, $f_d(a)$ is a per agent factor, $t_{use}(a)$ is the time an agent a is mobilized for our task. t_{begin} is the date the first agent starts moving and t_{end} is when the last agent reaches its target position with the object. Each handover has a cost $c_{HO}(i)$ related to comfort and safety. (n is the number of handover in the solution, A is the set of available agents)

$$c = f_{use} \cdot \sum_{a \in A} t_{use}(a) \cdot f_d(a) + f_{time} \cdot (t_{end} - t_{begin}) + f_{HRI} \cdot max(c_{HO}(0), \ldots, c_{HO}(n))$$

3.4.2 Resolution

Our main model is the graph G representing the problem. The use of a search algorithm in this graph is still problematic as the number of expansions can be huge, and computationally expensive. In the rooms environment (Figure 3.1) the number of neighbours of a node reaches 3000 in average [1]. The use of a classic A* would cause evaluating all these handover actions, which is computationally intractable.

1. The number of neighbours of a node can be approximated as follows. Let A be the agent holding the object in that node, and n its corresponding node in G_A, so $n.ag = A$. N is the set of neighbour nodes of n in G_A. Let $R(X, Y)$ be the maximal distance for a handover between agents X and Y, and $r(X, Y)$ the minimal distance (collision free). d is the discretization step of the 2D grid. The number of neighbours reachable by a handover action from each node involving A in G is then at most

$$\frac{1}{d^2} \sum_{p \in N} \left(\pi \cdot R(A, p.ag)^2 - \pi \cdot r(A, p.ag)^2 \right)$$

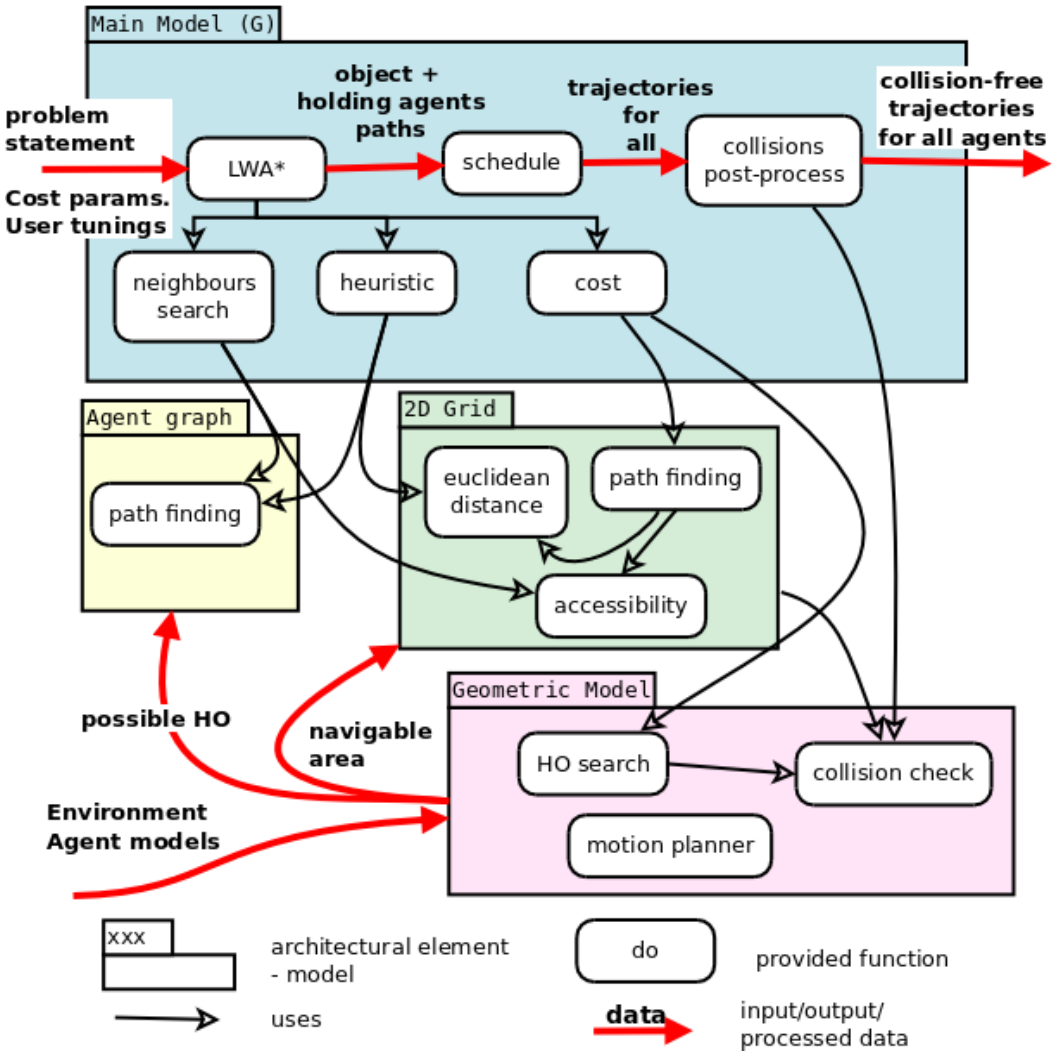


Figure 3.5: Overall architecture, showing interactions between the different level planners and models, and their main functionalities.

To reduce the number of handover evaluations, we use a lazy variant of A* which gives a first estimation of a handover cost, sparing the expense of a call to the handover search tool. The real cost will be computed only if the handover appears to be relevant. We use as search algorithm the Lazy Weighted A* (LWA*) variant [Cohen 2014]. This algorithm has proven bounds of sub-optimality inherited from Weighted A* [Likhachev 2003], and can perform faster when it involves computationally expensive evaluations. LWA* algorithm is based on A* algorithm, which searches for the shortest path using a heuristic. When a node is expanded, the son nodes are given three values: the g value is the distance (cost) between the origin and the son node, the h value is the heuristic, *i.e.* the estimation of the distance (cost) remaining to reach the target, and the f value is the sum $g + h$. In the next iteration, the unexpanded node with the smaller f value will be expanded, until the target node is reached.

In the weighted variant, the h value is increased by a factor, f becomes $g + \varepsilon.h$ with $\varepsilon \geq 1$, thus adding a depth-first flavour to the search, but decreasing the quality of the solution of at most that factor ε.

The lazy variant gives to expanded node sons a temporary g value, which is optimistic and faster to compute than the real cost. It computes the real cost only when the node is selected to be expanded, *i.e.* is the one with the smaller f value. Its g and f values are updated and it is put back in the list of nodes to be expanded.

This approach can save a substantial number of costly evaluations, postponing them to when they are really needed. In our case, such costly evaluations are those of the handover search tool.

3.4.2.1 Heuristic and cost

The cost function evaluates an edge between two nodes n_A and n_B of G. There are two cases, according to which action is represented by the edge: elementary navigation or handover. For the navigation, the cost is only distance related (energy, time). For the handover, we need to evaluate the motion with the geometric tools and estimate the cost based on the humans comfort and the action duration.

The heuristic function (Algorithm 1) guides the search through the environment and the possible handovers. It is based on G_A and the navigation grid. It first searches for an agent sequence that can bring the object to the target agent. This search is made in the agent graph and takes into account minimal handover costs and no navigation (line 2). Then, on the basis of this agent sequence, it searches the minimal cost related to the navigation. In our model, it is using the cheapest agent for the whole distance (in the Euclidean sense, line 4). At this step, it is not known yet if it is possible or not, but it guarantees that the heuristic is admissible. It then adds the estimation of the handovers costs, which must be computed with an admissible heuristic too (line 7).

Algorithm 1 Heuristic function for the main search algorithm

```
1: function HEURISTIC(n,n_goal)                    ▷ n and n_goal are nodes of G
2:     path ← SHORTESTPATH(G_A,n,n_goal)
3:     for each agent a of path do
4:         d ← min(d, DISTANCECOST(a) )
5:                                                 ▷ cost for a to go from n to n_goal
6:     for each handover HO in path do
7:         h ← h+ HANDOVERHEURISTIC(HO)
8:     return h + d
```

3.4.2.2 Handover tests

Even with the use of weighted and lazy variants of A*, the time consumed doing geometric computations is still high. When it is given a situation where no handover is possible, the handover search tool will fail after trying a number possibilities, which is highly time consuming. In order to speed up this process, we add simpler checks to discard impossible situations. The first one is based on distance, knowing the arm reach limits of each agent. The second is a collision test with the object alone: if there is no path for it from an agent to the other, the handover is not possible. The third is an inverse-kinematics test where we check that both agents can grasp the object simultaneously. The last one tests several pre-defined handover positions for collision, and if it succeeds applies a cost to it related to effort and comfort for humans when relevant. This cost is applied to the edge in the graph G to be used by the search algorithm. Optionally, the full motion can be computed, but the previous test is sufficient in most scenarios. Note that the handover search tool can be substituted with any state of the art algorithm as soon as it can find a solution from 2D poses of each agent and compute its cost.

3.4.3 Schedule

The path provided by the search algorithm lacks information about the agent states at every moment. The post-process phase enables us to add all navigation actions that do not involve the object, *i.e.* the navigation to go from initial position to a handover place to get the object, and the navigation from a handover place to the initial position after giving the object. They already have been computed during the search but need to be added to the solution.

The resulting solution is unordered, We use Simple Temporal Networks (STN) [Dechter 1991] to schedule all tasks.

The following assertions are applied to build the STN: *(i)* a handover can start only when both agents are at their handover position; *(ii)* after a handover, agents can leave only when the handover is fully finished.

Knowing the duration of each task, we can use the STN to compute all tasks optimal start and end dates for the plan to be executed faster.

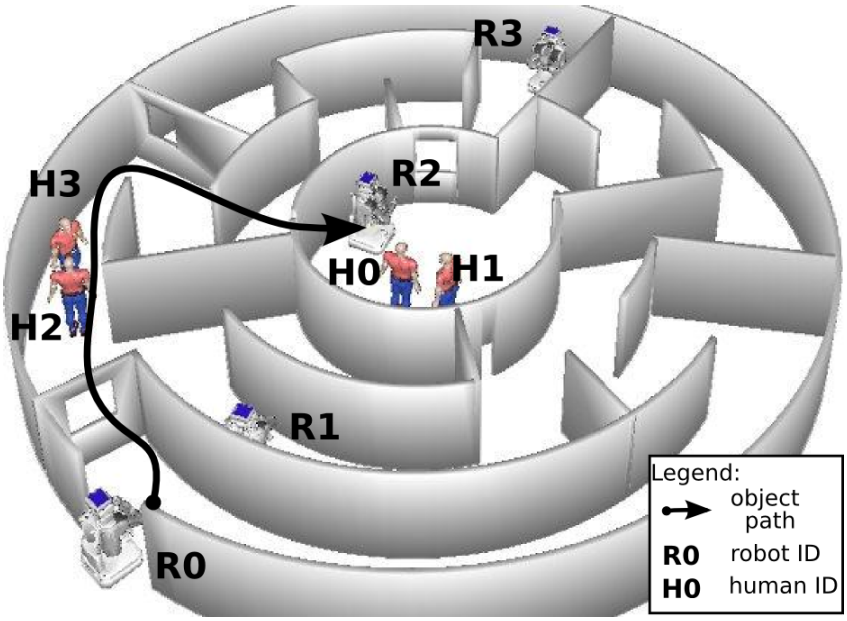


Figure 3.6: The maze example, with a possible path for the object shown by the black arrow. In this example, the object is successively held by R0, H2, H3, R3, H0. The use of multiple agents saves time for the delivery while asking more work to humans. Their use is still limited as most of the navigation is done by robots. Under other settings, the whole navigation could be done by the robot R0, which is slower but does not disturb humans at all. Or the same path for the object could be done while being held mainly by R1, but it would cause the humans to leave the place for the robot to go across.

3.4.4 Collision post-process

The final step of the planning process is to guarantee the feasibility of the trajectories according to collisions between agents. Following the hypothesis formulated in section 3.4, we did not check for agent-agent collisions during the search. As it can still occur, this step explicitly verifies if every agent trajectory is feasible, and in the case of collision, tries to find a solution. Methods for solving multi-robot navigation problems often require that all the agents are already navigating, meaning these existing approaches cannot solve a problem where an agent not involved in the task is blocking the path of another agent [Bennewitz 2001, Simeon 2002, Alami 1995, Khambhaita 2017].

To find feasible paths for the multi-agent navigation problem, we use the schedule generated by the planner which let us know the position of any agent at any time. When a collision between agents is found, paths are recomputed taking into account all the agents involved in that collision. In most cases, still follwing the hypothesis of an open space, the path of the navigating agent is modified locally to go around the colliding agent. In cases where this cannot be done, like in narrow corridors, we search for a *safe place* where the blocking agents can go to leave clear place for the navigating one to get through. For each agent colliding with the navigating agent, we find a position where it is not on the path; we then solve this a classical multi-robot navigation problem. This part of the algorithm is highly time-consuming, and is called only when we have found a solution and we are almost sure to keep it. This post-processing reduces the solution quality without any specified bound, but it only modifies the solution locally, as least as possible.

3.5 Results & Discussion

The system has been tested on various environments, including a real-life test with our two PR2 robots. We show that it computes in a satisfying time to allow on-line use in small building environments and with a number of agents around ten, and perform even better with an initialization phase. We will discuss the scalability, and adaptability to other kinds of scenarios and the integration in more complex systems, with other tasks, will also be examined.

3.5.1 Examples

We have tested our algorithm in environments of various complexities and likeliness [2].

Example 1 - rooms

The environment presented in Figure 3.1 is a realistic scenario with medium complexity ($18 \times 16 m^2$, 10 agents). The agents share common areas, where handover

2. The simple scenarios with only one handover are not presented in this paper, although, we tested them and found satisfying results.

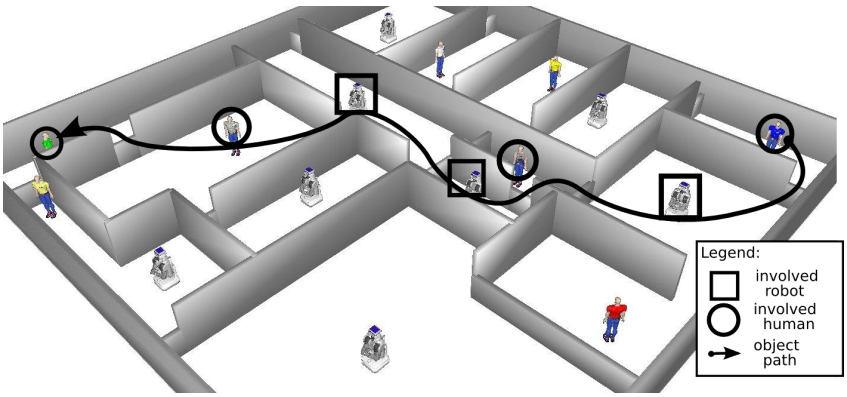


Figure 3.7: The big rooms example, with a solution represented by the black arrow. Seven agent out of sixteen are part of the plan. This plan tries to minimize delivery time. Other solutions avoiding humans would use only robots and pass by the room at the bottom of the image.

are possible everywhere. This property increases the complexity: each node in G has as many neighbours as the number of possible handovers with each agent sharing the same area, in addition to the navigation neighbours. The node number in this example[3] is about 64000 (we use a discretization step of 0.15 in all the examples, and for this one, there is an average of five possible agents per cell –each cell of the 2D grid is reachable by 5 agents in average). In this environment, the A* algorithm is efficient as the heuristic is close to the real cost.

Synthesis: $18 \times 16m^2$, 10 agents, 64000 nodes, efficient heuristic.

Example 2 - maze

Figure 3.6 is a maze where there is always a direct solution where the first and target agents can meet to perform a single handover. Windows allow faster delivery if the object is handed over through them between intermediary agents. The A* heuristic gets trapped in this environment as a solution is rarely close to the straight line. There are 102400 nodes (8 possible agents per cell).

Synthesis: $18 \times 16m^2$, 8 agents, 102400 nodes, inefficient heuristic.

Example 3 - big room

The environment in Figure 3.7 is a large environment ($25 \times 25m^2$) where the 16 agents are in rooms connected by doors or windows. The graph G is relatively small (41500 nodes) due to the small number of agents who share the same area (1.5 possible agents per cell in average) even if the size of the environment is bigger

3. In order to compute the node number we use the formula: $surface/(discr.step)^2 \times$ number of agents per cell.

Table 3.1: Computation times for each environment

environment		Computation time		solution	
	ε	mean	SD	cost[a]	SD
large rooms	10	20.5	32.0	1.10	0.27
	4	26.0	39.0	1.02	0.24
	1	61.3	62.9	1.00	–
maze	10	2.07	3.6	1.42	1.53
	4	2.11	3.8	1.37	1.68
	1	13.04	26.5	1.00	–
rooms	10	3.6	6.5	1.73	1.73
	4	2.5	3.1	1.46	1.46
	1	20.4	31.0	1.00	–
apartment	10	12.4	26.3	1.49	1.13
	4	16.0	48.7	1.18	0.31
	1	33.3	104	1.00	–

[a] relative to the optimal solution cost ($\varepsilon = 1$) found for the same problem instance

than in the other examples. The A* heuristic does not get trapped as in the maze, but solutions usually differ from straight lines.

Synthesis: $25 \times 25m^2$, 16 agents, 41500 nodes, normal efficiency for heuristic.

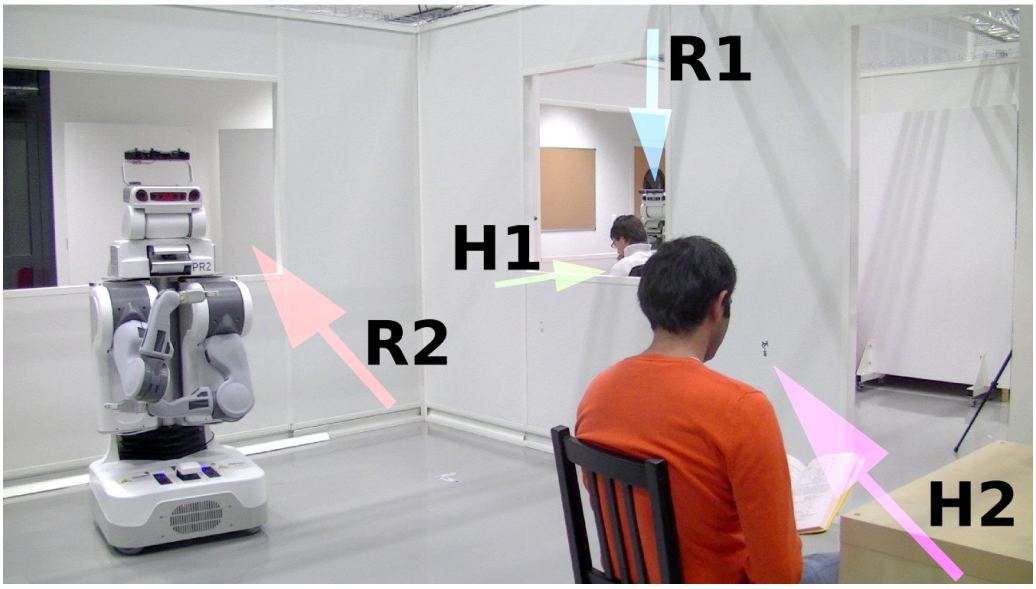
Example 4 - real robot

We tested our algorithm in a real environment (Figure 3.8), it is a small environment ($8*15m^2$) with few agents (2 humans and 2 PR2 robots). There are 16000 nodes, with 3 possible agents per cell. In our implementation, one robot computes the solution (without the trajectories) and share it with the other robot, then each one of them computes its own trajectories based on this solution.

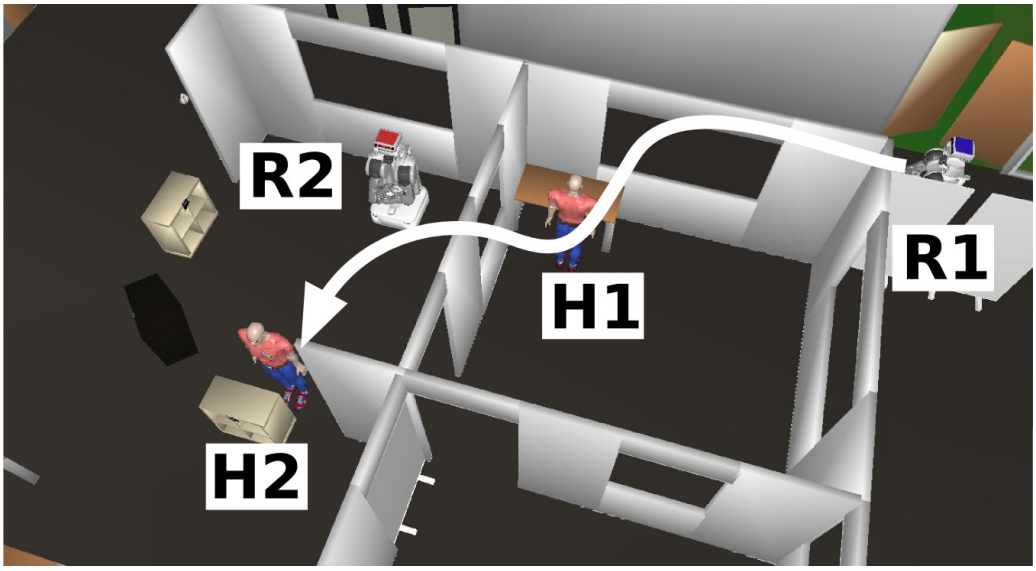
Synthesis: $8 \times 15m^2$, 4 agents, 16000 nodes, normal efficiency for heuristic.

3.5.2 Results

Table 3.1 gives mean computation times and their standard deviation for studied environments. Each mean time is computed with 40 samples with randomly selected start and goal agents. Half of the samples give priority to the execution time (deliver as fast as possible), and half prioritize human comfort (by avoiding including humans in the solution) The program is run on an Intel® Xeon® Processor E3-1271 v3 (8M Cache, 3.60 GHz), it uses one core only.

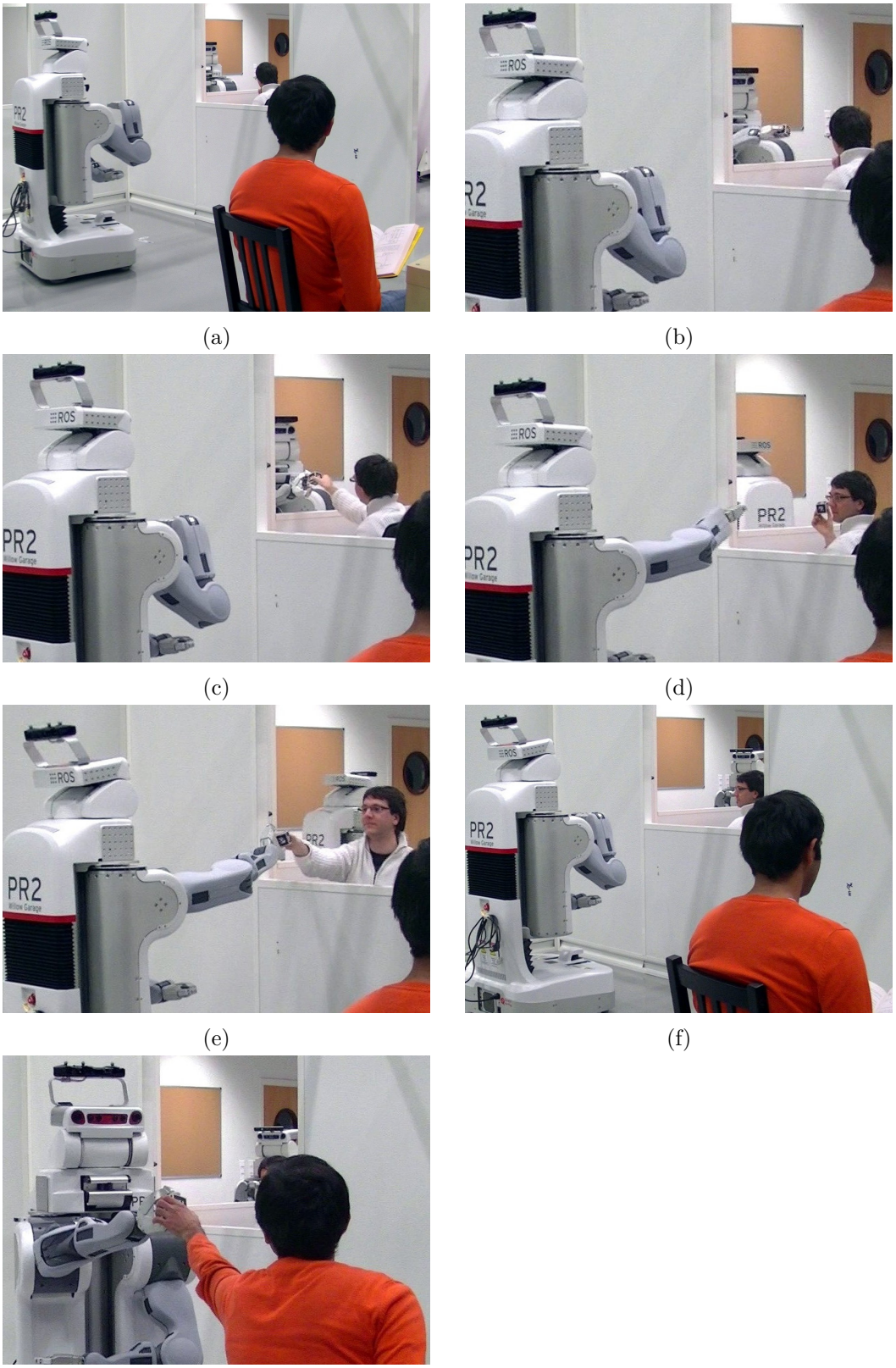


(a)



(b)

Figure 3.8: The real robot example. The bottom picture is the 3D model, with the object path represented as a white arrow. The solution involves successively R1, H1, R2, H2, and the object is handed over through the windows, minimizing H1 efforts, and brought directly to H2 so he doesn't have to move.



(a) (b) (c) (d) (e) (f) (g)

Figure 3.9: Snapshots from the implemented system demonstration video presented in Figure 3.8.

3.5.3 Discussion

3.5.3.1 Computation time

It is much faster to compute with higher epsilon value as many expansions are spared due to the depth first tendency of the Weighted A*. The loss on the cost is far under the upper limit of sub-optimality (ε) as our heuristic is really optimistic. For the lower epsilon, high variance is due to the diversity of the problem instances (different starts and targets agents), some of which are easier to compute than others. Note that the possible handovers are memorized making following request resolution faster than the first ones. The lazy variant spare a number of real expansions: averaged on 80 random runs in the rooms environment (Figure 3.1), the true cost is computed only for 0.46% of the explored nodes. The ratio between the computation time of a real cost evaluation and the temporary cost estimation is almost 200, with $67ms$ for the former and $0.334ms$ for the latter. Statistically, without the lazy variant, evaluating explored nodes would be more than 100 times as long. That factor reaches 400 in the rooms environments, while still providing exactly the same solutions. This reinforces the relevance of the Lazy Weighted A* variant use in our case.

It appears that environment influence the computation time strongly. Big environments or environments with a high number of agent per connected components of the navigation grid are more expansive to explore for the search algorithm. The "rooms" environment compared to the others is very straight forward: the heuristic is quite good, so even if there are lots of nodes in the graph G, only few of them will be expanded thanks to it.

The humans are explicitly taken into account in the cost computation. When the cost are oriented toward humans comfort, the algorithm will try to avoid including human in the sequence solution, but if the costs are oriented toward a fast delivery, the agent are considered equally in order to achieve the task.

3.5.3.2 Adaptability

This approach can be adapted to any kind of scenario, but will have limitations according to its characteristics. Mainly, the complexity of the problem explodes with the size of the environment and the number of agents. This can be limited if agents are limited to smaller parts of the environment as in Figure 3.7. Larger environments or more complex ones (compared to those studied here) would be successfully solved by our planner, but the computational time does not allow online use for such situations. Though, such complex cases are supposed to be rare and do not enter under the scope of our work. Indeed, for larger environments, with a high number of agents, incertitude would increase dramatically, and planning for a solution encompassing all motions of all these agents has little interest. For long range deliveries, it is certainly wiser to consider it as a Pick-up and Delivery Problem (PDP, see Section 3.2), as handovers are time-consuming and induce dependence on multiple agents and uncertainty on success and duration; a navigation-only solution

appears to be more interesting. Our method is developed for situation with a few agents – three up to ten – in environments where handovers between more than two agents can make the task faster or more comfortable for humans.

3.5.3.3 Improvements

Our approach can be improved on many points. The use of adapted Monte-Carlo methods could give similar or better results, it could make possible to have a more complete model, or more accurate. An interesting feature to add is the ability to plan object exchange without handover, using place and pick actions, to relax synchronization constraints of the handovers. We use STN to compute optimal schedules, with fixed durations for each task but some have high uncertainty, either concerning robots (risk of failure, difficulty to perform handover) or humans (they will not do exactly what was planned). Adding this information to the plan will enable the use of appropriate tools to execute plans with uncertainty [Morris 2000]. Other improvements concern the model used. Instead of a grid with static discretization step, we can imagine using quad-tree structure [Finkel 1974] to optimize the number of nodes. Work can be done at improving the human related cost computation methods, the heuristic can be made more accurate.

3.6 Conclusion

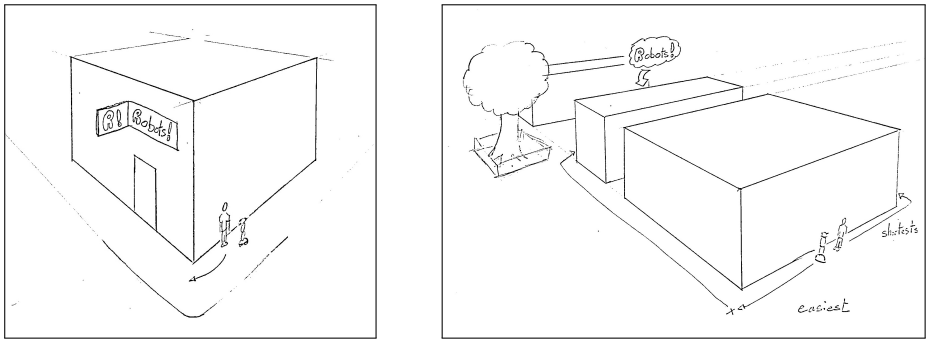
The task planner presented in this chapter aims at solving a complex task with both high dimensionality and an objective function unfriendly to optimizers. The main interest of the solution proposed is to use task level representation of the problem (the agent graph) as a heuristic for guiding the search in the physical space. The Multiple Agent Handover task involves collaboration between robots and humans, who have different goals and engagement in the task. The solver addresses it by considering all agents as “controllable”: it does plan for human motions, making the assumption that they are compliant, but also by trying to reduce the amount of effort they have to provide. We hence build interactions that are more acceptable to humans.

Chapter 4

Guiding and Providing Route Directions

Deciding where to go to provide landmark based directions

Contents



(a) The robot could either just say where the shop is, or guide the human around the corner for him to see it.

(b) There are two ways to get to the shop. The longest path can be better because a salient landmark (the tree) close to the shop is visible early on that path. The robot could guide the human to show the landmark, or just provide directions, based on that landmark.

Figure 4.1

4.1 Introduction

In the development of features for the service and entertainment robot for malls, in the scope of the MuMMER project, we have been working on the task of indicating directions to visitors. We started with a naive system that simply points at the requested place if it is visible, and say some short speech like "it's there". In the case the place was not visible from the human perspective, the robot would point in the direction of the location and say "it is in this direction". We rapidly found this behaviour unsatisfactory and started to make the robot move to place itself between the human and the pointed object, making easier for her to see both the robot arm and the pointed object. This looked like a task that needed some planning, and since the robot is moving, the human could move too, especially as she will move in any case, as she is asking for route directions. This made it a joint action, as both agents had to put effort in reaching a common goal: making the human partner reach his/her destination.

Before going in more details, let us have a first overview of what happens with real-life guides. If you go in a public place and ask your direction to an employee committed to providing information to visitors, said employee will most likely point a direction and give you some instructions to reach your destination ("This way, take the first street on your left,..."). In some trivial cases, they will point directly at your destination ("It is just here"). In some cases however, the employee may move around and take you to a position where they can show you some (previously hidden) landmark ("It is just behind this corner"), thus simplifying the directions and easing your task. And if you are lucky, they will guide you to your destination.

We wanted the robot to be able to reproduce these various ways a person can help others using speech and gestures while eventually moving around. The partners

of the MuMMER project were interested in such task, as also relating to the fields of dialogue, perception, and ergonomy in the more global Human-Robot Interaction domain. We are still continuing to develop and integrate software components within our team and within the MuMMER project.

The scenario could be summarized as follow for a robotic guide:

— an interactive robot, placed near an information desk in a public space, is available to provide information and route directions
— it can move a little (say several meters around its base) in order to place itself and ask its human addressee to move with it in order for both of them to reach a configuration where it can point to one (or several) landmark(s) and utter route direction information
— the robot is not intended to accompany the persons to their destination but to help way-finding.

Interestingly, the literature abounds in contributions related to specific aspects of the task, but we found none addressing the complete scenario we presented above. Existing systems distinguish between:

— robot guides that accompany or open the road to persons;
— systems providing only route directions;
— systems providing route directions using landmarks where it is assumed that the pertinent landmarks to point at are already visible from the human point of view.

Concerning the guiding robots, the propositions range from the first museum guides able to navigate in crowded environments like Rhino [Burgard 1998], Minerva [Thrun 1999], more recent robots with social and interactive capabilities like Rackham [Clodic 2006], Robox [Siegwart 2003], FROG [Evers 2014] and lately robots guiding in large areas like SPENCER [Triebel 2016], Obelix [Kümmerle 2013], ACE [Bauer 2009]. However, the focus of these robots was more to open the road or accompany a person until they reach the destination or through several attractions. In systems like SPENCER, the robot only accompanies the persons to their destination, while the museum robots guide their visitors through a sequence of predefined places from where some elements are pointed at. The later perform deictic gestures, shared attention, and dialogue, but they do not plan for the pertinent places to go to perform these actions, they are predefined.

We can observe the same things for robots providing route directions: there are many implementations and studies of such systems, but very little has been done when the robot and the human are placed in a way where they cannot see the landmark. Indeed most of the existing work assume that they are already placed in a favorable position and, if the human are not correctly placed, they assume that they will adjust. Additonally, the pertinent way a speaker and a listener share their space, move and point accordingly to enable the understanding and completion of way-finding direction has not been very much studied in the spatial cognition literature and has not been yet tackled as such in the human-robot interaction community.

The scenario we are addressing is similar to the one proposed in [Kanda 2010],

but with a major difference: in our case the robot is able to navigate with people to reach a perspective more pertinent to provide route direction. The scenario is in-line with what people expect from an information robot in a shopping mall according to [Satake 2015a].

As stated earlier, there is a shared goal, which is to make the visitor reach his desired destination. In this task, not only the robot action needs to be taken into account but also the one of the humans since they will create a mental model of the route, interpret the information, search for it in the environment, place themselves according to robot actions,... [Allen 2000]. That's why it is typically a human-robot joint task where the robot needs to have the abilities to estimate the perspective of the human, and to elaborate a shared plan involving the human and the robot that will allow to place both of them in a desired perspective.

We focus in this chapter on the decisional and planning component, called the Shared Visual Perspective Planner (SVP Planner), but this work is part of an overall project that aims to implement and evaluate the complete system. This involves the development of a number of components such as a human perception system [Duffner 2013], a human-aware reactive motion planner [Khambhaita 2017], a supervisor [Devin 2016] and a dialogue system [Papaioannou 2017b]. We will provide an overview of a more complete system in Chapter 5.

In the following, we first present related work in section 4.3. Then, we discuss our view of the problem statement and related definitions in 4.2. Information required about the task, environment and involved agents are presented in 4.4. In 4.5 we discuss the evaluation criteria for the task, and in 4.6 our implementation of a planner, with results in 4.7. We conclude and consider improvements in 4.8. Integration of this component in a real and working system is presented Chapter 5.

4.1.1 Definitions

We propose here some definitions we will use:

Visitor the person asking for help.

Guide the (robot) agent providing information.

Target the place the human visitors want to reach.

Landmarks indications, either verbal or physical, are based on landmarks, they include signposts, billboards, recognizable furniture or building elements [Nothegger 2004].

Guiding we will use this term in the meaning of physically going with the guided persons to some place (not necessarily the target).

Pointing the action of extending one arm in the direction of an object in order to indicate its position to another person; a successful pointing task results in the other persons seeing and unambiguously identifying the object.

Route Directions refers to the information one gives to the visitors in order to have them find their way toward a target. They can be a mix of ver-

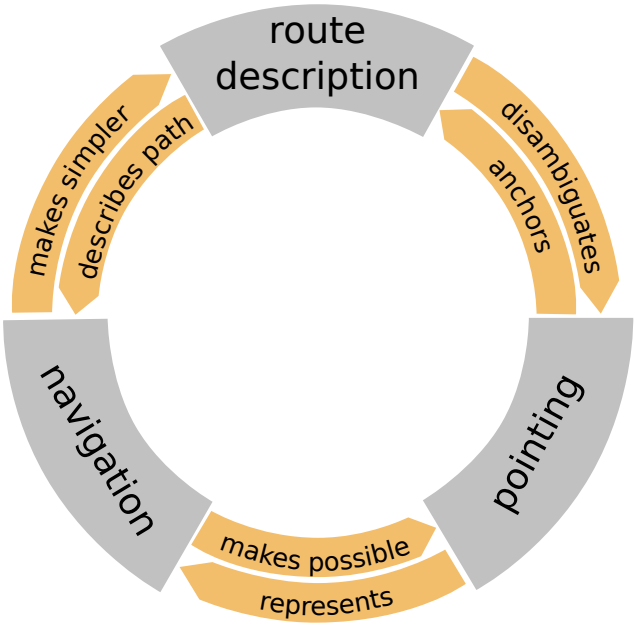


Figure 4.2: Schematic representation of the links between the three task modalities.

bal indications, pointing in the physical environment or on a map given to describe paths or locations.

Visibility the visibility of an object from a perspective is quantified by the size of the object in the field of view, i.e. a solid angle.

4.2 Problem Statement

The problem addressed here is related to several modalities: speech, gestures (including deictic) and navigation. These modalities are deeply connected in the sense that they support each other, as illustrated in Figure 4.2. The robot uses speech to describes a navigation path the visitor has to follow to reach his/her destination; gestures improve speech by anchoring it to landmarks or describing actions; the robot can navigate with the human to get closer to the target thus simplifying the description speech, and moving may also allow different (better) deictic gestures by reaching a more appropriate perspective.

All in all, there is a continuity of solutions between providing route directions from the starting point to guiding the visitor to his/her destination, including guiding a little to a good perspective to provide route directions. Capturing this whole task requires taking a decision on which solution to pick in this continuum. We present here a framework and a planner deciding *where* the route directions should be provided, and which landmark will be pointed at by the guide, regarding an evaluation of dedicated costs.

The Shared Visual Perspective Planner (SVP Planner) we developed and present in the following considers a task being a sequence of the following steps:

1. the robot guides –physically accompanies– the visitor to some place;
2. it points at an object;
3. it provides route directions based on the pointed object;
4. the human leaves with an improved knowledge on the route they have to take to reach their destination;

in that order, but with each step being optional. The SVP Planner chooses the extent of each of these steps by choosing a place where the directions will be provided. It searches for a position for both the visitor and the guide where the latter can point some landmark(s), and from where the visitor can reach his/her destination following a path with a simple description provided by the robot. It is important to notice that all of these four steps are taken into account by the SVP planner to evaluate solutions of the task as a whole. Doing so requires appropriate information on the task, the partner and the environment.

4.3 Related work

To understand the pertinent aspects of the task we aim at solving, we based our work on existing research in the fields of human cognition and communication and on similar robotic systems. The purpose here is mainly to understand what type of information and computation is needed for our planner to produce a solution that can then be used by a system performing the action. This chapter does not present how the action is executed, however the planner has to know as much as needed about the action that will be performed. For that purpose, we review here contributions related to the tasks of guiding, providing route directions and pointing. Both the human cognition studies and robotic or system implementations are presented.

4.3.1 Using and Selecting Landmarks for Route Directions

Using landmarks while providing route directions is common practice and natural for the humans. Landmarks are used in the route indication to describe places, particularly those where an action must be taken or those the person should direct to or avoid. When the landmarks are visible, they can be pointed at directly, for the visitor to identify them unambiguously. The pertinent way to use landmarks in route direction is discussed in [Daniel 2003], and they show that propositions connecting landmarks and actions to take (like “at the parking lot, turn right”) are particularly important. Criteria to select the landmarks are presented in [Nothegger 2004], where they propose a system to automatically choose them taking into consideration the pertinence with respect to the route to describe and the salience. In [Clark 1983] they study how humans perform and interpret demonstrative references (including gaze and pointing), and they show that common ground, shared knowledge and speech is used to disambiguate the object that are referred to. [Allen 2000] proposes “best practices” for the choice of route direction based

first on a temporospatial ordering of the statements and then on the use of shared knowledge to convey common ground during the interaction.

4.3.2 Pointing

In situated dialog, physical signals intended to direct the addressee attention to an element of the environment can be sorted into two classes: "directing-to" where an agent perform an action intended to direct the attention of the partner, and "placing-for" where an object is placed to gain attention of the partner [Clark 2005]. In our task, the robot has to direct attention to landmark, mostly using pointing gestures, but cannot place them for the visitor to see them; however it can *place* the visitor to a location where she can see the landmark, which is probably another kind of "placing-for" action. In [Sauppé 2014] a study is conducted to highlight the rich design space for deictic gestures and the necessity to adapt them to physical, environmental, and task contexts. Another key aspect for the synthesis of the robot pointing gestures is their legibility [Holladay 2014]. Interesting studies have been done in the analysis of gestures accompanying verbal route directions, [Allen 2003] shows that deictic gestures are very frequent and that they are generally related to an immediate speech element.

[Hato 2010] studies the cognitive models used by humans to point at regions in space and to interpret such gestures, and propose an implementation of such model.

The synthesis of a working system using a combination of speech and gesture in order to achieve deictic reference on objects of the environment with humans has been discussed in [Brooks 2006].

4.3.3 Placements to Share Visual Perspective

[Wong 2010] provides a pertinent analysis of the stages to a successful pointing gesture. They mention the need for the viewer to be able to see both the gesture and the referent as well as the necessity of holding the gesture until coming to mutual agreement with the observer about what is being pointed at.

In [Kelly 2004] the importance and role of the "Shared visual space" is stressed, along with the difficulties met by humans to estimate the perspective of others.

Concerning issues linked to placement planning, there are substantial results on planning sensor placement (e.g. [Chen 2004]) as well as planning a position for a robot to perform shared attention and perspective with a human [Marin-Urias 2009], but we have found no contributions on planning shared perspective for both the human *and* the robot.

Also, we have not found any reference regarding the way a speaker and a listener place themselves and possibly move during the explanation of the route. This is the reason why a first field study was conducted with partners of the MuMMER project, concerning human-human route guiding in the context of a large Mall in Finland [Belhassein 2017].

4.3.4 Route Directions

Route directions are indications provided by the guide to the visitor for him to reach the desired destination. It is done mostly thanks to dialogue, and is often improved by gestures, as we have seen above [Daniel 2003, Allen 2000, Allen 2003]. Route directions are successfully communicated to the visitors when understood them and can navigate to their destination.

Some studies show that humans uses gestures to represent what the visitor will see or do in the future, and that such gestures facilitate the communication of spatial information. The speaker can represent spatial information either from *route perspective* or *survey perspective*. That is by projecting themselves in a mental tour, or taking a bird's-eye view. [Alibali 2005] for example discusses the purposes of gestures in spatial cognition and communication. Gestures are helping speakers to think about space and decompose their speech in temporal units, they are also effectively improving the communication of spatial information. They also outline variability in gesture productions dependent on individuals and task.

Systems able to provide route directions have been widely studied by T. Kanda and his co-authors. In [Okuno 2009] they propose a model for a robot that generates route directions by integrating three crucial elements: utterances, gestures, and timing. In [Kanda 2010] they present a mall robot with multiple functionalities, including the provision of route directions; however in that version the route direction by themselves are simple and the system takes no decisions on how to indicate a route. In [Matsumoto 2012] they present a robot companion which can indicate locations of shops using previously visited places, but they do not consider providing route directions when the desired shop is not near a shop already known by the user. They present an architecture and field study including user feedback in [Satake 2015a]. The generation of route directions is based on their previous work [Morales Saiki 2011] where they model the environment from a route perspective, using maps and also acquiring data by physically visiting the environment. They are able to generate route directions using speech and gestures using that model, referring to landmarks that where detected while visiting the environment. In these later systems, they are still not planning for the placement of the agents nor choosing the best landmarks to point at. In [Satake 2015b] they summarize which notions are important for an information providing robot: building a knowledge about spatial relationships among shops and corridors, gaze, gestures and engagement monitoring.

Other authors also studied this task from a robotic perspective, [Bohus 2014] present a *Directions Robot* able to provide route directions using speech and pointing gestures. But neither do they plan for a perspective to go to provide information. [Chen 2017] present a robot specific to malls, *KeJia* which is a shopping assistant, also limited to accompanying persons from point A to B, but not providing route directions nor planning for perspectives to reach.

4.4 Model

Our approach relies on a variety of information about the environment and the agents, either symbolic, physical, or on mental states. Provided with these informations, the SVP planner is adaptable to any situation where a robot can provide route directions and point at landmarks, like city streets, museums, malls, amusement parks, offices, university campus...

4.4.1 Physical Environment Model

The environment has to be represented in three dimensions, its accuracy influences the pertinence of the visibility computations and navigation planning. All the obstacles to navigation or sight must be represented. The model must discriminate potential landmarks from each other and from other objects or obstacles to allow the computation of a specific landmark visibility. In our implementation, we represent the environment using 3D meshes, visibility of objects is computed with OpenGL (similar to what is used by [Marin-Urias 2009]). Each 3D object is associated with a unique identifier, and can actually be used as a landmark to be pointed at; the definition of which are pertinent landmarks to use for some route lays in the symbolic model.

4.4.2 Symbolic Environment Model

The decision algorithm needs information on a symbolic level, mostly about landmarks, to determine which are pertinent for the task. To choose landmarks we need to know how they relate to the route we want to indicate and how they influence the utterance of route directions. In our implementation this information is not computed, but provided as input. A planning request contains a list of landmarks that can be used for the task, each associated with a scalar representing the duration of the utterance of the route directions if this landmark is visible and used. It appears that this duration also represent the difficulty for the human to remember the indications. The SVP planner takes these values as input, however [Morales 2015] presents an environment model built for providing route directions that can be suitable to our approach. We will present in Chapter 5 a similar component we developed to work with the SVP planner.

4.4.3 Human Model (visitor)

We want the guide to adapt to different human visitor capabilities, so the system is accessible and do not discriminate certain persons by ignoring their specificities, and also adapt to a range of use cases. To adapt the solutions, our system can make use of information linked to:

- vision:
  - height of the subject eyes, to compute its perspective accordingly;
  - visual acuity to enforce the use of more visible and salient landmarks;

— navigation capacities: navigation speed, to compute plan duration and give more important penalties to long routes;
— urgency, to balance the importance of plan duration over the other criteria.

This information is taken as input, we believe they can be acquired or inferred through dialogue and perception (e.g. persons with a stroller or loaded shopping cart, persons in a wheelchair or with crutches are usually slower than average; a person in a hurry may express it verbally or through body attitude). In the integrated system presented in the next chapter, only the subject eyes height is actually computed from perception.

4.4.4 Robot Model (guide)

The robotic guide may be able to navigate, in which case the planner can take as input a maximal distance the robot can run from its initial position. We use a speed estimation to compute plan duration (that is probably not the maximal speed, but rather an average measured speed in the same situation – environment, crowd...). Our approach can indirectly take into consideration capacities of the robot by tuning some related parameters: accuracy of the pointing gesture and gaze estimation, dialogue capacities (inducing a higher cost of dialogue based tasks).

4.4.5 Domain Parameters

Some parameters are defined domain dependently. The guide may be allotted a limited amount of time to serve each visitor, or on the contrary be expected to help each engaged visitor as much as possible, *i.e.* provide the maximum effort to solve a request once it has been asked to the guide.

4.5 Evaluating the Solutions

The decision is based on estimation and comparison of the possible solutions to the task. The solution evaluation has to take into account:

chances of success : the simpler the indications, the higher is the probability that the human will remember them and reach the destination;

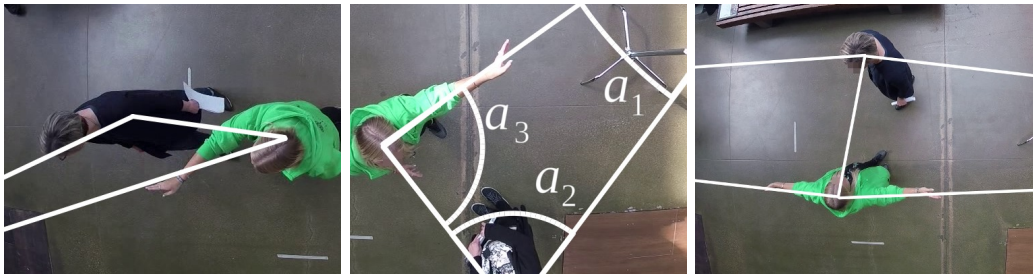
optimality for the visitors in terms of duration and effort;

optimality for the guide according to its global objectives – serving as much visitors as possible or providing the best quality of service for the individuals.

We will present how the SVP planner evaluates a solution regarding each step of the task.

4.5.1 Placement for the Pointing

When pointing at an object, the guide objective is that the visitor identifies it unambiguously. To achieve this, it may be helpful to (1) reduce the difference of perspective, by getting the two agents almost aligned with the object. Stress is put



(a) (b) (c)

Figure 4.3: Three examples of real pointing scenarios. The person wearing a green (light) sweatshirt is the guide pointing at a landmark (two in (c)); in white are represented the triangle formed by the landmark, visitor and guide. (Image credits: authors of [Belhassein 2017])

on the alignment when the object is difficult to distinguish because it is small in the field of view. A secondary objective for the guide is to (2) relieve the visitor from some physical or mental effort by placing itself between the visitors and the destination, so they don't have to turn their head around to successively look at the pointing arm or gaze and in the pointed direction. A last objective on the pointing position is (3) for the guide to be able to monitor the visitor gaze, and speak to them; but the guide also needs to enforce pointing with gaze, so it should be able to look at the visitor and the at the chosen landmark.

These properties are estimated by building a triangle whose vertices are the visitor, guide and pointed object centers, as represented in Figure 4.3. The three angles (see Figure 4.3b) denote the above mentioned properties of the pointing position. The angle (a_1) at the pointed object vertex correspond to the difference between the perspectives of the agents. The angle (a_2) at the visitor vertex reflects how much they have to move to switch from looking at the guide being pointing and the pointed object; it also indicates if the guide sees the visitors face when they look at the target, allowing gaze detection or not. The third angle (a_3) is for the guide to look at the object and the visitor. Naturally, we, as designers, tend to put more emphasis on reducing the two first angles, as doing so should reduce visitor efforts and improve chances of success.

4.5.2 Guiding

The joint navigation step is evaluated considering the distance run while guiding, and the duration of the guiding step thanks to the speed estimations provided as input. When a guiding step is necessary, the solution evaluation is penalised by a constant value that represents the time needed to ask the visitors to move and explain what they should do.

4.5.3 Route Directions

The utterance of route directions to the visitor, or more ambitiously the construction of a dialogue in which the directions are given to the visitor, is likely to be a time consuming step of the task. Even more importantly, it is a critical part for the success of the task: too complex instructions will be likely to lead the visitor to get lost or simply abandon the task and find another way of reaching their objective, making the guide counterproductive. The duration and complexity of the route directions is directly related to the number of steps of the route [Daniel 2003]. The guide will need to find simpler routes, use visible landmarks to simplify them, move to a place where such landmark is visible. This is illustrated in the example of Figure 4.10b where the guide uses a landmark next to the door and starts its route directions from that point, hence removing one step in the route to explain (the one to reach the door from the current position).

4.6 Implementation

The SVP Planner decides where the robot and human should go to reach a good shared perspective from which efficient route directions can be given. This step is a key decision of the overall task, as the position will determine all the other steps. This is why in the algorithm presented below, we use an objective function that encompasses all the task, rather than just evaluate the quality of the pointing and perspective. Our solution is designed to match the high coupling of all the steps of the task.

From the presentation of the model and evaluation in the previous sections, a request to the planner must contain the following data (details will be given later):

— p_{g_0} and p_{v_0}: initial position of the robot (guide)and the human (visitor);
— p_{v_d}: visitor's destination position;
— L: list of landmarks we can point at with the associated indication durations;
— height of the eyes of the human;
— height of the "eyes" of the robot: those that the human consider to be the eyes of the humanoid robot, not the camera actually used for perception;
— s_v and s_g: speed estimations for each agent;
— D_{max}: maximal distance the robot can run from its initial position;
— V_{min_v}: minimal visibility score to consider an object visible by the visitor;
— K_x: other parameter to tune the importance of each aspect of the task with respect to each other in the choice of the best solution (like optimizing the duration over the visibility,...)

The planner produces as output:

— placements for both the guide and the visitor $X = (p_g, p_v)$;
— list of landmarks from L that are visible from those placements.

4.6.1 Search Space

The planner decomposes the area accessible to the guide in a two-dimensional grid, and searches for the best pair of positions

$$X = (p_g, p_v) = ((x_s, y_s), (x_v, y_v))$$

for the guide and visitor, expanding from the initial positions $X_0 = (p_{g_0}, p_{v_0})$.

The SVP planner also searches for the best landmark $l \in L$ to use. A solution is then written as a tuple made of one landmark and the position for each agent: (l, p_g, p_v).

The destination state is $X_d = (p_{g_0}, p_{v_d})$ (the guide goes back to its initial position and the visitor reaches the destination; if the destination is outside the grid, p_{v_d} is the position of the visitor where they will leave the grid).

4.6.2 Constraints

The constraints are computed for a tuple (l, X). A solution is valid only if all the constraints are respected.

Visibility constraint ensures that the two agents see the landmark, hence that shared perspective and joint attention are possible

$$v(l, p_g) \geq V_{min_g} \text{ and } v(l, p_v) \geq V_{min_v}$$

Interaction distance constraint We want to keep the interaction distance within 20% of the desired distance

$$(||p_v\vec{p}_g|| - D_I)^2 < (D_I \cdot 0.2)^2$$

where $||p_v\vec{p}_g||$ is the distance between the agents and D_I is the desired distance interaction. The value of D_I relates to the proxemic theory, and should ensure a social interaction distance, typically around 1 to 1.5 meters. (The 20% error is actually a parameter linked to grid pace.)

guide range constraint keeps the guide within a certain navigation distance from its initial position (see Section 4.6.3.1 for details on navigation)

$$\lambda_g(X) < D_{max}$$

guide time constraint limit the duration of the task for the guide

$$T_{Guide}(X) + T_{Return}(X) + T_{Indic}(l, X) < T_{max}$$

4.6.3 Costs

In addition to these constraints, our implementation takes the following parameters into account to choose the best solution possible among the valid ones.

4.6.3.1 Navigation

To estimate the distances and duration of the navigation phases, we use a two dimensional grid with the same pace as the visibility grid. It allows us to compute shortest paths with Dijkstra Algorithm [Dijkstra 1971]. We compute the distances from three points, giving distances between these points and any point in the grid. We compute distances from p_{g_0}, p_{v_0} and p_{v_t}, respectively providing the following path lengths for any X in the grid:

— distance run by the guide $\lambda_g(X) = \lambda(p_{g_0}, p_g)$;
— distance run by the visitor $\lambda_v(X) = \lambda(p_{v_0}, p_v)$;
— remaining distance to reach the target for the human $\lambda_{Target}(X) = \lambda(p_v(X), p_{v_d}(X))$.

We compute an estimation of the joint navigation (guiding) step duration as

$$T_{Guide}(X) = max(\lambda_g(X)/s_g, \lambda_v(X)/s_v)$$

where s_g and s_v are the respective average speed estimations of the agents. We also have the durations

$$T_{Destination}(X) = \lambda_{Destination}(X)/s_v$$

for the human to reach the target and

$$T_{Return}(X) = \lambda_g(X)/s_g$$

for the robot to return to its base.

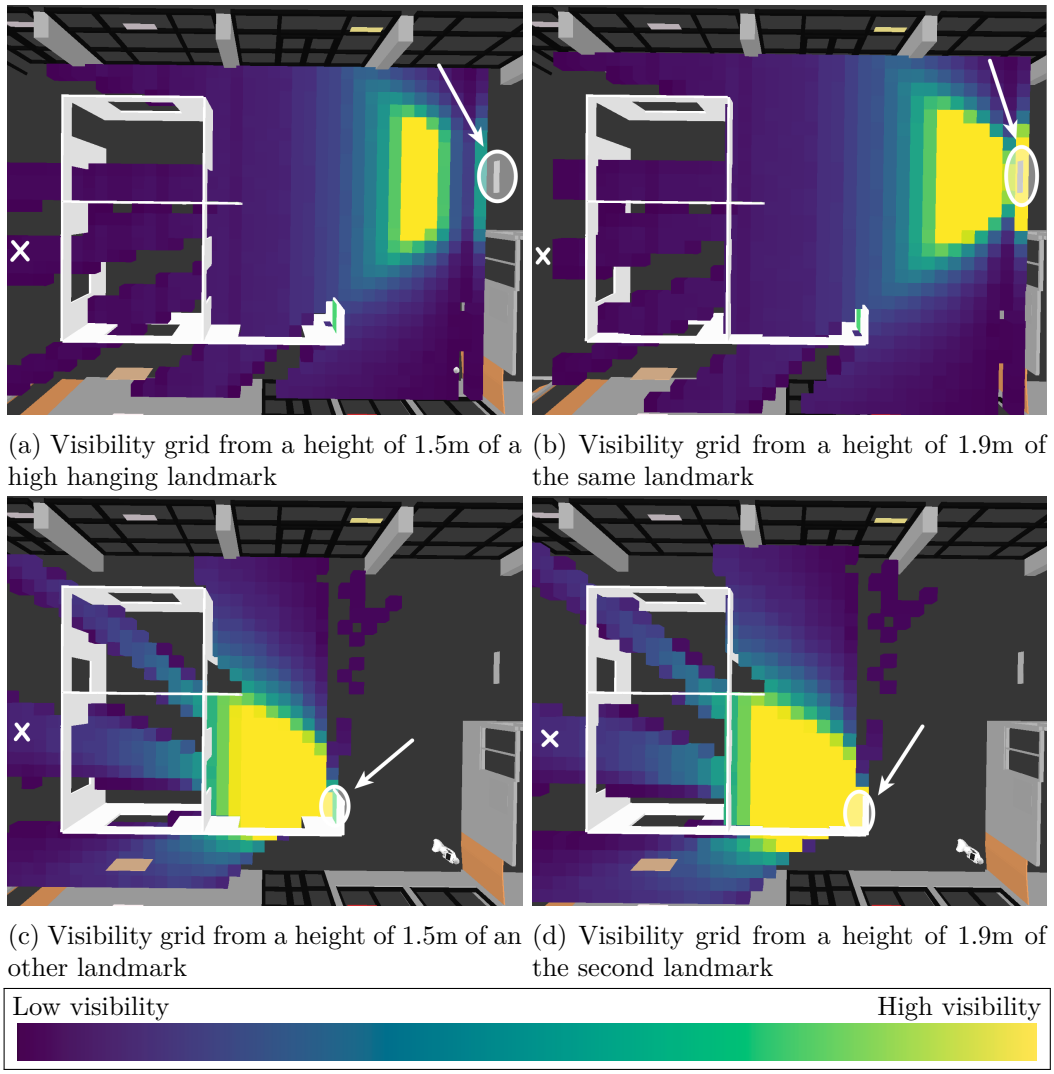
4.6.3.2 Landmark Visibility

For each landmark l and position X we compute the visibilities of l by the guide and the visitor:

$$v(l, p_s), v(l, p_v)$$

To speed up the computation, the visibility score is precomputed, because it is an expansive step. The 3D space is sampled with a 3D grid that holds a score representing the perceived size of the objects in any direction from each cell center. The grid pace in x and y is the same as the navigation grid, and the visibility scores are computed from several viewpoint height, allowing to better estimate the perspective of persons of different size (the values of $v(l, p)$ for various size of humans). Sample visibility grids are shown in Figure 4.4.

The visibility computation itself is done by assigning each object a unique color, rendering the 3D scene with OpenGL and counting the number of pixel of each color, from each cell of the grid. To avoid issues related to distortion, the field of



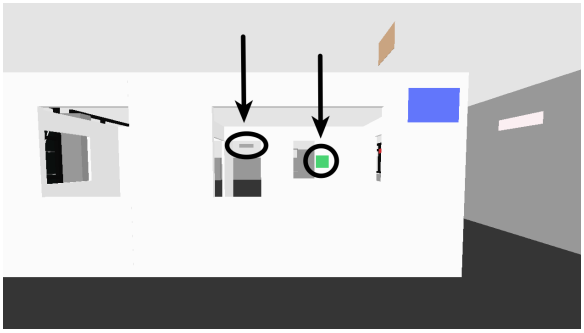
(a) Visibility grid from a height of 1.5m of a high hanging landmark

(b) Visibility grid from a height of 1.9m of the same landmark

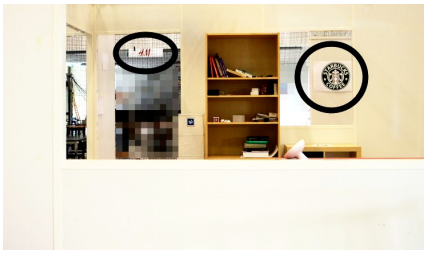
(c) Visibility grid from a height of 1.5m of an other landmark

(d) Visibility grid from a height of 1.9m of the second landmark

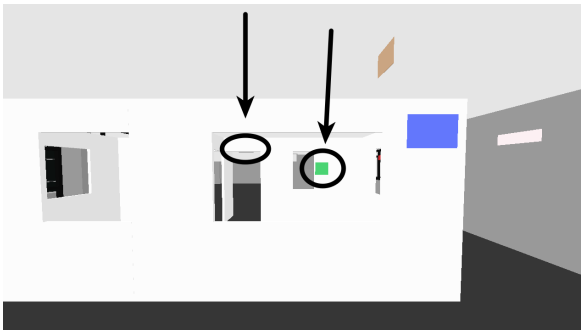
Figure 4.4: Grids representing the visibility of two landmarks in the G. Giralt building environment, at two different viewpoints height. The grids are actually three dimensional, we represent here two 2D slices of two visibility grids. Cells are cubes of 40cm sides. Lighter/yellow cells are those from where the visibility of the object is the best, while from dark blue ones the object is hardly visible. Transparent cells correspond to the object being not visible enough to be pointed at. We see how the visibility measure is determined by distance and obstacles. The landmark for which the visibility is shown is highlighted by a white ellipse and arrow, the cross on the far left is the position from where the perspectives of Figure 4.5 are taken.



(a) Perspective from a height of 1.5m, the two highlighted landmarks are visible



(b) Picture of the same perspective in the real environment



(c) Perspective from a height of 1.9m, the hanging landmark is hidden by the part of the wall above the door

(d) ...in the real environment

Figure 4.5: Two perspective taken from the position marked in Figure 4.4, with the same landmarks indicated by a black ellipse and arrow.

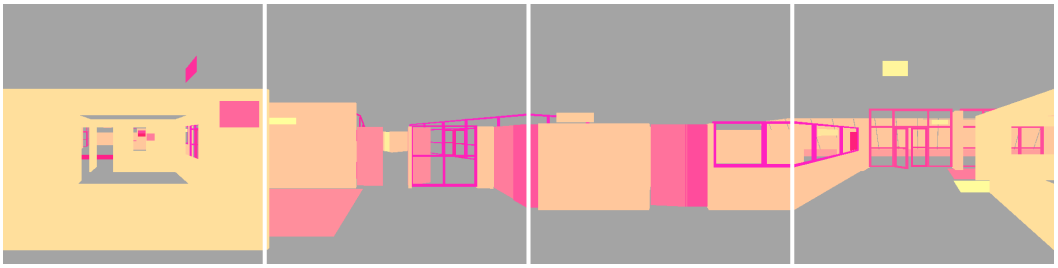


Figure 4.6: A view of how visibility of objects are computed. Each of the four images represent are renderings with a $90° \times 90°$ virtual camera, pointing in four orthogonal directions, thus creating a $360° \times 90°$ field of view. Each object of the scene are given a unique color, and pixels of each color are counted to know the size of the object in the field of view. Above a certain number of pixels, the object is considered visible from the point where the camera are placed for the rendering. The place from which these images are taken roughly correspond to the location indicated in figure 4.4, and the left-most image correspond to the perspectives shown in figure 4.5.

view is split in six section of a range of 90 degrees in each direction. The figure 4.6 shows the images rendered for computing the visibility of objects from a single point, with an field of view of $360° \times 90°$.

4.6.3.3 Route Directions

For each landmark l, we know the route direction based on that landmark has a duration estimation $T_{Indic}(l, X)$. This value is given as an input of the request.

4.6.3.4 Pointing Conformation

We use the three angles $a_i(l, X), i = [1, 2, 3]$ representing the pointing conformation, computed from a triangle whose vertices are robot and human eyes and the center of the landmark (as shown in Figure 4.3b).

4.6.3.5 Cost Function

The cost function combining those presented above is:

$$\begin{aligned} cost_l(l, X) = \Big(& (T_{Guide}(X) + T_{Indic}(l, X))(K_v + K_g) \\ & + T_{Destination}(X).K_v + T_{Return}.K_g + K_v.V(l, X) + 1 \Big) \\ & \times \Big(\sum_{i=1}^{3} a_i(l, X) K_{ai} \Big) \end{aligned} \quad (4.1)$$

Where $V(l, X) = max(0, V_{min} - v(l, X))$ and the K_x are inputs of the algorithm reflecting the properties presented in 4.4.

This is the cost for a landmark and position. As we want to choose the best landmark to point at, the cost $c(X)$ at a position X is the best of the $c(l, X)$, that is

$$Cost(X) = \min_{l \in L}(cost_l(l, X))$$

where L is the set of landmarks provided in the request.

4.6.4 Search Algorithm

Our implementation does a search by propagation from the cell containing X_0. The propagation is based on a set of *open* cells, where neighbors of previously closed cell are added, except when the closed cell break some evaluation constraint. This prevents the algorithm to explore all the possibilities.

The Table 4.1 exhibit planning times in the environments presented in 4.7. We can clearly see that our approach is not well suited for robot able to navigate large distances or cluttered environments needing a precise grid sampling. More appropriate approaches should be tested, like simulated annealing or hill climbing from several randomized points.

We can also consider that we are not actually looking for the global optimum, rather a *good* solution. Experience shows that cost function often inaccurately fathom the task, hence searching for the exact optimum of the objective function is excessive; when they are functions expansive to evaluate or with many local optima, it becomes unreasonable.

4.6.5 Considering Multiple Landmarks

The purpose of this work was first to choose the best place and the best landmark to point at, in a specified environment and among a list of landmarks. The choice is made by taking into consideration the whole guiding task. When multiple landmarks are provided, the implementation presented above searches for a place to point at the best landmark possible, that is choose the tuple (l, X) with the lower cost $c(l, X)$. Nevertheless, testing showed us that it would be useful to choose the landmarks to point at: the planner would be provided a list of landmarks that can help to describe the route, and it would try to show as much as possible. This behavior is achieved by using a constraint that enforces that at least one of these landmarks is visible (not all), and where the average of the landmark costs is taken into account in the solution cost.

Another improvement we found useful is to be able to consider optional and mandatory landmarks. In this case we consider the worst mandatory landmark in the constraint (to ensure that they are all visible), the average cost of all the visible landmarks in the solution cost, and the number of non-visible optional landmarks, also in the cost, to minimize the number of non-visible landmarks and maximize the overall visibility of the visible ones.

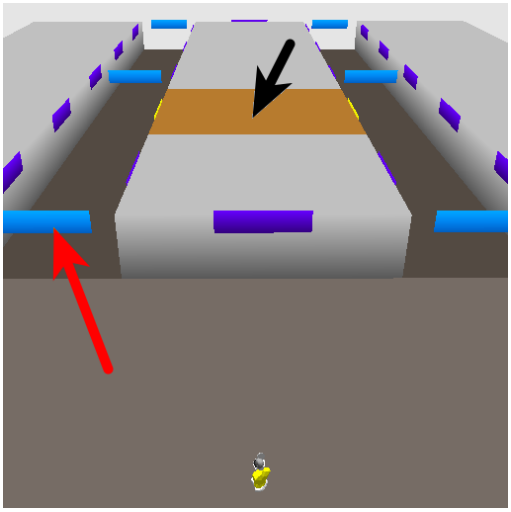
4.6.6 Choose the Best Route

One step further, the planner could be provided multiple alternative routes, and choose the best one based on the same cost function. Indeed, we try to capture the whole task in this cost. So this would be achieved by simply running the planner for each route, and pick the one which provides the solution with the best cost.

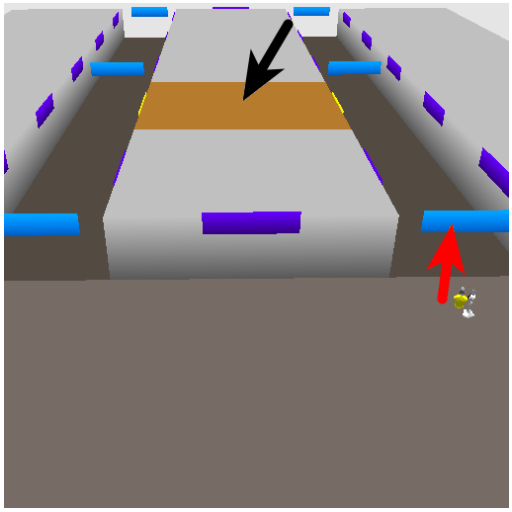
4.7 Examples

We present examples in two environments. The first one is a virtual mall (Fig. 4.7) with a central hall where the robot can navigate, and two corridors leading to numbers of shops. The shops in the central row are accessible from the two corridors. The available landmarks are either signboards indicating the shops sitting in their corridor, and shop fronts.

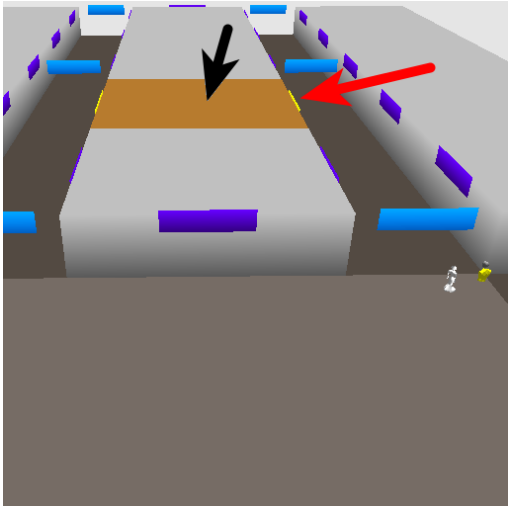
In the examples of Figure 4.7, we forbid the guide to move more than 1.8 meters away from its initial position. We see how the robotic guide is showing different landmarks depending on its position. The three solutions are obtained with the same settings and only a different initial state. Figure 4.7b and 4.7c illustrate how



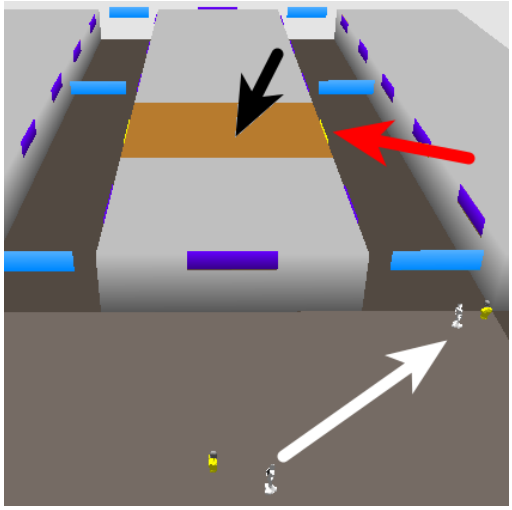
(a) The most visible landmark from this initial position is the signboard of the left corridor.



(b) Agents are facing the right corridor, but the shop sign is not visible enough from that perspective.



(c) Changing the initial position of agents, the shop sign becomes visible enough to be pointed at directly.



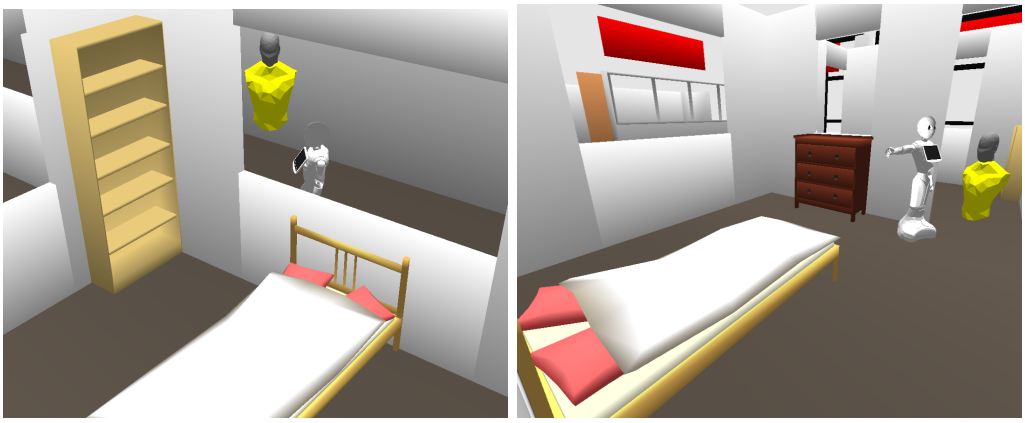
(d) With the same initial position as (a), the guide is allowed to guide the human along the white arrow to a perspective from where the shop is visible, hence deeply simplifying route directions to provide.

Figure 4.7: The first three images show solutions where guiding is forbidden – the robot is allowed to move 1.8m around initial position, which is not enough to significantly change the perspective. Red arrow indicates pointed landmark, black arrow (center) is the target location (second shop of the central row, accessible via the two corridors, highlighted in orange); on the contrary, in (d) the guide is incited to guide (white arrow) to simplify the route directions, the initial positions are the same as in a, but the guides takes the human to see the shop-sign.

Table 4.1: Planning times in the lab environment (pace= $0.3m$) (Fig. 4.8) and in the mall (pace= $0.8m$) (Fig. 4.7). The radius is the distance the robot must not exceed from its initial position. Times are averaged over 5 run, variance is minor due the deterministic nature of the algorithm. State numbers are the counts of states (X) explored by the propagation algorithm.

Grid pace (m)	Area radius (m)	State number	average time (s)
0.3	10	143,467	51.04
0.3	5	42,811	11.96
0.3	2	8,442	2.14
0.8	20	12,775	3.39
0.8	10	8,029	2.33
0.8	5	1,615	0.47

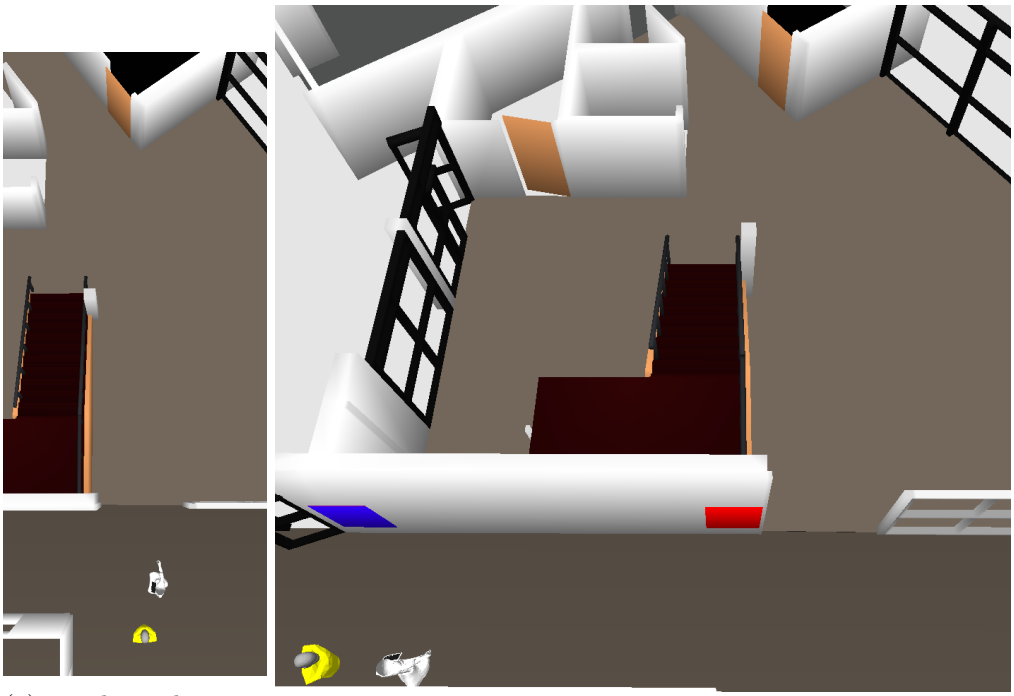
Figure 4.8: Overview of the Georges Giralt building 3D model, with virtual signs added to serve as landmarks.



(a) In this solution, the robot shows the bed to the human through a "window", limiting the joint navigation length.

(b) With the same initial situation but with a small person (child) who cannot see the bed through the window, the robot guides to get in the bed room.

Figure 4.9: Two distinct solutions to the same problem caused only by a different size of the visitor.



(a) The robot indicates the restroom door (top of the picture) to the human

(b) The robot points at a sign to indicate where the human should go to approach and see the restroom door (both come from the left side of the picture). Speech example: "When you cross the door next to the red sign there, you'll see the restroom door on the other side of the hall."

Figure 4.10

a small change in the perspective can lead to significantly simpler route directions: here from "the second shop on the left side of this corridor" to directly pointing at the desired target. When allowing the robot to guide the visitor on greater distances while inciting to find solutions with simple route directions, the robot guides the visitor to a location where the shop is visible (Fig. 4.7d).

The second example (Fig. 4.8) is the ground floor of the Georges Giralt building at LAAS-CNRS, featuring an entry hall, offices and meeting room, and a large hall with an experimenting apartment. The main hall is around 12 by 20 meters the apartment occupies a 9x9 meters square.

In Figure 4.10 we show how the guide can make use of landmarks situated on the path to the target and balance between guiding and providing route directions. In Figure 4.10a the robot is guiding the visitor to a place where the target is visible, leading to simple route directions; whereas in Figure 4.10b, we set a low speed to the robot, so the SVP planner prefers not to guide the human, and use a landmark to indicate a waypoint for human navigation (while the robot is still able to guide the human, the planner prefers no to).

In this same environment, Figure 4.9 illustrates the ability to take into account different human morphologies (sizes) and adapt to their perspective when pointing at an object that can be hidden by obstacles, leading to very different solutions, in this case with a small child unable to look over a window edge.

4.8 Conclusion

In this chapter, we have presented an original decisional framework for a robot to provide route directions to users of a public space, discussed the task model and its evaluation criteria and finally presented example solutions from an implemented planner.

This framework is the basis for a larger project, presenting a possible component for a complete robotic assistant able to guide human visitors using both route directions and navigation – and combining them.

In this framework, other components are required to provide data used for evaluating the options, along with components that will embody and execute the task. The ongoing work in creating such system is presented in the next chapter, along with already promising results.

Chapter 5

The Robot Guide

Contents

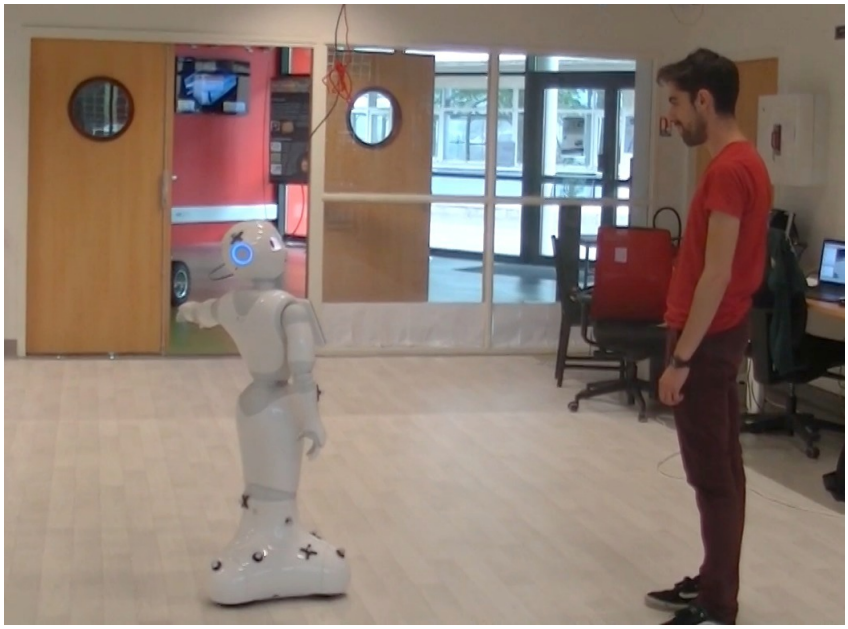


Figure 5.1: The robot describing the route in the environment.

5.1 Introduction

In the previous chapter, we presented a planner designed for the task of providing route directions, with the robot able to point at landmarks, provide route directions using them, and navigate with the human to places where the landmarks are visible. The planner takes into account the various objectives, constraints and parameters of the task to find the place where both human and robot should move to in order to get a satisfying solution for the human. In the previous chapter we focused on the task definition and the planner itself. We showed that the SVP planner is central in this task resolution but it is dependent on information about the route, its description and the landmarks that could be useful to point at, along with the model of the environment. Also, the implantation on real robot requires perception of the human, localization, navigation, motion, and a dedicated supervisor.

In the following we will present the integration of SVP planner in a working system, which components are being developed by some of our partners in the MuMMER project or teammates. We will briefly present these components, and focus on the architecture and in-lab tests results.

The components we developed and integrated as of now are at the number of six:

— A situation assessment module that maintains what the robot and the humans know about the scene in order to reason about beliefs;
— SVP planner that computes where the robot and the human need to be placed in order to point at and describe the route toward selected landmarks;
— A human-aware navigation module to move from a place to another or to

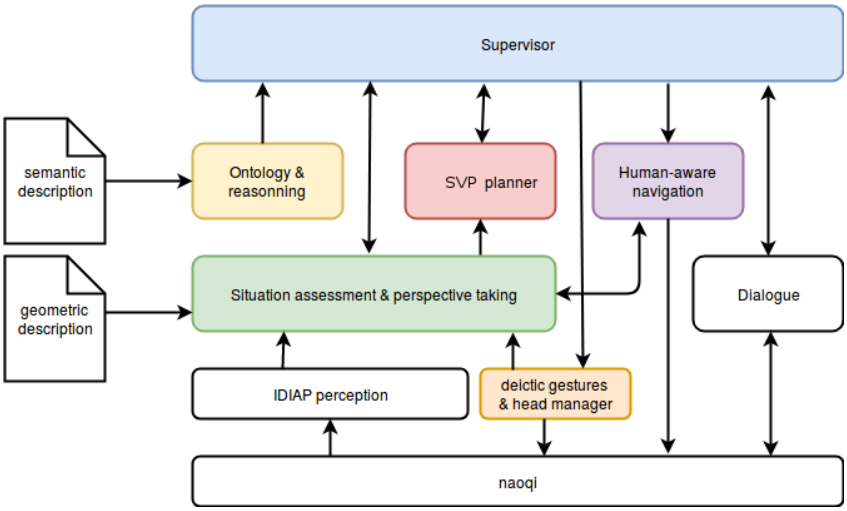


Figure 5.2: The general architecture of the system

approach a person;
- A semantic route description module that produces a semantic description to be verbalized by the robot;
- An head management module to handle where the robot has to look, according to the signal received from the other components;
- A supervisor to coordinate the above modules, monitor the interaction and repair or recover on execution issues.

Here, we voluntary do not present the perception and the dialogue parts because they were handled by other partners of the MuMMER project. The task has been implemented in a laboratory that has been adapted to emulate a shopping center. However, the architecture is generic enough to be extended to other environment.

5.1.1 Acknowledgements

The architecture and integration presented in this chapter is a team work, and the individual components other than the SVP planner presented previously are developed under the supervision of Dr. Rachid Alami by their respective authors:

Human-Aware Navigation from Guilhem Buisan;

Supervision is developed by Amandine Mayima;

Situation Assessment components are developed by Yoan Sallami;

Semantic Model and Ontology is from Guillaume Sarthou;

Head Manager and Pointing motions mainly developed by Guilhem and Yoan.

5.2 Global architecture

We consider the task to be a joint activity, we base our work on previously existing architectures for human-robot joint action. [Lemaignan 2017, Devin 2016]

5.3 Component Descriptions

5.3.1 Geometric and symbolic models

In order to plan for motion, navigation and also compute perspectives from anywhere in our environment, we need a geometric representation of it. It is a CAD model of the building made of named groups of three-dimensional meshes. Each group is an object which can be considered individually as a landmark and an obstacle. Window glass are not considered for perspective computations, but they are for collisions. This environment model is shared between the various components needing it, it is a standard format for CAD files (in our case we use blender format, but our softwares uses the assimp library [1] which supports many file formats). Figure 5.3 represents the model of our lab building disguised in a mall with some landmarks; the image also shows the names of the objects.

The routes to reach a location in the environment are computed using a topological representation of the area served by the robot. It represents areas (regions) linked by interfaces (gateways, stairs, elevators,...) and corridors, as pictured in Figure 5.4. The topology is managed by an ontology which also knows about all the shops or places with names and categories. This representation is similar to the one presented in [Satake 2015a], and like they do, it would be possible to describe in this ontology the items sold by each store or the services provided to better understand the customer request.

In order to provide route perspective, we need to represent and name every pertinent element of the environment, and not only shops, so they can be referred to during the description of the route to the visitor. This is why we represent the topology of the environment, like recommended by previous studies [Morales 2015, Yeap 1988, Yeap 1999].

5.3.2 Route description

When giving directions to a person, it is important to choose a route according to its length, but also on the complexity of the route and associated description. Other characteristics of route elements that may influence the difficulty for the visitor to find its way should also be taken into account; these may be the presence of salient landmarks, of signposts indicating the route to the destination. The knowledge about the meaning of elements of the environment, their spatial and semantic relations needed to perform this reasoning is represented in an ontology. It describes the whole environment by expanding the spatial semantic hierarchy

1. The Open Asset Importer Lib `www.assimp.org`

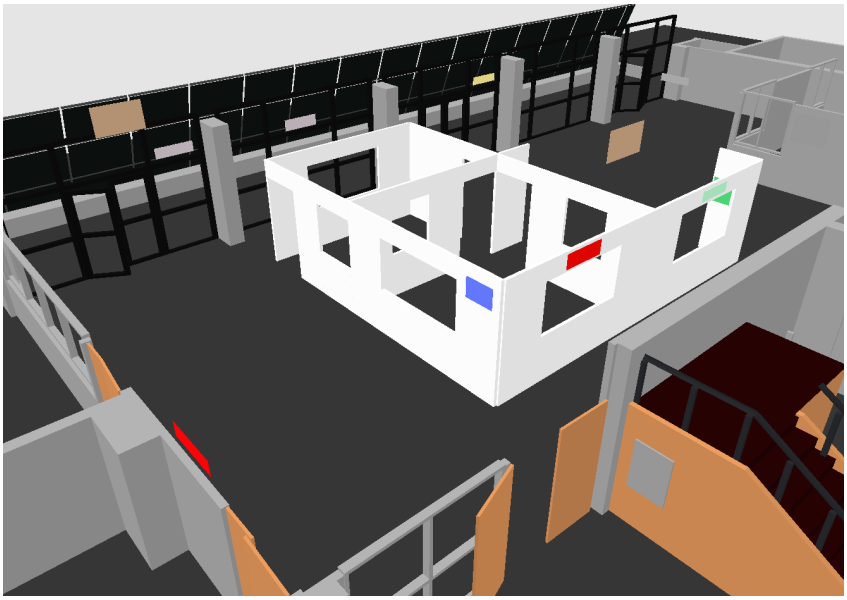


Figure 5.3: 3D mesh model of our lab environment, with signs added to serve as landmarks.

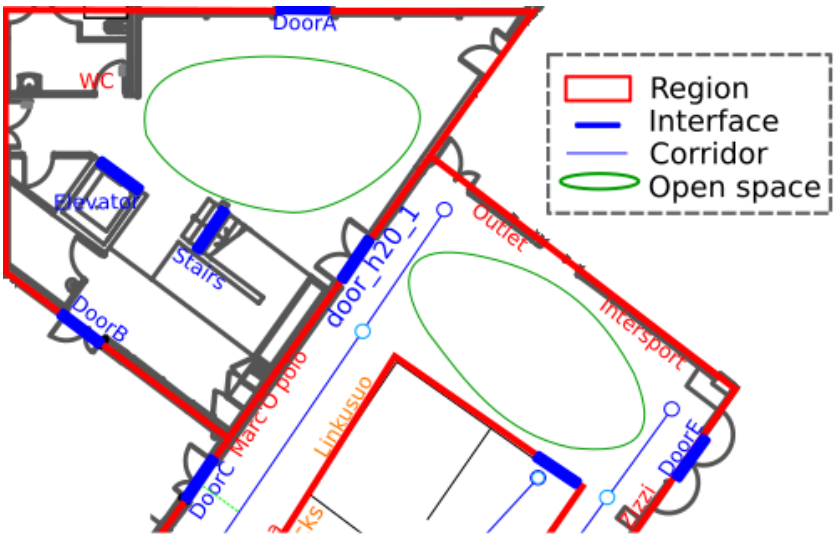


Figure 5.4: Representation of the semantic model of the environment used for route description.

that defines the concepts of places, paths and regions. [Kuipers 2000] refers to this knowledge as the topological level.

The knowledge about routes and environment in our ontology allows us to choose the best route as possible to use for direction giving. The relation between a signpost indicating the direction to the restrooms and the restrooms themselves can be used to provide route directions only to that signpost, the visitor then needs only to follow the direction of the signpost. Information about topological elements makes our system choose route without stairs if the customer is a person with reduced mobility.

The choice of the best route using the ontology knowledge is made by assigning to each possible route a cost, dependent on characteristics of the route elements and customer. A route R is made of N elements n, which are all associated to six characteristics noted σ_i, they respectively relate to saliency, accessibility, comfort, safety, explicability and speed. Customers characteristics are represented by a persona P which have a cost representing sensitivity to each criterion σ_i noted φ_{σ_i}. The cost function $f(P, R, M)$ determines the cost of a route R for a specific persona P considering the environmental context M:

$$f(P, R, M) = N \times \prod_{n=0}^{N} \prod_{i} (\sigma_{in} \times \varphi_{\sigma_i})$$

The current version does not consider length of the path yet; we focus on the complexity of the route description. However we expect our system to be able to use this criterion *e.g.* at the level of the SVP planner (see previous chapter, section 4.6.6).

Once a route have been selected, the ontology is queried to get two landmarks: the *target landmark* and the *interface landmark*. The *target landmark* indicates the destination, it can be a shop-front, or a shop-sign, or any salient element of the destination (a remarkable building, a statue,...). The *interface landmark* is the first landmark to indicate on the route for route description. It relates to the *interface* between the current *region* and the first one the visitor will have to reach, in our case they are mainly doors. Depending on the cases, there may be only one of these landmarks (either the destination is in the same region, hence there is no interface landmark, or the destination is not visible from the robot area –a far away place). Both the target and the interface landmarks may be used because in some cases the target is not directly reachable (different area) but visible. It is the case when it is on a different floor and it is visible thanks to a mezzanine or balcony (this is often the case in the mall of the MuMMER project).

5.3.3 Situation Assessment

This component is in charge of gathering the data from perception modules to produce a geometric and symbolic environment representation allowing the system to reason on the context. It also provides the system with theory of mind by

maintaining multiple alternative spatio-temporal models of the world. Each model is both a geometric scene tree and a timeline of predicates, thus maintaining the beliefs of the robot along with the inferred beliefs of the humans (first level theory of mind).

This component is based on Underworlds [Lemaignan 2018], itself inspired from previous systems, specifically SPARK (SPatial Reasoning And Knowledge) [Sisbot 2011] and TOASTER (Tracking Of Agents and Spatio-TEmporal Reasoning) [Milliez 2014]

Like in those previous systems, a 3D model of the environment and its objects has to be provided. Underworlds performs sensor fusion to place the objects of the environment accordingly. We use the exact same model as in the SVP planner, for consistency sake.

5.3.3.1 Input data from perception

In this work we use visual and auditive informations provided by perception in order to determine the position and orientation of the line of sight of the humans and if they are speaking (based on the direction of the sound and the face detection). The input are the following parameters :

— The head pose and gaze orientation for each human in the field of view of the robot;
— The human joint positions in the 2D image;
— The object the human is looking at, with a confidence;
— The probability of the human being speaking, using direction of the sound.

5.3.3.2 Perspective taking and gaze monitoring

Reasoning about the visibility is crucial in this application because we need different strategies if the landmark to show is or not visible from the human perspective. To know the visibilities of each landmark during interaction we have attached virtual cameras to each human and we compute the visibilities by rendering the meshes in the scene like in Figure 5.5. This algorithm outputs a list of predicates indicating which object is seen by which agent. The predicate is called $isVisibleBy$.

5.3.3.3 Beliefs computation

The task being addressed as a joint action, where the objective is to produce a change in the beliefs of the human concerning the route towards a desired place and its location. The subtask of pointing to objects also induce a change of the human beliefs. The situation assessment module aim is to maintain an estimation of the beliefs of each human we interact with. The environment model is filtered using the perspective taking for each human, the alternative world representation are then used to compute predicates added to the timeline of each human, hence representing their beliefs. This monitoring is done continuously (at the rate allowed

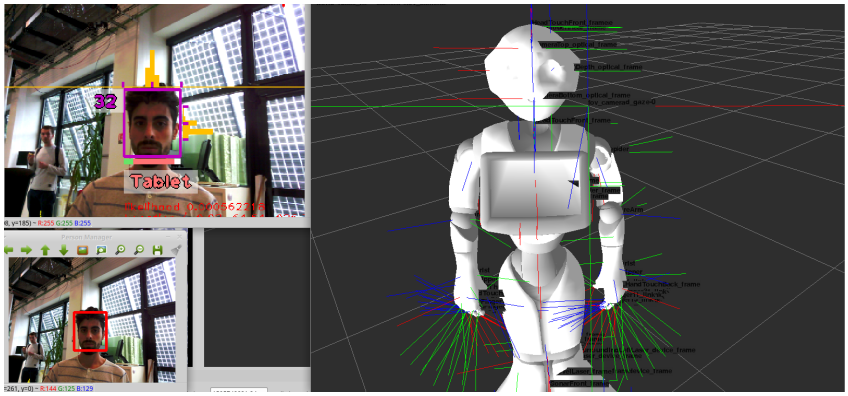


Figure 5.5: Perspective taking computed from marker-less detection. The left part represent the face detection (bottom) and person recognition and gaze estimation (top) overlaid on the camera image; the right image is the computation of what the human sees (or rather what the robot believes the human sees).

by the camera image rate or CPU power). The beliefs of about the locations of objects is pictured in Figure 5.6. This allows for example to know at any moment if the robot belief concerning a human belief like the location of a landmark, or put in other words, if the robot thinks the human saw a landmark.

This information is crucial to be able to get a feedback on the task execution and eventually recover and repair plans.

5.3.4 Supervision

The supervision component role is to coordinate and monitor the interaction as a human-robot joint activity, including engagement monitoring and plan repair. It has two submodules: the interaction component and the route description task component.

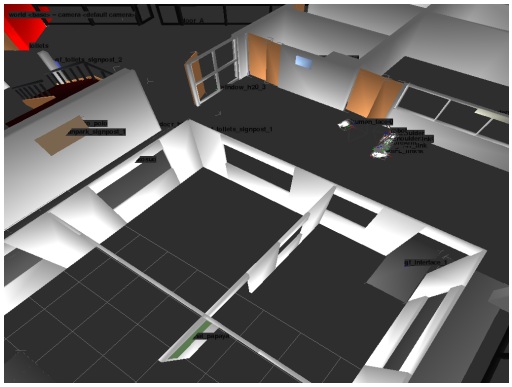
5.3.4.1 Interaction session

The Interaction component selects the human to interact with, and opens an interaction session. It also decides when the session closes, either because the human leaves or because the robot or the human closes it (*e.g.* by a goodbye). The selection of the human to interact with is done only when there is no interaction session already open, and it checks if the human is near the robot, looking at it and speaking to it.

During the interaction session the person can chat or play with the robot. Some keywords trigger a specific task, like the route description.

5.3.4.2 Route description task

The whole task is managed by this component controlling the calls to the various components and relaying information between them.



(a) The robot beliefs contain the entire environment and all the knowledge about the scene

(b) At initialization the human beliefs contain only itself. Then objects, landmarks or others agents are added and updated depending on the predicate *isVisibleBy*

Figure 5.6: The beliefs filter allow to compute the first order of Theory of Mind by knowing not only what the humans see but also what they have seen.

The implementation of this component is based on SMACH [Bohren 2011], a python library to build hierarchical state machines. SMACH allows to describe high-level actions and to handle their execution.

It takes as input the name of the destination the human partner wants to reach and it outputs the task success or failure.

The state machine, illustrated Figure 5.7, can be decomposed as:

1. Call the semantic route description component (5.3.2), which provides a list of routes, each one associated with a cost. The supervisor will select the best route, which also comes with up to two landmarks to use:
   — a landmark indicating an interface if the target is in a region different from the one currently occupied by the agents;
   — a landmark indicating the location of the target (like the shop-front, or a signboard indicating it), if it can be pointed at from the area accessible by the robot.
2. If a route have been found, the robot asks the human if it has to give them route directions.
3. The supervisor calls the SVP planner to get the optimal pointing position for the human and the robot with respect to the landmarks got in (1). The SVP planner also returns the landmarks that are visible from there:
   — the target or,
   — the passage or,
   — the target and the passage or,
   — none
4. The supervisor estimates if the human really needs to move, using the human position computed by the SVP planner: if it is close to the current position,

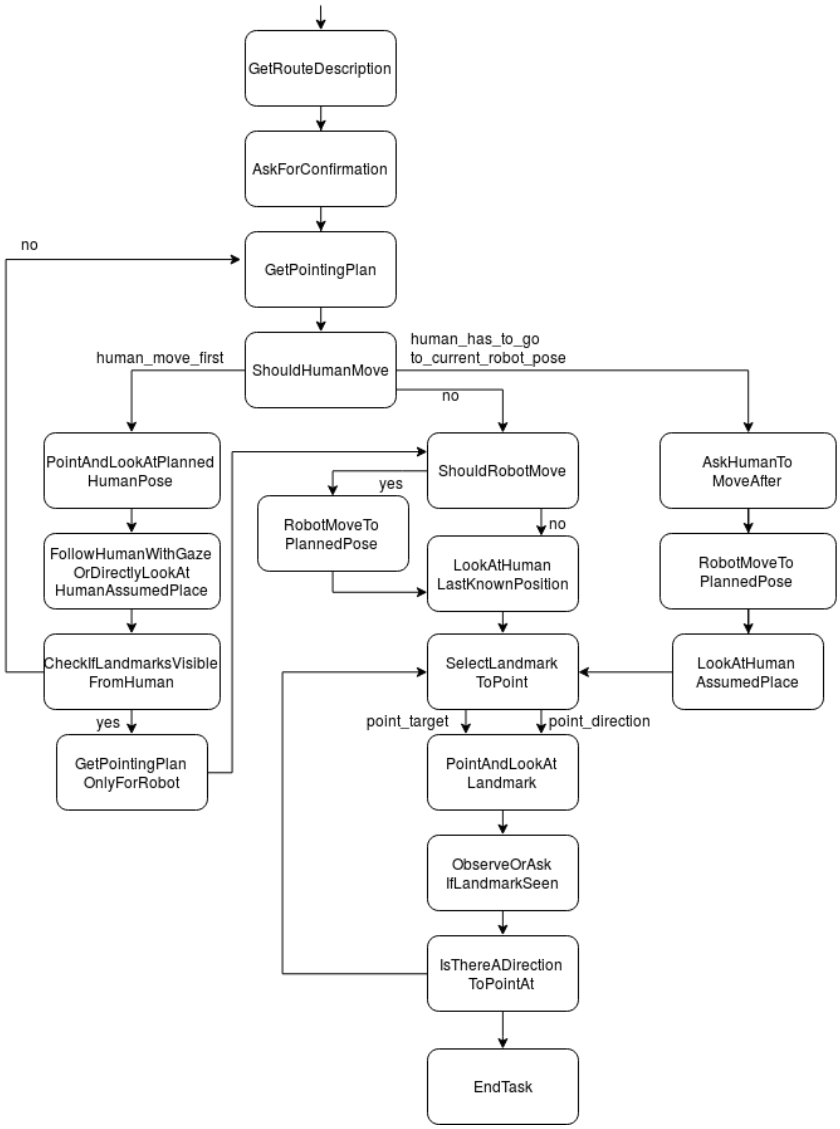


Figure 5.7: Diagram of the route description task.

the human will not be asked to move because the human will do it naturally without the need of a more explicit cue.

In the version currently implemented, we focus on scenarios where the agents do not navigate much, hence they can navigate turn by turn. According to where the agents are and need to go, the robot may or may not explicitly ask the human to move and point at the place she should go, explaining that it want to show something that is visible from there.

5. Now, the robot and the human are well-positioned and the robot can show the landmark(s). First, it points and looks at it and then looks at the human to check thanks to the situation assessment component if she is seeing it (i.e. looking in the right direction). If after a short time, it is not able to determine if the human has seen the landmark, it directly asks to her.

   If the human has looked at all the landmarks shown, the task ends with success. Otherwise, the robot asks if it should point at the landmark again.

5.3.5 Navigation

The robot navigation during an activity in collaboration with humans has to observe rules related to safety, acceptability and legibility. In our specific case the navigation of the robot can occur as a co-navigation with the human, needing a navigation component able to generate motion for guiding a person. The component featured in this project is based on previous work in our team: the Human Aware Time Elastic Band (HATEB) [Khambhaita 2017]. Currently, it provides two services related to our guiding task:

Approach a human to get the robot face a human. The planner dynamically adapts to the human when he moves;

Move to a location navigates to a location taking surrounding humans into account.

The particularity of HATEB is that it takes into account multiple constraints related to robot and humans to generate motions that are more legible and acceptable to perform a more efficient collaboration. It can also plan motion for both the human and the robot in order to better take humans into account, anticipating their trajectory a few seconds in the future to be more readable and not purely reactive, and generate deadlock-free navigation plans.

However, if the robot parameters used in the optimization are known by design, estimating those parameters for the human (e.g. their goal) is still a challenge. In our case, the robot shows the human goal position during the pointing, this allows the human goal estimation to be more accurate and thus, to propose co-navigation solution more acceptable. HATEB plans are quite stable, they do not change dramatically as the time passes and trajectories are dynamically recomputed, hence providing good estimation of future robot positions. We make use of this to have the robot look in the direction it is going, to again increase motion legibility of short term moves.

The original HATEB presented in [Khambhaita 2017] was however modified to suit to the needs of the *approach a human* task: in this case, the time-to-collision constraint, making the robot avoid navigating towards the human at high speed and low distance, was relaxed to expose an engaging behavior (still reducing its speed when getting close to the human). However the time-to-collision constraint is still active for navigation among humans to avoid threatening them.

Currently we do not take into consideration cases where the agents need to navigate further than a few meters, and we make the robot point at the location the human should go. This interaction is unnatural and so we plan to improve the navigation component so that it can co-navigate even on short distances with the human, simply asking to follow, until they get close to the planned positions. The robot would end its navigation by motions proposing a position to the human.

5.3.6 Head Management

The robot gaze is used in activities in collaboration with humans to increase attention, engagement and to show awareness, like previous work showed [Mutlu 2006, Zaraki 2014, May 2015]. In our scenario, the robot gaze is used to improve navigation legibility, direct attention towards the pointed objects or the human. The Pepper robot, like many humanoid robots, has most of its sensors attached to the head, which has to move to monitor the environment, humans, objects. The head of the robot then appears a critical shared resource, needing a dedicated component to enforce sharing policy.

We implemented the Head Manager component to centralize head move requests, choose and sequence them depending on the context. It takes in three kinds of head move targets:

monitoring target to direct gaze and sensors towards attention point;

speaking target to look at the human the robot is speaking with;

acting target to provide readability of robot actions, like in the navigation task (see section 5.3.5).

The targets are provided as absolute position in the robot world, along with a priority. They are provided by the respective components requesting the head moves, receiving a target of some type discards pending targets of the same type.

Another type of input is available: coordination signals. They always preempt any other target, and allow the robot to send potentially complex collaborative cues. This input is a list of points to look at, with conditions specifying when to look at the next target. Conditions can be a duration or a predicate from the situation assessment component (*e.g.* look at the human until *human isLookingAt robot* becomes true, then look at the pointed target).

Except the coordination signals which always have priority above other targets, targets are chosen according to their priority weighted by a task-dependent value provided by the supervisor for each target type.

5.4 Experimental Results

At the time of writing we are testing our system in our laboratory building, to which we added virtual stores and real landmarks representing them. Various scenarios where run to test and demonstrate the various aspects of the system and its adaptability. The following parameters where varying:

— destination landmark existence
— interface landmark existence
— landmark(s) visible from human initial position
— landmark(s) not visible from any nearby position
— visitor height
— visitor having a shopping-cart
— visitor distracted while robot speaks (and recovery by the robot)

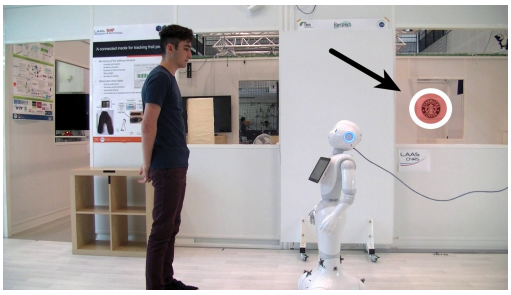
We shot some of the tests, and we will present in the following some pictures extracted from theses shots. Figure 5.8 shows a sequence of snapshots for a simple interaction: the visitor asks for the location of a shop, which is visible only if he moves a few steps. The robot asks him to move to such location it points at, checks the human gets there, navigates close to him, says that the shop is there while pointing at it. The robot, while pointing, also looks at the landmark it is pointing at to perform shared attention, then looks back at the human and asks if he saw his destination. And he did, says so to the robot, and the visitor can simply walk to his destination.

Figure 5.10 shows two different runs with only one difference: the height of the subject eyes; all other parameters including the initial positions are identical. The initial position of the human is approximatively the one pictured in the figures 4.4 and 4.5 (Chapter 4): from there, a tall person cannot see the landmark, but a smaller one can. The SVP planner gets this information from the situation assessment, and computes a position from where the robot and the human can see the landmark.

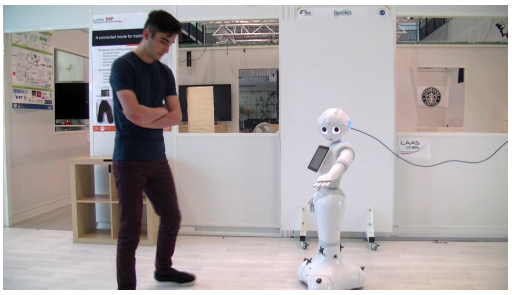
When the destination of the person is on another floor, the supervisor makes the robot ask the human preference concerning the use of stairs or elevator. This information is communicated to the route planner which applies corresponding penalties to the computed routes. The preference about the use of elevator could be inferred from perception in some cases, but we have no implementation for the detection of cart pushers (figure 5.11a) or wheelchair riders.

We also tested for the recovery behaviours implemented in the supervisor. Two cases are considered: the human does not go to the planned position the robot indicated (as in figure 5.11c), and the human says he did not see the landmark the robot pointed. For now, in both cases the robot only repeats the corresponding actions, only once. It does not try to recover after a second failure and simply apologises and closes the interaction.

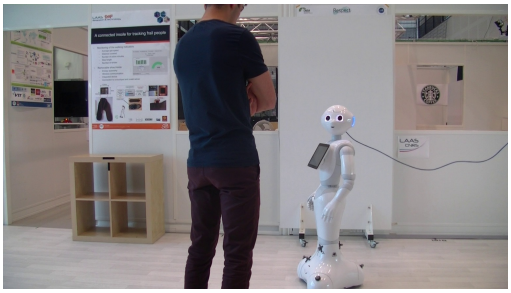
During the deployments and tests, we encountered several difficulties related to perception limitations. Many errors where due to an approximative localization of the robot, giving errors on its position and orientation in the scene, hence producing inaccurate pointing gestures. The position found by the SVP planner for the human



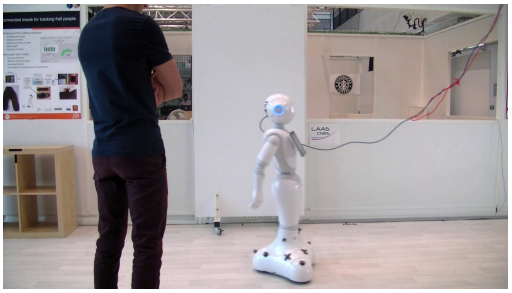
(a) H:“I’m looking for Starbucks”



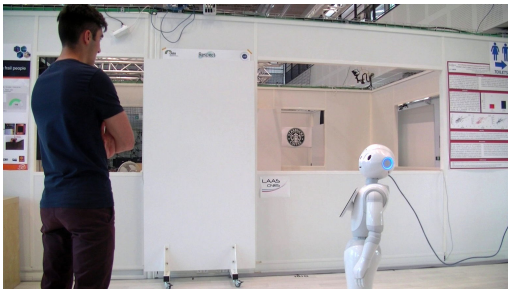
(b) R:“Can you go there, you’ll see better”



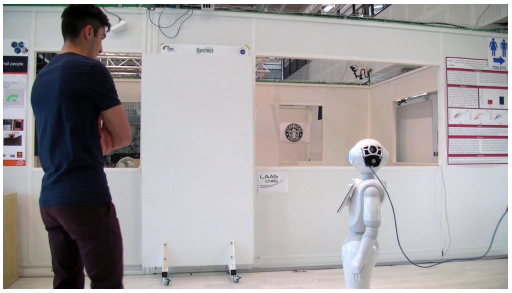
(c) Robot checks the human position...



(d) ...before navigating to its own position.



(e) Robot points at the Starbucks shop sign



(f) and looks at it to start shared attention

(g) R:“have you seen where you have to go?”

(h) Human perspective from the planned position, the shop-sign is at the top left of the picture.

Figure 5.8: A simple example where both the visitor and the robot have to move a few steps to see the destination. The target landmark, a shop-sign, is highlighted in (a) and (h). In (g) we can see the person did not go where planned, and hence had to bend slightly on the right to see the landmark; he naturally did it to try to better share the perspective of the robot.

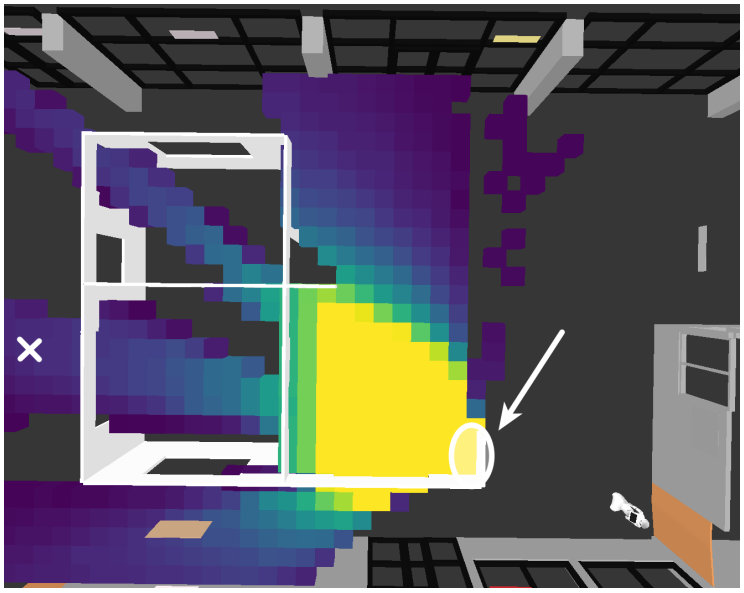


Figure 5.9: Visibility grid from a height of 1.9m of the Starbucks shop sign.

is communicated by a pointing gesture, and an error could lead the human to get to a place where they actually don't see the landmark. In figure 5.8 such error is visible, the human needs to bend slightly to see the pointed landmark.

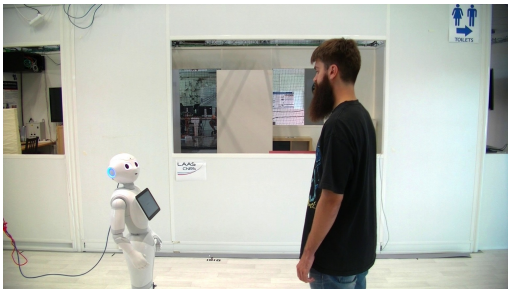
An other difficulty we are facing is related to the human reidentification. The guiding task leads the robot to often stop tracking the human face, during navigation or pointing. The need for a system able to robustly reidentify partners strongly appeared in this task. Even more for a system supposed to work in a crowded environment.

5.5 Future Work

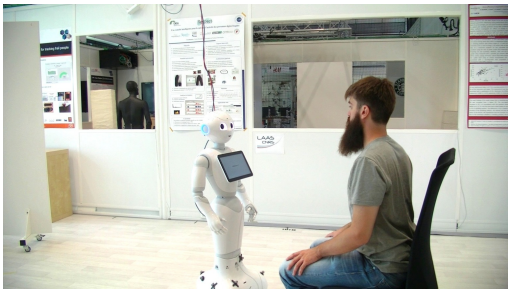
We presented here a system fully operational to provide route directions to humans using the Shared Visual Perspective Planner presented in the Chapter 4. The development of the system was also providing a feedback on the design choices of the SVP planner, allowing to evaluate the whole task more accurately in the search, and hence produce a better solution to be executed.

In activities involving the collaboration with humans, our claim is that this accuracy in the planning can only be attained by highly integrated components, where informations are shared between them and planning components are enabled with a knowledge on how the task is executed.

We are looking forward to further integrate and test our system with our partners' work in the scope of the MuMMER project. Preliminary work was done to integrate it with perception tools for human reidentification, and to trigger the task from the versatile social dialogue system Alana [Papaioannou 2017a]. We successfully integrated with the Alana system to understand human speech (requests and



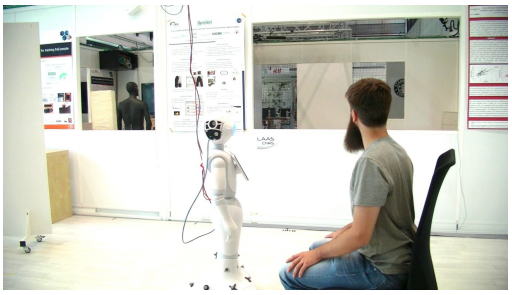
(a) initial position with a tall person



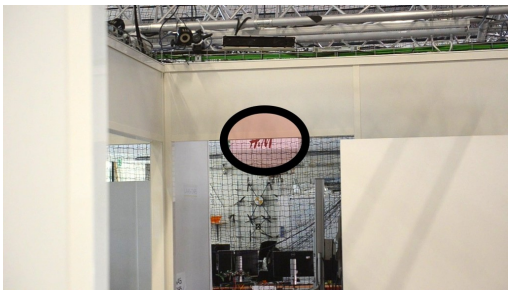
(b) same initial positions with a sitting person

(c) perspective from the tall person initial position: the landmark is not visible (behind the wall part above the door opening)

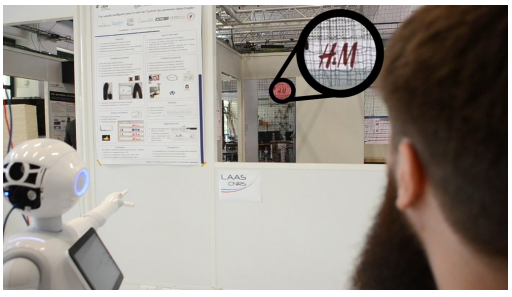
(d) SVP planner found positions from where the landmark is visible



(e) ...while with the sitting person, there is no need to move.



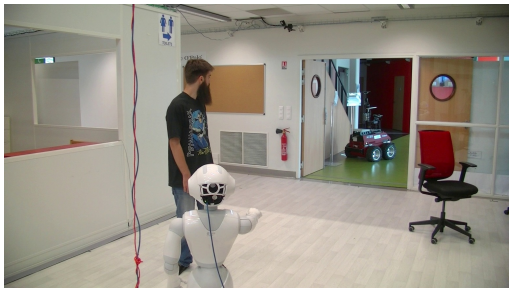
(f) From the planned position, the landmark is now partially visible



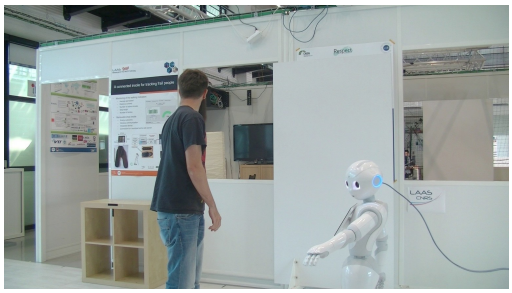
(g) Perspective from the sitting person

Figure 5.10: Two identical cases are presented here, at one exception: on the left-hand the person is too tall to see the destination landmark from its initial position, and on the right-hand the person is sitting and can see the landmark from there.

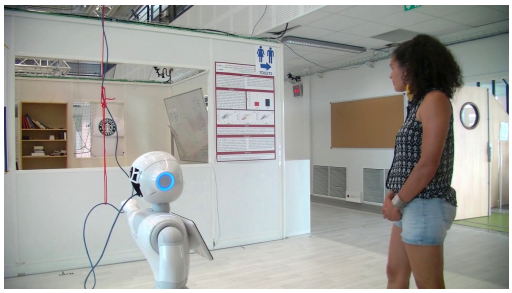
(a) The visitor has a shopping cart, and his destination is at the first floor, the indicated route uses the elevator.



(b) As in (a), but when asked the visitor request to take stairs



(c) Here the visitor is distracted while the robot shows him where to go. The robot then perceive that the human did not move, and supervision level makes the robot repeat its instruction.



(d) Another example with the robot pointing at the Starbucks sign.

Figure 5.11: Some miscellaneous excerpts from other test videos.

answers), provide indications and instructions to the human, while the human could interrupt the task normal flow to chat with the robot and then continue the guiding task.

Chapter 6

Action Planning for Human and Robot Teams

Contents

6.1 Introduction

The tasks presented in this thesis, like most manipulation tasks, fetch & carry, handover, rely on finding some intermediary positions before computing motions linking them. For a pick-and-place action, a robot searches for a valid grasp, a place where to put the object, and tries to find feasible motions to reach these positions; in case of failure, it could try a different grasp, or a different place, until a valid solution is found. In the case of a friendly robot, feasibility is not enough, we want more properties on the behaviour of the robot. In our work, these properties are represented as objective functions of an optimization problem. Also, these wanted properties may vary depending on the task we want the robot to perform, the robot itself, the environment or the human(s) involved in the task.

When interacting with several humans, or most importantly in a human environment (public places, offices), the robot must behave differently with respect to each individual. Regarding a property based on proxemics, that dictates the robot to stay at some distance from the humans to respect their comfort, this rule must be respected mostly for humans not taking part to the task. A fetch & carry task illustrates this: the robot may need to navigate around to humans not related to the task, but in the end it will have to handover an object to a specific human, needing to approach them in a closer interaction distance.

Our idea is that in this context, it is important for all the planners, from symbolic to geometric, to share information about the task being planned, in order to make the right decisional choices at the right level of abstraction, with appropriate algorithms and objective functions.

The framework initiated by Mamoun Gharbi under the supervision of Rachid Alami, the Geometric Task Planner (GTP) [Gharbi 2015b], a tool linking symbolic and geometric planning, is aware of both planning spaces. We developed a functionality that configured the motion-planning algorithms and refining steps according to the task being executed.

This chapter will first present the GTP framework basics, and then show how we use it for planning collaborative tasks.

6.2 Concepts

We will introduce some concepts used in our framework, as a simple formalization of the problem solved.

6.2.1 Actions

An action in our framework is defined by symbols as commonly used in task-planning [Ghallab 2016, Lozano-Perez 1987, Alami 1998]. In this context, the goal is rarely, if not never, specified as a unique and complete specification of a geometric and symbolic state. Rather, the goal state is specified in terms of symbols and geometric relations to reach, that can be represented as constraints – often non-linear

– on the target world-state. However the full geometric and symbolic specification of the initial state of the world is needed to start the search from there.

For being solved, we split actions in several elementary steps consisting of a single trajectory executed by an agent, ordered with simple temporal constraints (*starts after*, *starts concurrently*, *no synchronisation*). Each elementary step corresponds to a single motion solved with a standard motion-planning algorithm [LaValle 2006]. This implies we know the full initial and final world-states of each step to completely solve an action. Finding these world-states (except the first one, which is a required input) is the main challenge of action planning and is solved with specific algorithms for each action, we will not present them in details here.

6.2.2 Action Request (input)

We define a way of describing an action in a generic way, to have a uniform request format; it is formed only of symbols. The elements of the request are:

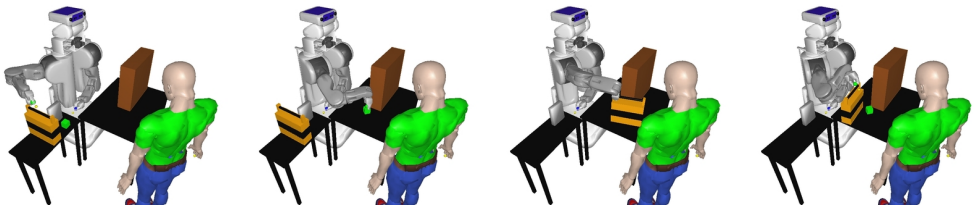
— Action type: the kind of action we want to compute
— Main agent: the agent performing the action
— Target agent(s): other agents to take into account
— Main object: manipulated object
— Support object(s): supports we use (e.g. for placements)
— Target object: a target for navigation actions
— Arm: arm, hand or gripper of the main agent to use
— Target coordinates
— Constraints on the final state: symbolic facts that must be observed in the end state (see below)

Each action can use different subset(s) of these parameters. Some can be optional, like a support to place an object, in that case the solver will have to choose it also.

To compute an action, we need to know the initial state of the world. The action is computed on a "snapshot" of the world state, like motion planning usually does. The world state used to plan an action can either be the current world state (from sensing), or a world state computed from a planned but not yet executed action. The former is used when planning and executing only one action at a time, the latter when computing a longer plan, a list of actions, before its execution.

6.2.3 Action Solution (output)

An action solution is a feasible motion that reaches a world-state satisfying the constraint related to that action definition, and is represented as an ordered list of trajectories. Whenever GTP cannot find a solution, because of invalid prerequisite, infeasible plan, or missing information, it produces a status report where the reason of failure is detailed when it can be known. GTP also outputs facts computed over the final state.



(a) initial state. (b) robot places first object reachable and visible by human. (c) the second object breaks the visibility and reachability of the first one. (d) both object affordances are respected.

Figure 6.1: Example of action planning with fact preservation. The request is a sequence of two pick/place reachable actions, with preservation of the visibility and reachability affordances of the human. (a) shows the initial state with three objects (an orange box, a green cube and a grey book). (b) shows the end state of the second action, where the robot placed the green cube reachable by the human (the first action is the pick of the cube). (c) is a possible end-state of the fourth action that breaks the constraints on the green cube affordances, while (d) preserves it. Figure 6.2 shows the input given to GTP to produce this solution.

6.2.4 Facts

Facts are symbolic expressions of relations between objects or agents in a given world-state. They can be used to determine symbolic effects of actions, to inform higher levels of planning. They can also be given as constraints over the final world-state of an action, *e.g.* in order to preserve effects of previous actions, or initial facts. We define a few facts, either relating objects: `isOn`, `isIn`, `isNextTo`; or relating objects and agents (affordances): `isReachableBy`, `isVisibleBy`. GTP can be asked to solve an action (e.g. *place Box on Table*), while respecting some additional constraints (e.g. *Cube is reachable by robot and by worker*). Such input is shown in Figure 6.2, corresponding to the solutions illustrated in Figure 6.1.

6.2.5 Sequence Planning

Instead of planning a single action, GTP can be given a sequence of them, $\{a_0, \ldots, a_n\}$, allowing backtrack to solve or improve the plan. Backtracking involves finding an alternative solution to an action a_j when GTP does not find an acceptable solution to an action a_k, with $j < k$. Backtrack is intended (1) to find feasible plans, when an action influences the feasibility of a following action, (2) to respect affordance constraints over the final state of the sequence, (3) improve the global quality of the sequence, with respect to human comfort or efficiency. Sequence planning is illustrated in Figure 6.1.

The manner the plan is refined can influence the performance of the planner. When planning in sequence, motions can be computed only once the whole sequence of actions have been found, when all action solutions have been found and lead to a valid final state. This saves time as the motion planner is called only once for each

```
action pick:
    mainAgent:PR2_ROBOT,
    mainObject: GREEN_CUBE,
action placeReachable:
    mainAgent: PR2_ROBOT,
    supportObject: TABLE1,
    constr: GREEN_CUBE isReachableBy HUMAN1,
    targetAgent: HUMAN1,
action pick:
    mainAgent: PR2_ROBOT,
    mainObject: ORANGE_BOX,
action placeReachable:
    mainAgent: PR2_ROBOT,
    supportObject: TABLE1,
    constr: ORANGE_BOX isReachableBy HUMAN1,
    constr: GREEN_CUBE isReachableBy HUMAN1,
    targetAgent: HUMAN1
```

Figure 6.2: Symbolic part of the request for the example presented in Figure 6.1, as a list of action requests. Each indented block contains the exhaustive list of symbols used to compute an action, including the constraints expressed as facts.

action. In cluttered environments, the user can choose to plan motions whenever an action solution is tested, to ensure their feasibility before going on planning the next action.

Sequence Planning is related to the fact that the different actions of a given plan might strongly influence each other, and GTP addresses this by considering the various way a single action can be solved.

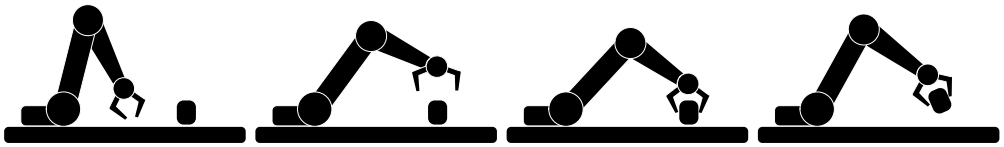
6.2.6 Models

We aim at creating a generic framework for any kind of robot, hence users can add their own robot to the framework with limited effort.

6.2.6.1 Geometric Model

The environment must be given to GTP at initialization. It is composed of static obstacles, movable objects, humans and robots. Individual robots, objects and humans are defined in URDF (Unified Robot Description Format [1]). URDF allows to describe robots made of rigid bodies and joints between them. As GTP may be required to plan actions for humans, we need a model of humans compatible with our tools, that is a robot description. Humans can be represented with various levels of detail according to the task needs, including a complete kinematic representation of the human.

1. http://wiki.ros.org/urdf



(a) Initial (b) Approach (c) Grasp (d) Disengage

Figure 6.3: From an initial configuration (a), GTP decomposes the pick action in three steps: approaching the object (b), grasping it (c), and lifting it away from its support (d). The configuration in (a) is provided as an input, the three other are generated by the *pick* planner, then motions are generated between each successive configuration.

6.2.6.2 Planning Parts

The definition of an action holds a symbol representing the parts of the robot they need: *navigation part* for navigation task, *manipulation part* for manipulation tasks. Those parts, labelled lists of degrees of freedom, are defined for each robot, which can have several of each type (*e.g.* one for each arm). The ability of a robot to perform an action is defined by the existence of the corresponding planning part.

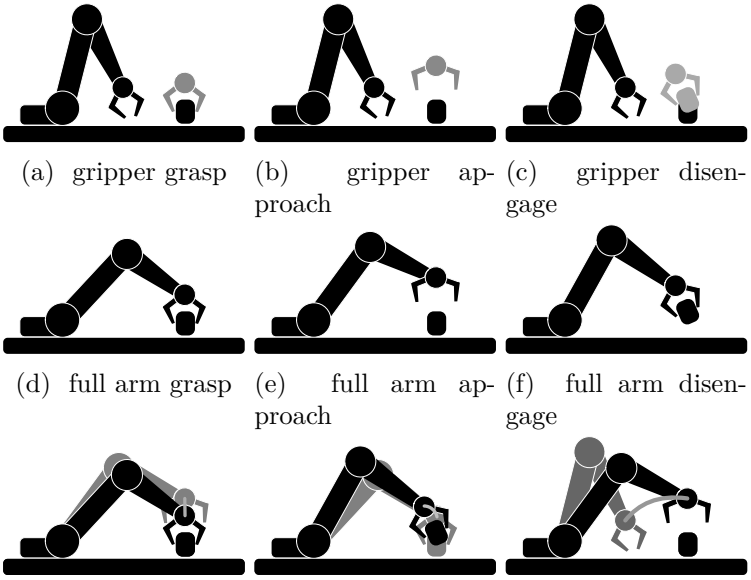
6.2.6.3 Grasps and Surfaces

For each manipulation planning part, an end effector must be defined. Grasping configurations are defined for each pair of end effector type and object type. Stable configurations must be defined for manipulable objects. GTP must also know the support objects where manipulable objects can be placed (table tops, shelves,...).

6.3 Algorithm

6.3.1 Basic Task Solving

If we consider only the feasibility of the task, and forget about optimization criteria and human preferences, a task solver generally works as follows. An initial state is given, and a goal is represented by symbols (*e.g.* the object X is picked by the robot R). To get a motion that reaches a world state satisfying the goal, we need to instantiate such world state (in our example, a world state where X is held by gripper of R). In some tasks, there are intermediary configurations that are very constrained, and can hardly be found by a motion planner (*e.g.* the grasping configurations are almost in collision, and grasping an object changes the geometry and relations of the objects). The approach of GTP is to chose those configuration and split the motion plan in several parts. The choice can be made at random (*e.g.* selecting a grasp among all the possible grasp). Once all the intermediary and final world states are chosen, motions are generated (see Figure 6.3). This ordering of the search is designed to avoid spending much time in the motion planning, which can be very slow when solution quality is sought.



(a) gripper grasp (b) gripper approach (c) gripper disengage

(d) full arm grasp (e) full arm approach (f) full arm disengage

(g) grasp motion (h) disengage motion (i) approach motion

Figure 6.4: Computational time required to find valid positions and motions is improved by simply searching for them in the order of the above figures. We start by searching for gripper-object only valid positions (a-c); we then compute the full arm configuration for these positions (d-f); and finally computing the motions (g-i). The approach motion is computed last as it is generally the longest step, while all the previous steps are extremelly fast in comparison. Whenever a position or motion is not valid, the candidate is discarded and eventually an other one is selected. This ordering guarantees to spend the least time possible in evaluating candidates by discarding invalid ones quickly (with respect to other orderings).

6.3.2 Find a Good Solution to the Task

Most of the time, a feasible functional solution is not enough, a *good* one is required. The preferences are represented as cost functions matching a configuration, a world state or a motion to a scalar value. A naive approach would be to run an implementation of the basic task solver introduced above many times, and as it is stochastic, it would produce different solutions each time. We could then just pick the best solution. We could also compute only one solution and try to optimize it. Such approaches are clearly under-efficient and some improvements can be made to the basic task solver to make it more efficient in finding good solutions.

The first one is to simply use well-known optimal motion-planning algorithms like T-RRT [Mainprice 2011]. It provides better motions but they are still constrained by the final and intermediary configurations generated by the task solver. Again we could sample these configurations, compute a multitude of motions for these alternatives and choose the best one. But optimal motion planning is a computationally expensive task, so it is preferable to call it only on sets of configurations that are promising to give a good result.

So we iteratively refine the solution. We first sample possible intermediary configuration sequences, we compute a cost for each configuration and each sequence (called a *candidate*). We select the best sequence based on this cost, and compute the motion plan for it. If the motion planning fails, we select the next best candidate. The solution found is not the optimal one, but we intend to reach good quality.

6.3.3 Detailed Algorithm

The main algorithm is detailed in Algorithm 2. It first creates a list of candidates: alternatives final and intermediate world-states, sampled among all the possibilities (l. 2). Each world-state is checked against fact constraints if any; all candidates breaking the constraint are removed from the list. Then all candidates are evaluated (l. 3) computing the cost of each world-state. They are sorted from lower to higher cost. Motion plan is computed for each successive candidate until a valid one is found.

Note that the motion planning is done after the candidate selection, so the motion cost is *not* taken into account for that selection. Indeed, computing motion-plans for each sampled alternative would be too computationally expensive. In other words, we use the world-state cost as an heuristic for the costs of an alternative and we do not use the "real" cost that includes the motions.

The number of sampled candidates for each task is domain- and task-dependent. If none of them is valid, the task is discarded and the planner fails for that task. In the case of sequence planning, an alternative plan is searched.

6.3.4 Components

Here are detailed the main components of the algorithm.

Algorithm 2 Action computation algorithm

```
1: function ComputeAction(t)
2:     candidates =SampleSolutions(t)
3:     Evaluate(candidates)
4:     sort(candidates)                          ▷ sort by increasing cost
5:     while no solution and candidates not empty do
6:         cur=getNextCandidate
7:         ComputeTrajectories(cur)
8:     end while
```

6.3.4.1 Human-related Costs

Generating socially acceptable actions requires fitting to some constraints. We define them by costs, used in several steps of the action computation: when selecting the possible positions and configurations and for motion-planning. Cost of a motion reflects its safety, legibility and human comfort. We define costs for configurations, trajectories and actions (solutions), that reflects the same properties and are used at different steps of the computation. As explained in 6.3.2, the motion cost generally does not influences the alternative selection, but it is used during the motion-planning to generate good quality trajectories.

6.3.4.2 Alternatives and World-State Instantiation

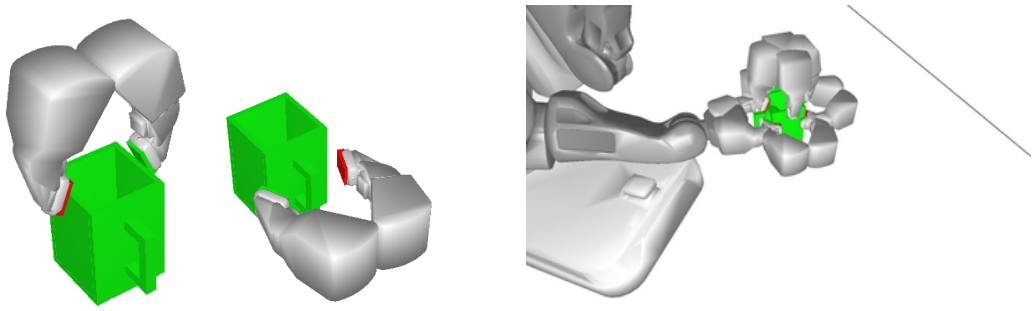
As discussed earlier, the main challenge of action-planning is finding the target and intermediary world-states corresponding to the requested action. The choices are to be done within a sub-space defined by constraints specific to the action or imposed by the user. An *alternative* is an instantiation of all the world-states needed for a single action. Since the number of possible alternatives for an action is virtually infinite (due to position continuity), they are sampled in different ways, according to the action and the type of variable. They are either discrete (arm to use for a pick action) or continuous (target position to approach an object). For small number of options, like arm to use, we exhaustively search through all the possibilities. For large ones, like positions, we use sampling to find good options, based on the human-related costs (here we could use optimization methods to find a (locally-)optimal solution). Examples of *choices* are presented in the table 6.1.

6.3.4.3 Configuration Computation

In the process of world-state instantiation, we use optimal inverse-kinematics (IK) algorithm to compute robot configurations based on object positions and grasps, while respecting position constraints and optimizing the human-related costs. Figure 6.5 shows possible grasps for an object and the IK solution (robot configuration) computed for the chosen one.

Table 6.1: Examples of choices to make for some actions.

Action	Fixed parameter	Choices
Pick	Object	Arm Grasp
Place	Support	Object position
Place		Support Object position
NavigateTo	Target object	Robot (x, y, θ) position



(a) Example of two possible position of the PR2 gripper to graps the mug. Similar grasps are defined all around the mug.

(b) Sampled grasps for a mug (including those of (a)) and full robot configuration for one of them, for the PR2 Robot.

Figure 6.5: GTP samples among possible alternatives to select the best one. Here it chooses the position of the robot gripper to grasp an object. With the good heuristic, a pertinent choice can be done so the subsequent motion will be of better quality.

6.3.4.4 Motion Planning

Computing all the trajectories based on the initial state and on the configurations found in the previous step. We use a cost-based motion planner with human-related costs [Jaillet 2010, Mainprice 2011].

6.3.4.5 Facts

These predicates are computed on a world-state, so they can only represent static properties. The framework proposes to the user different ways of computing each type of facts (influencing on their accuracy and computational cost). By default, only facts needed for the action (either required by the action definition or by user input constraint) are computed, but one can ask to compute facts on each action final world-state. Those automatically computed facts can be used by higher levels of reasoning.

Algorithm 3 Sequence solving and backtracking algorithm

```
    function ComputePlan(list)
        plan_cur = {}
        while len(plan_cur) < len(list) and time < time limit do
            sol = ComputeAction(list[len(plan_cur)])
 5:         if sol then
                append sol to plan_cur
            else
                Backtrack(plan_cur)
        end while
10: function Backtrack(plan)
        sol=pop back plan
        sol_new = ComputeAlternative(sol)   ▷ fails when too much alternatives
    have been computed for this action
        if sol_new then
            append sol_new to plan
15:     else
            Backtrack(plan)
```

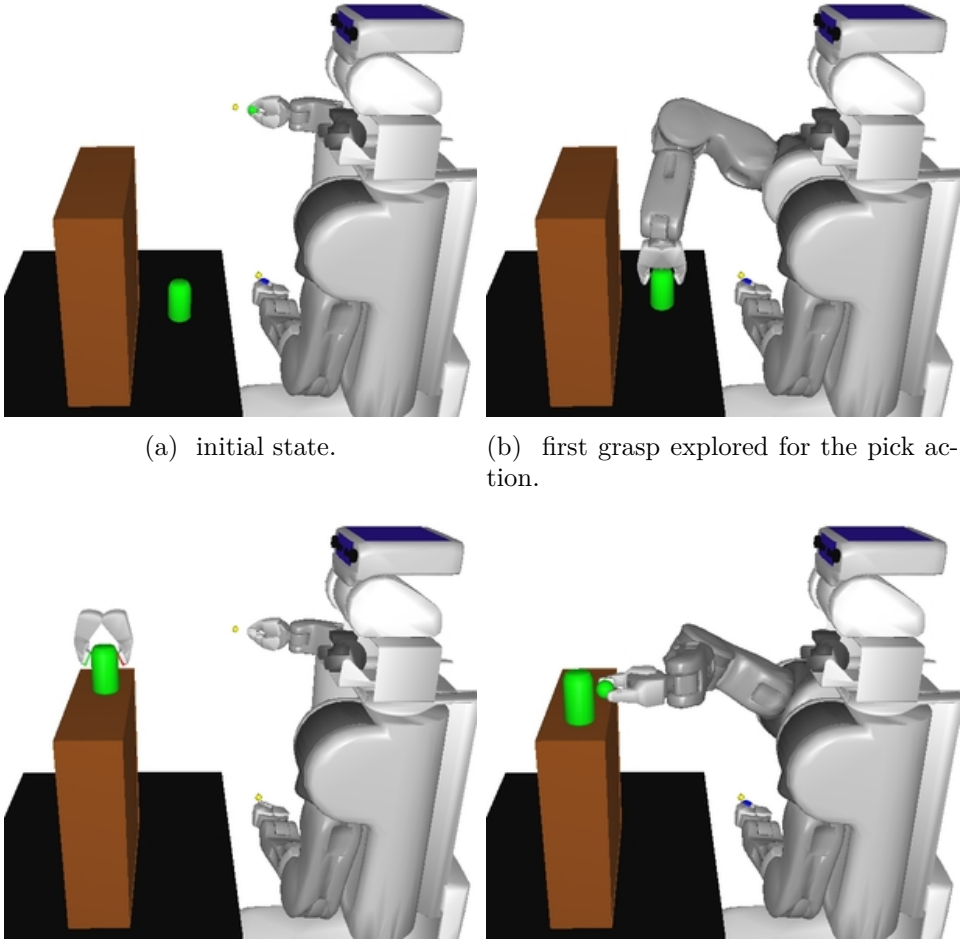
6.3.4.6 Backtracking

The backtracking (see Algorithm 3) is possible only when planning a sequence of actions. It focuses on choice instantiations, *i.e.* grasp selection, place positions and so on. The backtracking is triggered whenever no solution is found to an action; it tries to find an alternative to the previous action and recomputes following actions. If a certain number of alternatives have been tested for an action it is discarded and the backtrack goes one step deeper (line 12). This number of tested alternatives per action is typically set to 10, and determines the rate at which the algorithm will backtrack deeper in the history, discarding complex actions (*i.e.* with much invalid alternatives). This approach is very naive but efficient for most short plans, when the domain is not too constrained or geometrically cluttered. Figure 6.6 shows an example of backtracking on a constrained pick-and-place. As mentioner earlier, backtracking could also be used to improve solution quality by searching for better solutions of individual actions that improve the global plan, but this is not implemented nor tested yet.

6.4 Planning Motions for Human-Robot Joint Action

We will highlight in this section our contributions to GTP to make it able to generate solutions to tasks that complies with what is expected from a robot acting in collaboration with humans.

The framework was built by its authors with the intent to solve planning problems with high dependency between symbolic and geometric levels, with multiple agents, including humans who could be involved in the shared plan. The human was



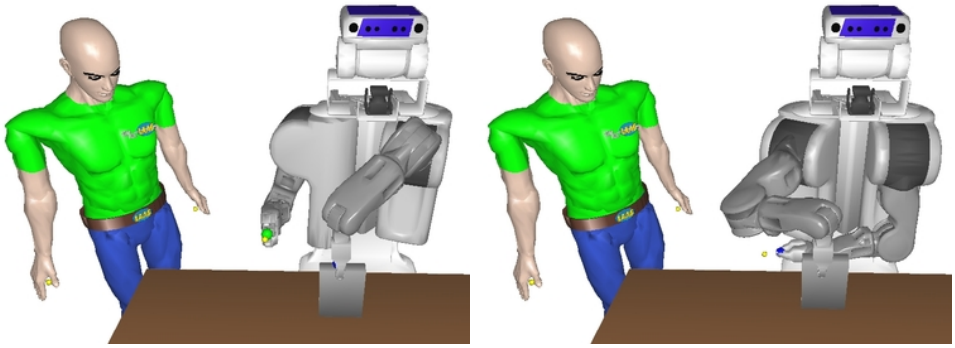
(a) initial state.

(b) first grasp explored for the pick action.

(c) a sampled place position for the object and gripper position unreachable by the robot arm.

(d) end state of a valid solution to the place action, after backtracking on the pick action.

Figure 6.6: The robot has to pick the green can and place it on the brown furniture. In (a) the object can be grasped by the robot in a lot of different manners, such as (b). However, the place action is constraining, the surface being too high to place the can while holding it from its top (c). The planner backtracks on the grasp configuration to allow placing on a constrained surface (d).



(a) (b)

Figure 6.7: The alternative sampling allows making decision improving the quality of the plan with respect to human comfort and acceptability. Here, choosing the robot left arm (a) to pick the object before performing the motion plan saves a lot of computational time compared to computing the motion plan for both solutions and choosing the best one. The cost used to select the arm here is related to the distance to the human. Using other costs would certainly prove more pertinent (like also taking into account visibility and legibility)

taken into account only because it is an acting agent we can plan for, and eventually motions where generated taking into account human safety and preferences.

The feature we brought to GTP in this direction are:

— optimizing the motions

— optimizing not only the motion between 2 states, but also the 2 states

— optimizing the right objective at the right time

We will present them in details here.

6.4.1 Optimal Motion Planning

GTP makes use of standard motion-planning algorithm to link two world states, meaning that it needs to find initial and goal states to formulate a valid motion-planning problem [2]. The motion plan is built incrementally from the initial world state, target positions are chosen randomly, in some subspace defined by the task. If the motion plan do not find a solution within the alloted time, a new position is shot, and eventually it backtracks on previous positions or tasks.

The motion is computed with a standard motion-planner, which may be human-aware [Mainprice 2011] when relevant.

6.4.2 Improving Motion by Improving Requests

In this framework, while motion-planner are possibly human-aware, they are constrained to their start and goal configurations. As the positions are chosen

2. there are however motion planners that can take in several objectives, including continuous objectives.

by GTP, which is a planner informed of the world geometry and also of human-related preferences, we are able to chose better positions taking into account human preferences and heuristics on the consequent motion cost. Figure 6.7 illustrates how GTP is able to chose the best arm to use for a pick action near a human, even before having the motion plan computed.

To achieve this, it may not be enough to use the configuration cost used by the motion planner which is a poor heuristic in the general case. In the example of Figure 6.7, we could use a cost which tries to maximize distance between bodies of the robot and bodies of the human, depending on the bodies (reflecting the fact that it is most important to stay away from the human head than its arms, or that the robot body is more threatening than its gripper, for example). Such cost produces rather good motions. Using this configuration cost would lead GTP to select the alternative (b) because the robot right arm is actually further away from the human than in (a), but the subsequent motion has a cost from far worse than with the left arm. This is solved either by choosing a different cost that is suitable for both the motion planning and the heuristic, or using two different functions for the motion planning and for the heuristic. In our example we use for heuristic a modified version of the cost function, that considers only the moving robot parts. (Using that cost for motion planning do not give satisfactory results.)

To summarize, GTP creates good solutions to manipulation actions by choosing which robot part(s) to move, how to grasp the objects, and using human-aware motion-planner for trajectory generation.

6.4.3 Different Objective for Different Task

As stated earlier, we want the tasks to be solved taking into account preferences related to the collaboration with humans. For each elementary step of each action type, the user can configure which cost function to use for the optimal motion planning and the selection of the sampled intermediary world-states.

6.5 Conclusions and Future Work

We have shown in this chapter how the GTP framework, designed for planning motions for collaborative actions involving humans and robots, have been improved to plan better motions depending on the context. We address the problem of choosing the optimization criteria depending on the context in a versatile and powerful framework.

We believe that GTP is a tool that could suit many needs, by having a generic interface for many types of task requests. It could be augmented with new types of tasks, starting with the two dedicated task solvers presented in this dissertation for the Multiple Agent Transport Problem (MHO, Chapter 3) and the Guiding and Route Direction task (SVP planner, Chapter 4).

Chapter 7

Conclusion: Towards Robots Enabled for Seamless Collaborative Task Solving With Humans

Contents

7.1 Introduction

Our work presented in this dissertation focuses on building systems – or system components – that enable robots to respond to situations where they need to *plan* for *collaborative tasks*, using *joint-action* theory to build *shared-plans*.

— planning, deciding what will be done in details, even down to the geometrical level, is needed to solve some complex problems;
— collaborative tasks, where several agents have to accomplish actions that may be dependent on other's actions;
— human-robot joint-action, considering joint-action theory from the human point of view makes robotic systems more efficient and acceptable when working with humans.

The Multiple Handover (MHO) and Shared Visual Perspective (SVP) planners presented in chapters 3 and 4 produce a plan on the geometric level using specific knowledge on the task implemented in the solver. They also rely on information expected to be provided by other components of the system to better consider humans and their preferences.

The Geometric Task Planner (GTP) presented in chapter 6 is an example of how information can be shared between components reasoning at different levels of abstraction and used to improve task resolution and human comfort.

Chapter 5 presents a complete system addressing a collaborative task where it is required to plan geometric actions along with reasoning about information to share.

All of these components and systems are planning for both the human and robot partners of the activity, as actions by any partner may be required to reach the objective.

We produce plans that should be adapted to varying situations. This adaptation is crucial in our approaches as it has to be fast to allow seamless and continuous fitting to the real world from what was planned in the projected world model of the robot. Such adaptation techniques may include reusing information from other levels, and fall in the architecture models we were contributing to.

While working on the projects and topics developed in the previous chapters of this dissertation, we stumbled upon a multitude of connected problems. Some of them got our interest and we started addressing them, because we found no satisfactory or applicable existing solution. They mostly relate to motion planning in a human-robot collaboration context, which requires accounting for uncertainty, dynamics, and manipulating multiple optimization objectives.

7.2 Accounting for Uncertainty by Adapting in Real-Time

When working in a dynamical environment, constantly evolving as we plan and act, we need to permanently monitor and adapt to those changes. When working

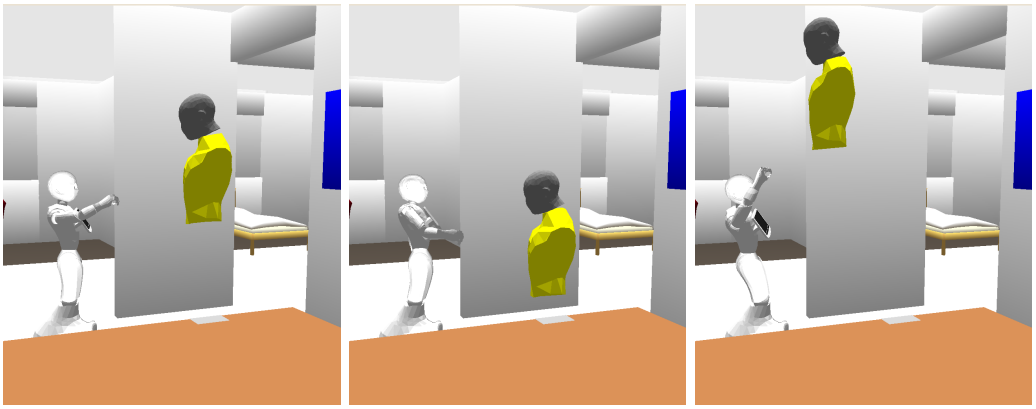
with humans, adapting becomes of prime importance as the environment cannot be predicted with exactitude. We can distinguish three main types of plan adaptation: replanning, anytime planning and local adaptation. Replanning is producing a new plan for the same task with the same tools used to create it at first, but considering the updated world state. Replanning is pertinent when changes in the environment are important. Local adaptation searches for a new optimal solution near the previous one, when the environment did not changed much and a close solution suffices. Anytime planning is meant to continuously improve a solution, even while it is being executed; with some modifications, it can be used for adapting to environment changes.

7.2.1 Continuous Replanning

A now famous approach of adaptation to dynamic environment is the one used by the ROS package `move_base` for robot navigation planning and execution [Marder-Eppstein 2010]. The planner first computes a *global plan* that takes into account only known static environment (building walls, furniture). Then during the execution of the motion, a *local plan* is constantly generated, taking into account robot and environment dynamic. Also, in the first step, they produce only a path, that is a list of position waypoints, while in the second step, they produce a velocity profile to be executed by the motion controllers using the Dynamic Window Approach technique [Fox 1997]. Other authors used this framework with different local planner. [Khambhaita 2017] propose a method for a robot that cooperatively navigates with a human. The local planner is based on *timed elastic band* [Roesmann 2012] to optimize the motion with respect to an objective of motion legibility [Kruse 2014, Dragan 2013a]. This approach of global/local planning relies on fast local planner, as they are run at a constant rate. This rate influences how quickly the system can react to environmental changes. Moreover this approach can be computationally expensive, by always recomputing motions even when the environment is not changing.

Approaches with a single (global) planner also exists, like [van den Berg 2006] where they use a roadmap that is updated on environment changes and shortest path is searched in an anytime fashion. They needed to perform multiple optimization and simplification on the model so that it can be used on-line.

We have initiated work to implement a continuous replanning method based on HATEB planner [Khambhaita 2017] generalized for N degrees of freedom. The work flow is to generate a path from standard motion-planning algorithm, generate a trajectory from this, and them use time-elastic bands to optimize it towards human- and task-oriented objective functions, that can take into account motion dynamics.



(a) (b) (c)

Figure 7.1: Example of adapting an interactive motion, here extending the robot arm towards the human to initiate a handover or a fist-bump. The position we want to retarget is (a). In (b) and (c) the position is adapted to a human at a different position and of different size. This method is well suited to adapt to local changes; if the situation is too different (human on the side, behind an obstacle) a more complete replanning is required.

7.2.2 Local Optimization

Some local optimization tools are specialized or suitable for the local adaptation of motions. [Zucker 2013] presents the Covariant Hamiltonian Optimization for Motion Planning (CHOMP) which can be fast enough to be used in online continuous replanning.

A completely different approach proposed by [Ho 2010] is to use spacial relations to describe a motion. Unlike standard motion representation, the trajectory is not a sequence of robot state or commands, but a sequence of distances between specific points of the robot and the environment. The poses and motions can then be adapted to changes in the environment while still preserving the semantics of the motion. We have been supervising work on the implementation of a similar approach adapted to robotics. It allows both to retarget positions and motions to a different robot, and adapt to environment change. The Figure 7.1 shows examples of interactive position adaptation to different human morphologies. We used descriptors only between the robot and the human, with increased weight on the descriptors attached to the hand, because of the particular importance of it in the action being performed. This choice of descriptors where made by hand, but it send us back to the GTP framework, where such choice could be automated thanks to task knowledge.

7.2.3 Deciding When to Replan, When to Adapt

Most of the approaches related to the continuous adaptation of motions are actually run at a fixed rate, allowing them to capture the changes in the environment

as much as possible. The rate at which the update is done must be chosen according to the dynamic of environment and the computation cost of the algorithm with respect to the computational resource available.

The variety of adaptation technique existing and their differences can clearly be an advantage if they are combined together in a single system. For example the relationship descriptors of [Ho 2010] could be used to adapt to small changes in the environment, while CHOMP or the anytime PRM of [van den Berg 2006] would be used to adapt to bigger changes. Also, sometimes a complete replanning from scratch could be better than adapting existing material.

We started a draft implementation of a tool able to detect and quantify the changes in the environment, using geometric data and semantic information, to decide the most opportune adaptation policy to take, among:

— continue using the current motion;
— locally adapt the current motion;
— replan reusing existing planning material;
— plan a new motion from scratch.

Our approach is to use all the information available to make this choice, so we based ourselves on the GTP framework. So we have access to the task we are performing, the role of the agents, the objects that have specific meaning in the task, and the planned and current state of these. We are then able to decide if the change of position of a human is relevant with respect to the task: a far away human not involved in task can move freely without us needing to replan; on the contrary, if an object moved away and was actually the one we were about to pick, we need a complete new plan.

This led to the design of a *distance* function between two world-states that is also dependent on the semantics of the task being executed. This distance can then be compared to some threshold to consider replanning or adaptation.

This same distance have been used to estimate the difference between a new task request and already solved tasks. The idea is to reuse motion-planning graphs for similar requests. We designed a tool shifts a graph position, checks graph nodes and edges against updated environment collisions and remove them when needed. The graph is then provided to the same motion-planning algorithm that generated it, with the new initial and target positions, to find a path between them using the exploration already represented in the graph. The combination of the task distance and graph reuse tools, integrated in GTP, showed an average computation time similar to the standard GTP. We take this result as promising as we only scratched the surface of this approach.

7.3 Cost Design

In the field of motion-planning for human-robot interaction, most algorithms are based on an objective function to generate a good robot behaviour. While methods for solving the so-called cost-based motion-planning problem abounds, with a wide

range of objective functions capturing as much desired properties, it appears that we are still far from having cost functions that encompass the multitude of objectives that have been said to be pertinent for each specific task. Such objective functions represent the criteria we want to optimize, and in our human-robot interaction context, their design can be complex. Indeed, lots of criteria need to be aggregated into a single objective function, while preserving the meaning we want to put in it (i.e. low cost solutions are still the good ones).

Difficulties arise when we want to find a motion optimal with respect to multiple objectives. The first issue we encounter is how do objectives compare to each other; if they are – and they often are – contradictory, which ones prevails? This problem is known as a Multi-Objective Optimization, and is widely studied as it is of prime importance in many fields like economics and engineering. [Marler 2004] reviews and discuss approaches of multi-objective optimization.

Another difficulty is that most optimal motion-planning algorithm requires that the cost of a motion can be computed on portions of it and then aggregated. This can be inappropriate for some objectives. The main problem of this is that compensation phenomena can occur, that is the motion scores really poorly on some portion, but the rest of the motion is good. The aggregation of the entire motion cost is then acceptable. A way of avoiding compensation is to define constraints. Breaking a constraint at some point of the motion invalidates the whole motion: no compensation is possible here. If all constraints are respected, a cost can then be evaluated. Over allowing to better evaluate motions, this approach can save time by not computing the cost function if a constraint is violated. There is nothing new in proposing this, however most existing motion-planning solutions do not allow to specify multiple constraints of validity (even often the only one considered is the collision).

The computation of a motion cost by its subparts is not suitable either for objectives that are dependent on conditions on the rest of the motion.

Bibliography

[Ahuactzin 1999] J.M. Ahuactzin and K.K. Gupta. *The Kinematic Roadmap: A Motion Planning Based Global Approach for Inverse Kinematics of Redundant Robots*. IEEE Transactions on Robotics and Automation, vol. 15, no. 4, pages 653–669, 1999. (Cited in pages 12 and 130.)

[Alami 1995] R. Alami, F. Robert, F. Ingrand and S. Suzuki. *Multi-Robot Cooperation through Incremental Plan-Merging*. In Proceedings of 1995 IEEE International Conference on Robotics and Automation, volume 3, pages 2573–2579 vol.3, May 1995. (Cited in page 30.)

[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 pages 78 and 155.)

[Alibali 2005] Martha W. Alibali. *Gesture in Spatial Cognition: Expressing, Communicating, and Thinking About Spatial Information*. Spatial Cognition & Computation, vol. 5, no. 4, pages 307–331, December 2005. (Cited in pages 44 and 146.)

[Alili 2009] Samir Alili, Rachid Alami and Vincent Montreuil. *A Task Planner for an Autonomous Social Robot*. In Hajime Asama, Haruhisa Kurokawa, Jun Ota and Kosuke Sekiyama, editors, Distributed Autonomous Robotic Systems 8, pages 335–344. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. (Cited in pages 15 and 133.)

[Allen 2000] Gary L. Allen. *Principles and Practices for Communicating Route Knowledge*. Applied cognitive psychology, vol. 14, no. 4, pages 333–359, 2000. (Cited in pages 40, 42, 44, 145, and 146.)

[Allen 2003] Gary L. Allen. *Gestures Accompanying Verbal Route Directions: Do They Point to a New Avenue for Examining Spatial Representations?* Spatial Cognition & Computation, vol. 3, no. 4, pages 259–268, December 2003. (Cited in pages 43, 44, 145, and 146.)

[Barry 2013] Jennifer Barry, Kaijen Hsiao, Leslie Pack Kaelbling and Tomás Lozano-Pérez. *Manipulation with Multiple Action Types*. In Jaydev P. Desai, Gregory Dudek, Oussama Khatib and Vijay Kumar, editors, Experimental Robotics: The 13th International Symposium on Experimental Robotics, Springer Tracts in Advanced Robotics, pages 531–545. Springer International Publishing, Heidelberg, 2013. (Cited in pages 8, 19, 127, and 136.)

[Bauer 2009] Andrea Maria Bauer, Klaas Klasing, Tingting Xu, Stefan Sosnowski, Georgios Lidoris, Quirin Mühlbauer, Tianguang Zhang, Florian Rohrmüller, Dirk Wollherr, Kolja Kühnlenz and Martin Buss. *The Autonomous City Explorer Project*. In 2009 IEEE International Conference on Robotics and Automation, ICRA 2009, Kobe, Japan, May 12-17, 2009, pages 1595–1596, 2009. (Cited in page 39.)

[Belhassein 2017] Kathleen Belhassein, Aurélie Clodic, Hélène Cochet, Marketta Niemelä, Päivi Heikkila, Hanna Lammi and Antti Tammela. *Human-Human Guidance Study.* Tech-Report hal-01719730, LAAS-CNRS, CLLE, VTT, December 2017. (Cited in pages 43, 47, 146, and 148.)

[Bennewitz 2001] M. Bennewitz, W. Burgard and S. Thrun. *Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems.* volume 1, pages 271–276. IEEE, 2001. (Cited in page 30.)

[Blakemore 2001] S. J. Blakemore and J. Decety. *From the Perception of Action to the Understanding of Intention.* Nature Reviews. Neuroscience, vol. 2, no. 8, pages 561–567, August 2001. (Cited in page 12.)

[Boeuf 2014] Alexandre Boeuf, Juan Cortes, Rachid Alami and Thierry Simeon. *Planning Agile Motions for Quadrotors in Constrained Environments.* In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 218–223. IEEE, September 2014. (Cited in pages 12 and 130.)

[Bohren 2011] Jonathan Bohren, Radu Bogdan Rusu, E. Gil Jones, Eitan Marder-Eppstein, Caroline Pantofaru, Melonee Wise, Lorenz Mosenlechner, Wim Meeussen and Stefan Holzer. *Towards Autonomous Robotic Butlers: Lessons Learned with the PR2.* pages 5568–5575. IEEE, May 2011. (Cited in pages 67 and 153.)

[Bohus 2014] Dan Bohus, Chit W Saw and Eric Horvitz. *Directions Robot: In-the-Wild Experiences and Lessons Learned.* In Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), page 8, 2014. (Cited in pages 44 and 147.)

[Boucher 2012] Jean-David Boucher, Ugo Pattacini, Amelie Lelong, Gerard Bailly, Frederic Elisei, Sascha Fagel, Peter F. Dominey and Jocelyne Ventre-Dominey. *I Reach Faster When I See You Look: Gaze Effects in Human–Human and Human–Robot Face-to-Face Cooperation.* Frontiers in Neurorobotics, vol. 6, 2012. (Cited in page 11.)

[Bratman 1989] Michael E. Bratman. *Intention and Personal Policies.* Philosophical Perspectives, vol. 3, pages 443–469, 1989. (Cited in pages 10 and 129.)

[Brooks 2006] Andrew G. Brooks and Cynthia Breazeal. *Working with Robots and Objects: Revisiting Deictic Reference for Achieving Spatial Common Ground.* page 297. ACM Press, 2006. (Cited in pages 43 and 146.)

[Burgard 1998] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner and Sebastian Thrun. *The Museum Tour-Guide Robot RHINO.* In Autonome Mobile Systeme 1998, 14. Fachgespräch, Karlsruhe, 30. November - 1. Dezember 1998, pages 245–254, 1998. (Cited in page 39.)

[Cakmak 2011a] Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Jodi Forlizzi and Sara Kiesler. *Human Preferences for Robot-Human Hand-over Configurations.* In IEEE International Conference on Intelligent Robots and Systems, pages 1986–1993, 2011. (Cited in pages 18, 19, and 135.)

[Cakmak 2011b] Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Jodi Forlizzi and Sara B. Kiesler. *Human Preferences for Robot-Human Hand-over Configurations.* In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2011, San Francisco, CA, USA, September 25-30, 2011, pages 1986–1993. IEEE, September 2011. (Cited in pages 12 and 130.)

[Cakmak 2011c] Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Sara Kiesler and Jodi Forlizzi. *Using Spatial and Temporal Contrast for Fluent Robot-Human Hand-Overs.* In Proceedings of the 6th International Conference on Human-Robot Interaction, HRI '11, pages 489–496, New York, NY, USA, 2011. ACM. (Cited in pages 18, 19, and 135.)

[Caldiran 2009] Ozan Caldiran, Kadir Haspalamutgil, Abdullah Ok, Can Palaz, Esra Erdem and Volkan Patoglu. *From Discrete Task Plans to Continuous Trajectories.* In Procedings of ICAPS Workshop BTAMP, pages 42–49. Citeseer, 2009. (Cited in pages 8 and 127.)

[Cambon 2009] S. Cambon, R. Alami and F. Gravot. *A Hybrid Approach to Intricate Motion, Manipulation and Task Planning.* The International Journal of Robotics Research, vol. 28, no. 1, pages 104–126, 2009. (Cited in pages 9 and 127.)

[Chan 2012] Wesley P. Chan, Chris A.C. Parker, H.F.Machiel Van der Loos and Elizabeth A. Croft. *Grip Forces and Load Forces in Handovers: Implications for Designing Human-Robot Handover Controllers.* In Human Robot Interaction, pages 9–16, 2012. (Cited in page 18.)

[Chen 2004] S. Y. Chen and Y. F. Li. *Automatic Sensor Placement for Model-Based Robot Vision.* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 34, no. 1, pages 393–408, February 2004. (Cited in pages 43 and 146.)

[Chen 2017] Yingfeng Chen, Feng Wu, Wei Shuai and Xiaoping Chen. *Robots Serve Humans in Public Places— KeJia Robot as a Shopping Assistant.* International Journal of Advanced Robotic Systems, vol. 14, no. 3, May 2017. (Cited in pages 44 and 147.)

[Clark 1983] Herbert H. Clark, Robert Schreuder and Samuel Buttrick. *Common Ground at the Understanding of Demonstrative Reference.* Journal of verbal learning and verbal behavior, vol. 22, no. 2, pages 245–258, 1983. (Cited in pages 42 and 145.)

[Clark 2005] H. H. Clark. *Coordinating with Each Other in a Material World.* Discourse Studies, vol. 7, no. 4, pages 507–525, October 2005. (Cited in pages 43 and 145.)

[Clodic 2006] Aurelie Clodic, Sara Fleury, Rachid Alami, Raja Chatila, Gerard Bailly, Ludovic Brethes, Maxime Cottret, Patrick Danes, Xavier Dollat, Frederic Elisei, Isabelle Ferrane, Matthieu Herrb, Guillaume Infantes, Christian Lemaire, Frederic Lerasle, Jerome Manhes, Patrick Marcoul, Paulo Menezes and Vincent Montreuil. *Rackham: An Interactive Robot-Guide.*

In The 15th IEEE International Symposium on Robot and Human Interactive Communication (ROMAN2006), pages 502–509. IEEE, September 2006. (Cited in page 39.)

[Clodic 2017] Aurélie Clodic, Elisabeth Pacherie, Rachid Alami and Raja Chatila. *Key Elements for Human-Robot Joint Action.* In Raul Hakli and Johanna Seibt, editors, Sociality and Normativity for Robots, pages 159–177. Springer International Publishing, Cham, 2017. (Cited in pages 15 and 132.)

[Cohen 1991] Philip R Cohen and Hector J Levesque. *Teamwork.* Nous, vol. 25, no. 4, pages 487–512, 1991. (Cited in pages 10, 11, 128, and 129.)

[Cohen 2014] Benjamin Cohen, Mike Phillips and Maxim Likhachev. *Planning Single-Arm Manipulations with n-Arm Robots.* In Proceedings of Robotics: Science and Systems, Berkeley, USA, 2014. (Cited in pages 20, 27, 137, 139, and 140.)

[Coltin 2014] B. Coltin and M. Veloso. *Online Pickup and Delivery Planning with Transfers for Mobile Robots.* In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 5786–5791, May 2014. (Cited in pages 20, 136, and 137.)

[Daniel 2003] Marie-Paule Daniel, Ariane Tom, Elsa Manghi and Michel Denis. *Testing the Value of Route Directions Through Navigational Performance.* Spatial Cognition & Computation, vol. 3, no. 4, pages 269–289, December 2003. (Cited in pages 42, 44, 48, 145, and 146.)

[Dautenhahn 2006] Kerstin Dautenhahn, M Walters, Sarah Woods, Kheng Lee Koay, Chrystopher L Nehaniv, A Sisbot, Rachid Alami and Thierry Siméon. *How May I Serve You?: A Robot Companion Approaching a Seated Person in a Helping Context.* In Proc. Conf. Human-Robot Interaction, 2006. (Cited in page 18.)

[Dechter 1991] Rina Dechter, Itay Meiri and Judea Pearl. *Temporal Constraint Networks.* Artificial Intelligence, vol. 49, no. 1, 1991. (Cited in pages 28 and 141.)

[Devaurs 2013] Didier Devaurs, Thierry Siméon and Juan Cortés. *Enhancing the Transition-Based RRT to Deal with Complex Cost Spaces.* In Proc. IEEE ICRA '13, pages pp. 4105–4110, Karlsruhe, Germany, 2013. (Cited in pages 12 and 130.)

[Devin 2016] Sandra Devin and Rachid Alami. *An Implemented Theory of Mind to Improve Human-Robot Shared Plans Execution.* In The Eleventh ACM/IEEE International Conference on Human Robot Interation, HRI 2016, Christchurch, New Zealand, March 7-10, 2016, pages 319–326, 2016. (Cited in pages 15, 40, 62, 133, and 151.)

[Dijkstra 1971] Edsger Wybe Dijkstra. A short introduction to the art of programming, volume 4. Technische Hogeschool Eindhoven Eindhoven, 1971. (Cited in page 50.)

[Dornhege 2009] Christian Dornhege, Patrick Eyerich, Thomas Keller, Sebastian Trug, Michael Brenner and Bernhard Nebel. *Semantic Attachments for Domain-Independent Planning Systems.* ICAPS, 2009. (Cited in page 18.)

[Dornhege 2012] Christian Dornhege, Patrick Eyerich, Thomas Keller, Sebastian Trüg, Michael Brenner and Bernhard Nebel. *Semantic Attachments for Domain-Independent Planning Systems.* In Towards Service Robots for Everyday Environments, Springer Tracts in Advanced Robotics, pages 99–115. Springer, Berlin, Heidelberg, 2012. (Cited in pages 8 and 126.)

[Dragan 2013a] Anca D. Dragan, Kenton C.T. Lee and Siddhartha S. Srinivasa. *Legibility and Predictability of Robot Motion.* In 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 301–308. IEEE, March 2013. (Cited in pages 12, 93, and 130.)

[Dragan 2013b] Anca D. Dragan and Siddhartha S. Srinivasa. *Generating Legible Motion.* Proceedings of Robotics: Science and Systems Conference (RSS 2013), page NP, 2013. (Cited in page 14.)

[Dragan 2015] Anca D Dragan, Shira Bauman, Jodi Forlizzi and Siddhartha S Srinivasa. *Effects of Robot Motion on Human-Robot Collaboration.* In Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction, volume 1, pages 51–58, Portland, Oregon, USA, 2015. ACM. (Cited in pages 12, 13, 131, and 132.)

[Duffner 2013] Stefan Duffner and Jean-Marc Odobez. *Track Creation and Deletion Framework for Long-Term Online Multiface Tracking.* IEEE Transactions on image processing, vol. 22, no. 1, pages 272–285, 2013. (Cited in page 40.)

[Emery 2000] N. J. Emery. *The Eyes Have It: The Neuroethology, Function and Evolution of Social Gaze.* Neuroscience & Biobehavioral Reviews, vol. 24, no. 6, pages 581–604, August 2000. (Cited in page 11.)

[Erdem 2016] Esra Erdem, Volkan Patoglu and Peter Schüller. *A Systematic Analysis of Levels of Integration between High-Level Task Planning and Low-Level Feasibility Checks.* AI Communications, vol. 29, no. 2, pages 319–349, March 2016. (Cited in pages 8 and 126.)

[Evers 2014] Vanessa Evers, Nuno Menezes, Luis Merino, Dariu Gavrila, Fernando Nabais, Maja Pantic, Paulo Alvito and Daphne Karreman. *The Development and Real-World Deployment of FROG, the Fun Robotic Outdoor Guide.* In Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction, HRI '14, pages 100–100, New York, NY, USA, 2014. ACM. (Cited in page 39.)

[Finkel 1974] R.A. Finkel and J.L. Bentley. *Quad Trees a Data Structure for Retrieval on Composite Keys.* Acta Informatica, vol. 4, no. 1, 1974. (Cited in page 36.)

[Fox 1997] D. Fox, W. Burgard and S. Thrun. *The Dynamic Window Approach to Collision Avoidance.* IEEE Robotics & Automation Magazine, vol. 4, no. 1, pages 23–33, March 1997. (Cited in page 93.)

[Garrett 2014] Caelan Reed Garrett, Tomás Lozano-Pérez and Leslie Pack Kaelbling. *Heuristic Search for Task and Motion Planning.* In Proceedings of the Workshop on Planning and Robotics (PlanRob), pages 148–156, 2014. (Cited in pages 8 and 127.)

[Ghallab 2016] Malik Ghallab, Dana S. Nau and Paolo Traverso. Automated planning and acting. Cambridge University Press, 2016. (Cited in pages 8, 78, 126, and 155.)

[Gharbi 2015a] Mamoun Gharbi. *Geometric Reasoning Planning in the Context of Human-Robot Interaction.* Theses, INSA de Toulouse, September 2015. (Cited in pages 8 and 126.)

[Gharbi 2015b] Mamoun Gharbi, Raphael Lallement and Rachid Alami. *Combining Symbolic and Geometric Planning to Synthesize Human-Aware Plans: Toward More Efficient Combined Search.* In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), volume 2015-Decem, pages 6360–6365. IEEE, September 2015. (Cited in pages 15, 78, 133, and 155.)

[Gibson 1977] James J. Gibson. *The Theory of Affordances.* pages pp.67–82, 1977. (Cited in page 12.)

[Grosz 1988] Barbara J. Grosz and Candace L. Sidner. *Plans for Discourse.* Technical Report, BBN LABS INC CAMBRIDGE MA, 1988. (Cited in pages 11 and 129.)

[Grosz 1999] Barbara J. Grosz and Sarit Kraus. *The Evolution of SharedPlans.* In Foundations of Rational Agency, pages 227–262. Springer, 1999. (Cited in pages 11 and 129.)

[Hall 1969] Edward Twitchell Hall. The hidden dimension, volume 1990. Anchor Books New York, 1969. (Cited in pages 12, 13, 130, and 131.)

[Hato 2010] Yasuhiko Hato, Satoru Satake, Takayuki Kanda, Michita Imai and Norihiro Hagita. *Pointing to Space: Modeling of Deictic Interaction Referring to Regions.* In Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction, pages 301–308. IEEE Press, 2010. (Cited in pages 43 and 146.)

[Hauser 2009] Kris Hauser and Jean-Claude Latombe. *Integrating Task and PRM Motion Planning: Dealing with Many Infeasible Motion Planning Queries.* International Conference on Automated Planning and Scheduling, no. 1, 2009. (Cited in pages 9 and 127.)

[He 2015] Wuwei He and Daniel Sidobre. *Improving Human-Robot Object Exchange by Online Force Classification.* J. Hum.-Robot Interact., vol. 4, no. 1, pages 75–94, July 2015. (Cited in pages 19 and 135.)

[Ho 2010] Edmond S L Ho, Taku Komura and Chiew-Lan Tai. *Spatial Relationship Preserving Character Motion Adaptation.* In ACM Transactions on Graphics (TOG), volume 29, page 33. ACM, 2010. (Cited in pages 94, 95, and 159.)

[Holladay 2014] Rachel M. Holladay, Anca D. Dragan and Siddhartha S. Srinivasa. *Legible Robot Pointing.* In Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium On, pages 217–223. IEEE, August 2014. (Cited in pages 43 and 145.)

[Huang 2015] Chien-Ming Huang, Maya Cakmak and Bilge Mutlu. *Adaptive Coordination Strategies for Human-Robot Handovers.* In Lydia E. Kavraki, David Hsu and Jonas Buchli, editors, Robotics: Science and Systems XI, Sapienza University of Rome, Rome, Italy, July 13-17, 2015, 2015. (Cited in pages 12 and 130.)

[Jaillet 2008] L. Jaillet, J. Cortes and T. Simeon. *Transition-Based RRT for Path Planning in Continuous Cost Spaces.* pages 2145–2150. IEEE, September 2008. (Cited in pages 12 and 130.)

[Jaillet 2010] Léonard Jaillet, Juan Cortés and T Siméon. *Sampling-Based Path Planning on Configuration-Space Costmaps.* IEEE Transactions on Robotics, vol. 26, no. 4, pages 635–646, August 2010. (Cited in page 86.)

[Kaelbling 2011] Leslie Pack Kaelbling and Tomás Lozano-Pérez. *Hierarchical Task and Motion Planning in the Now.* In Proc. Conf. IEEE ICRA, 2011. (Cited in page 18.)

[Kanda 2010] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro and N. Hagita. *A Communication Robot in a Shopping Mall.* IEEE Transactions on Robotics, vol. 26, no. 5, pages 897–913, October 2010. (Cited in pages 39, 44, and 147.)

[Karlsson 2012] Lars Karlsson, Julien Bidot, Fabien Lagriffoul, Alessandro Saffiotti, Ulrich Hillenbrand and Florian Schmidt. *Combining Task and Path Planning for a Humanoid Two-Arm Robotic System.* TAMPRA, 2012. (Cited in page 18.)

[Kelly 2004] Jonathan W. Kelly, Andrew C. Beall and Jack M. Loomis. *Perception of Shared Visual Space: Establishing Common Ground in Real and Virtual Environments.* Presence: Teleoperators & Virtual Environments, vol. 13, no. 4, pages 442–450, 2004. (Cited in pages 43 and 146.)

[Khambhaita 2017] Harmish Khambhaita and Rachid Alami. *Viewing Robot Navigation in Human Environment as a Cooperative Activity.* In International Symposium on Robotics Research (ISSR 2017), page 18p., Puerto Varas, Chile, 2017. (Cited in pages 15, 30, 40, 69, 70, 93, 133, and 153.)

[Koay 2007] K. L. Koay, E.A. Sisbot, D.S. Syrdal, M.L. Walters, K. Dautenhahn and R. Alami. *Exploratory Study of a Robot Approaching a Person in the Context of Handing over an Object.* In AAAI Spring Symposium: Multidisciplinary Collaboration for Socially Assistive Robotics, pages 18–24, 2007. (Cited in pages 14 and 131.)

[Koga 1994] Yoshihito Koga, Koichi Kondo, James Kuffner and Jean-Claude Latombe. *Planning Motions with Intentions.* In Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIG-

GRAPH '94, pages 395–408, New York, NY, USA, 1994. ACM. (Cited in pages 12 and 130.)

[Kruse 2012] Thibault Kruse, Patrizia Basili, Stefan Glasauer and Alexandra Kirsch. *Legible Robot Navigation in the Proximity of Moving Humans.* In IEEE Workshop on Advanced Robotics and Its Social Impacts, Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on, pages 83 – 88, Munich, Germany, May 2012. (Cited in pages 14 and 131.)

[Kruse 2013] Thibault Kruse, Amit Kumar Pandey, Rachid Alami and Alexandra Kirsch. *Human-Aware Robot Navigation: A Survey.* Robotics and Autonomous Systems, vol. 61, no. 12, pages 1726–1743, December 2013. (Cited in pages 12 and 130.)

[Kruse 2014] Thibault Kruse, Alexandra Kirsch, Harmish Khambhaita and Rachid Alami. *Evaluating Directional Cost Models in Navigation.* In Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction, HRI '14, pages 350–357, New York, NY, USA, 2014. ACM. (Cited in page 93.)

[Kuipers 2000] Benjamin Kuipers. *The Spatial Semantic Hierarchy.* Artificial Intelligence, vol. 119, no. 1, pages 191–233, May 2000. (Cited in page 64.)

[Kümmerle 2013] Rainer Kümmerle, Michael Ruhnke, Bastian Steder, Cyrill Stachniss and Wolfram Burgard. *A Navigation System for Robots Operating in Crowded Urban Environments.* In 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013, pages 3225–3232, 2013. (Cited in page 39.)

[Kupcsik 2016] Andras Kupcsik, David Hsu and Wee Sun Lee. *Learning Dynamic Robot-to-Human Object Handover from Human Feedback.* CoRR, vol. abs/1603.06390, 2016. (Cited in pages 12 and 130.)

[Lagriffoul 2013] Fabien Lagriffoul, Lars Karlsson, Julien Bidot and Alessandro Saffiotti. *Combining Task and Motion Planning Is Not Always a Good Idea.* In RSS Workshop on Combined Robot Motion Planning and AI Planning for Practical Applications, 2013. (Cited in page 19.)

[Lagriffoul 2014] Fabien Lagriffoul, Dimitar Dimitrov, Julien Bidot, Alessandro Saffiotti and Lars Karlsson. *Efficiently Combining Task and Motion Planning Using Geometric Constraints.* The International Journal of Robotics Research, vol. 33, no. 14, pages 1726–1747, December 2014. (Cited in pages 8 and 126.)

[Lallée 2013] Stéphane Lallée, Katharina Hamann, Jasmin Steinwender, Felix Warneken, Uriel Martienz, Hector Barron-Gonzales, Ugo Pattacini, Ilaria Gori, Maxime Petit, Giorgio Metta, Paul Verschure and Peter Ford Dominey. *Cooperative Human Robot Interaction Systems: IV. Communication of Shared Plans with Naïve Humans Using Gaze and Speech.* In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 129–136, Tokyo, November 2013. IEEE. (Cited in page 11.)

[Lallement 2014] Raphaël Lallement, Lavindra De Silva and Rachid Alami. *HATP: An HTN Planner for Robotics*. In 2nd ICAPS Workshop on Planning and Robotics, Portsmouth, United States, June 2014. (Cited in pages 15 and 133.)

[Lallement 2016] Raphaël Lallement. *Symbolic and Geometric Planning for Teams of Robots and Humans*. Theses, INSA de Toulouse, September 2016. (Cited in pages 15 and 133.)

[LaValle 2006] Steven M LaValle. Planning algorithms. Cambridge university press, 2006. (Cited in pages 8, 12, 19, 23, 79, 126, 130, 136, and 155.)

[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 pages 15, 62, 133, and 151.)

[Lemaignan 2018] Severin Lemaignan, Yoan Sallami, Christopher Wallbridge, Aurelie Clodic, Tony Belpaeme and Rachid Alami. *UNDERWORLDS: Cascading Situation Assessment for Robots*, 2018. (Cited in pages 65 and 152.)

[Li 2015] Yanbo Li, Zakary Littlefield and Kostas E. Bekris. *Sparse Methods for Efficient Asymptotically Optimal Kinodynamic Planning*. In Algorithmic Foundations of Robotics XI, Springer Tracts in Advanced Robotics, pages 263–282. Springer, Cham, 2015. (Cited in pages 12 and 130.)

[Likhachev 2003] Maxim Likhachev, Geoff Gordon and Sebastian Thrun. *ARA*: Anytime A* with Provable Bounds on Sub-Optimality*. In Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS'03, pages 767–774, Cambridge, MA, USA, 2003. MIT Press. (Cited in page 27.)

[Lozano-Perez 1987] T. Lozano-Perez, J. Jones, E. Mazer, P. O'Donnell, W. Grimson, P. Tournassoud and A. Lanusse. *Handey: A Robot System That Recognizes, Plans, and Manipulates*. In IEEE International Conference on Robotics and Automation, volume 4, pages 843–849. Institute of Electrical and Electronics Engineers, 1987. (Cited in pages 78 and 155.)

[Lozano-Pérez 2014] T. Lozano-Pérez and L. P. Kaelbling. *A Constraint-Based Method for Solving Sequential Manipulation Planning Problems*. In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3684–3691, September 2014. (Cited in pages 8 and 127.)

[Mainprice 2010] Jim Mainprice, EA Sisbot, Thierry Siméon and Rachid Alami. *Planning Safe and Legible Hand-over Motions for Human-Robot Interaction*. In IARP Workshop on Technical Challenges for Dependable Robots in Human Environments, volume 2, page 7, 2010. (Cited in page 18.)

[Mainprice 2011] Jim Mainprice, E. Akin Sisbot, Léonard Jaillet, Juan Cortés, Rachid Alami, Thierry Siméon, J Cortes, Rachid Alami and T Simeon. *Planning Human-Aware Motions Using a Sampling-Based Costmap Planner*. In

Robotics and Automation (ICRA), 2011 IEEE International Conference On, pages 5012–5017, Shanghai, May 2011. IEEE. (Cited in pages 84, 86, 89, and 157.)

[Mainprice 2012] J. Mainprice, M. Gharbi, T. Siméon and R. Alami. *Sharing Effort in Planning Human-Robot Handover Tasks.* In 2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication, pages 764–770, September 2012. (Cited in pages 15, 16, 18, 21, 22, 25, 133, 134, 138, and 139.)

[Marder-Eppstein 2010] Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey and Kurt Konolige. *The Office Marathon: Robust Navigation in an Indoor Office Environment.* In International Conference on Robotics and Automation, 2010. (Cited in page 93.)

[Marin-Urias 2009] L. F. Marin-Urias, E. A. Sisbot, A. K. Pandey, R. Tadakuma and R. Alami. *Towards Shared Attention through Geometric Reasoning for Human Robot Interaction.* In 2009 9th IEEE-RAS International Conference on Humanoid Robots, pages 331–336. IEEE, December 2009. (Cited in pages 15, 43, 45, 132, and 146.)

[Marler 2004] R.T. Marler and J.S. Arora. *Survey of Multi-Objective Optimization Methods for Engineering.* Structural and Multidisciplinary Optimization, vol. 26, no. 6, pages 369–395, April 2004. (Cited in pages 96 and 160.)

[Matsumoto 2012] Takahiro Matsumoto, Satoru Satake, Takayuki Kanda, Michita Imai and Norihiro Hagita. *Do You Remember That Shop?: Computational Model of Spatial Memory for Shopping Companion Robots.* In HRI '12 Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction, pages 447–454. ACM Press, 2012. (Cited in pages 44 and 147.)

[May 2015] A. D. May, C. Dondrup and M. Hanheide. *Show Me Your Moves! Conveying Navigation Intention of a Mobile Robot to Humans.* In 2015 European Conference on Mobile Robots (ECMR), pages 1–6, September 2015. (Cited in pages 70 and 153.)

[Milliez 2014] G. Milliez, M. Warnier, A. Clodic and R. Alami. *A Framework for Endowing an Interactive Robot with Reasoning Capabilities about Perspective-Taking and Belief Management.* In The 23rd IEEE International Symposium on Robot and Human Interactive Communication, pages 1103–1109, August 2014. (Cited in page 65.)

[Moon 2014] AJung Moon, Daniel M. Troniak, Brian Gleeson, Matthew K.X.J. Pan, Minhua Zheng, Benjamin A. Blumer, Karon MacLean and Elizabeth A. Croft. *Meet Me Where I'm Gazing: How Shared Attention Gaze Affects Human-Robot Handover Timing.* In Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction, HRI '14, pages 334–341, New York, NY, USA, 2014. ACM. (Cited in pages 19 and 135.)

[Morales Saiki 2011] Luis Yoichi Morales Saiki, Satoru Satake, Takayuki Kanda and Norihiro Hagita. *Modeling Environments from a Route Perspective.* page 441. ACM Press, 2011. (Cited in pages 44 and 147.)

[Morales 2015] Yoichi Morales, Satoru Satake, Takayuki Kanda and Norihiro Hagita. *Building a Model of the Environment from a Route Perspective for Human–Robot Interaction.* International Journal of Social Robotics, vol. 7, no. 2, pages 165–181, April 2015. (Cited in pages 45, 62, and 147.)

[Morris 2000] Paul Morris and Nicola Muscettola. *Execution of Temporal Plans with Uncertainty.* In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI/IAAI), 2000. (Cited in page 36.)

[Mutlu 2006] B. Mutlu, J. Forlizzi and J. Hodgins. *A Storytelling Robot: Modeling and Evaluation of Human-like Gaze Behavior.* In 2006 6th IEEE-RAS International Conference on Humanoid Robots, pages 518–523, December 2006. (Cited in pages 70 and 153.)

[Mwangi 2018] Eunice Mwangi, Emilia I. Barakova, Marta Díaz-Boladeras, Andreu Català Mallofré and Matthias Rauterberg. *Directing Attention Through Gaze Hints Improves Task Solving in Human–Humanoid Interaction.* International Journal of Social Robotics, vol. 10, no. 3, pages 343–355, June 2018. (Cited in pages 15 and 132.)

[Nedunuri 2014] Srinivas Nedunuri, Sailesh Prabhu, Mark Moll, Swarat Chaudhuri and Lydia E. Kavraki. *SMT-Based Synthesis of Integrated Task and Motion Plans from Plan Outlines.* In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 655–662, Hong Kong, China, May 2014. IEEE. (Cited in pages 8 and 127.)

[Nothegger 2004] Clemens Nothegger, Stephan Winter and Martin Raubal. *Selection of Salient Features for Route Directions.* Spatial Cognition & Computation, vol. 4, no. 2, pages 113–136, 2004. (Cited in pages 40, 42, and 145.)

[Okuno 2009] Yusuke Okuno, Takayuki Kanda, Michita Imai, Hiroshi Ishiguro and Norihiro Hagita. *Providing Route Directions: Design of Robot's Utterance, Gesture, and Timing.* In Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference On, pages 53–60. IEEE, 2009. (Cited in pages 44 and 147.)

[Pan 2017] Matthew KXJ Pan, Vidar Skjervøy, Wesley P Chan, Masayuki Inaba and Elizabeth A Croft. *Automated Detection of Handovers Using Kinematic Features.* The International Journal of Robotics Research, vol. 36, no. 5-7, pages 721–738, June 2017. (Cited in pages 19 and 135.)

[Pan 2018] Matthew Keith Xi-Jie Pan. *Towards Enabling Human-Robot Handovers : Exploring Nonverbal Cues for Fluent Human-Robot Handovers.* PhD thesis, University of British Columbia, 2018. (Cited in pages 19 and 135.)

[Papaioannou 2017a] Ioannis Papaioannou, Amanda Cercas Curry, Jose L Part, Igor Shalyminov, Xinnuo Xu, Yanchao Yu, Ondřej Dušek, Verena Rieser

and Oliver Lemon. *Alana: Social Dialogue Using an Ensemble Model and a Ranker Trained on User Feedback.* Alexa Prize Proceedings, page 10, 2017. (Cited in page 73.)

[Papaioannou 2017b] Ioannis Papaioannou, Christian Dondrup, Jekaterina Novikova and Oliver Lemon. *Hybrid Chat and Task Dialogue for More Engaging HRI Using Reinforcement Learning.* pages 593–598. IEEE, August 2017. (Cited in page 40.)

[Pierno 2006] Andrea C. Pierno, Cristina Becchio, Matthew B. Wall, Andrew T. Smith, Luca Turella and Umberto Castiello. *When Gaze Turns into Grasp.* Journal of Cognitive Neuroscience, vol. 18, no. 12, pages 2130–2137, December 2006. (Cited in page 11.)

[Plaku 2010] E. Plaku and G. D. Hager. *Sampling-Based Motion and Symbolic Action Planning with Geometric and Differential Constraints.* In 2010 IEEE International Conference on Robotics and Automation, pages 5002–5008, May 2010. (Cited in pages 8 and 127.)

[Quispe 2014] Ana Huaman Quispe, Heni Ben Amor and Mike Stilman. *Handover Planning for Every Occasion.* In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 431–436. IEEE, November 2014. (Cited in pages 19 and 135.)

[Rios-Martinez 2015] Jorge Rios-Martinez, Anne Spalanzani and Christian Laugier. *From Proxemics Theory to Socially-Aware Navigation: A Survey.* I. J. Social Robotics, vol. 7, no. 2, pages 137–153, 2015. (Cited in pages 12 and 130.)

[Roesmann 2012] C. Roesmann, W. Feiten, T. Woesch, F. Hoffmann and T. Bertram. *Trajectory Modification Considering Dynamic Constraints of Autonomous Robots.* In ROBOTIK 2012; 7th German Conference on Robotics, pages 1–6, May 2012. (Cited in page 93.)

[Satake 2015a] S. Satake, K. Hayashi, K. Nakatani and T. Kanda. *Field Trial of an Information-Providing Robot in a Shopping Mall.* In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1832–1839, September 2015. (Cited in pages 40, 44, 62, and 147.)

[Satake 2015b] Satoru Satake, Keita Nakatani, Kotaro Hayashi, Takyuki Kanda and Michita Imai. *What Should We Know to Develop an Information Robot?* PeerJ Computer Science, vol. 1, page e8, June 2015. (Cited in pages 44 and 147.)

[Sauppé 2014] Allison Sauppé and Bilge Mutlu. *Robot Deictics: How Gesture and Context Shape Referential Communication.* pages 342–349. ACM Press, 2014. (Cited in pages 43 and 145.)

[Savelsbergh 1995] Martin WP Savelsbergh and Marc Sol. *The General Pickup and Delivery Problem.* Transportation science, vol. 29, no. 1, 1995. (Cited in pages 20 and 136.)

[Sebanz 2006] 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 127.)

[Siegwart 2003] Roland Siegwart, Kai Oliver Arras, Samir Bouabdallah, Daniel Burnier, Gilles Froidevaux, Xavier Greppin, Björn Jensen, Antoine Lorotte, Laetitia Mayor, Mathieu Meisser, Roland Philippsen, Ralph Piguet, Guy Ramel, Gregoire Terrien and Nicola Tomatis. *Robox at Expo.02: A Large-Scale Installation of Personal Robots.* Robotics and Autonomous Systems, vol. 42, no. 3-4, pages 203–222, 2003. (Cited in page 39.)

[Simeon 2002] T. Simeon, S. Leroy and J. Lauumond. *Path Coordination for Multiple Mobile Robots: A Resolution-Complete Algorithm.* IEEE Transactions on Robotics and Automation, vol. 18, no. 1, pages 42–49, February 2002. (Cited in page 30.)

[Simeon 2004] T. Simeon. *Manipulation Planning with Probabilistic Roadmaps.* The International Journal of Robotics Research, vol. 23, no. 7-8, 2004. (Cited in pages 19 and 136.)

[Sisbot 2007a] E. A. Sisbot, L. F. Marin and R. Alami. *Spatial Reasoning for Human Robot Interaction.* In 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2281–2287, October 2007. (Cited in page 23.)

[Sisbot 2007b] E.A. Sisbot, L.F. Marin-Urias, R. Alami and T. Simeon. *A Human Aware Mobile Robot Motion Planner.* IEEE Transactions on Robotics, vol. 23, no. 5, pages 874–883, October 2007. (Cited in pages 12, 14, 130, and 132.)

[Sisbot 2011] E. A. Sisbot, R. Ros and R. Alami. *Situation Assessment for Human-Robot Interactive Object Manipulation.* In 2011 RO-MAN, pages 15–20, July 2011. (Cited in page 65.)

[Sisbot 2012] E. A. Sisbot and R. Alami. *A Human-Aware Manipulation Planner.* IEEE Transactions on Robotics, vol. 28, no. 5, pages 1045–1057, October 2012. (Cited in pages 12 and 130.)

[Srivastava 2013] Siddharth Srivastava, Lorenzo Riano, Stuart Russell and Pieter Abbeel. *Using Classical Planners for Tasks with Continuous Operators in Robotics.* In International Conference on Automated Planning and Scheduling, volume 3, 2013. (Cited in pages 8 and 127.)

[Stilman 2007] Mike Stilman. *Task Constrained Motion Planning in Robot Joint Space.* October 2007. (Cited in pages 12 and 130.)

[Strabala 2013] Kyle Wayne Strabala, Min Kyung Lee, Anca Diana Dragan, Jodi Lee Forlizzi, Siddhartha Srinivasa, Maya Cakmak and Vincenzo Micelli. *Towards Seamless Human-Robot Handovers.* Journal of Human-Robot Interaction, vol. 2, no. 1, 2013. (Cited in page 18.)

[Şucan 2012] I. A. Şucan and L. E. Kavraki. *Accounting for Uncertainty in Simultaneous Task and Motion Planning Using Task Motion Multigraphs.* In

2012 IEEE International Conference on Robotics and Automation, pages 4822–4828, May 2012. (Cited in pages 8 and 126.)

[Thrun 1999] S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte and D. Schulz. *MINERVA: A Second-Generation Museum Tour-Guide Robot.* In Proceedings of the IEEE International Conference on Robotics and Automation, volume 3, pages 1999–2005, 1999. (Cited in page 39.)

[Tomasello 2005] Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne and Henrike Moll. *Understanding and Sharing Intentions: The Origins of Cultural Cognition.* Behavioral and brain sciences, vol. 28, no. 05, pages 675–691, 2005. (Cited in pages 10, 11, 128, and 129.)

[Triebel 2016] Rudolph Triebel, Kai Arras, Rachid Alami, Lucas Beyer, Stefan Breuers, Raja Chatila, Mohamed Chetouani, Daniel Cremers, Vanessa Evers, Michelangelo Fiore, Hayley Hung, Omar A. Islas Ramírez, Michiel Joosse, Harmish Khambhaita, Tomasz Kucner, Bastian Leibe, Achim J. Lilienthal, Timm Linder, Manja Lohse, Martin Magnusson, Billy Okal, Luigi Palmieri, Umer Rafi, Marieke van Rooij and Lu Zhang. *SPENCER: A Socially Aware Service Robot for Passenger Guidance and Help in Busy Airports.* In Wettergreen D., Barfoot T. (Eds) Field and Service Robotics, volume 113 of *Springer Tracts in Advanced Robotics*, pages 607–622. Springer, Cham, 2016. (Cited in page 39.)

[van den Berg 2006] J. van den Berg, D. Ferguson and J. Kuffner. *Anytime Path Planning and Replanning in Dynamic Environments.* In Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pages 2366–2371, May 2006. (Cited in pages 93 and 95.)

[Vesper 2014] Cordula Vesper and Michael J. Richardson. *Strategic Communication and Behavioral Coupling in Asymmetric Joint Action.* Experimental Brain Research, vol. 232, no. 9, pages 2945–2956, September 2014. (Cited in page 12.)

[Vesper 2017] Cordula Vesper, Ekaterina Abramova, Judith Bütepage, Francesca Ciardo, Benjamin Crossey, Alfred Effenberg, Dayana Hristova, April Karlinsky, Luke McEllin, Sari R. R. Nijssen, Laura Schmitz and Basil Wahn. *Joint Action: Mental Representations, Shared Information and General Mechanisms for Coordinating with Others.* Frontiers in Psychology, vol. 07, January 2017. (Cited in pages 11, 12, and 130.)

[Wong 2010] Nelson Wong and Carl Gutwin. *Where Are You Pointing?: The Accuracy of Deictic Pointing in CVEs.* In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 1029–1038. ACM Press, 2010. (Cited in pages 43 and 146.)

[Yao 2005] Zhenwang Yao and K. Gupta. *Path Planning with General End-Effector Constraints: Using Task Space to Guide Configuration Space Search.* In

2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1875–1880, August 2005. (Cited in pages 12 and 130.)

[Yeap 1988] Wai K. Yeap. *Towards a Computational Theory of Cognitive Maps.* Artificial Intelligence, vol. 34, no. 3, pages 297–360, April 1988. (Cited in page 62.)

[Yeap 1999] Wai K. Yeap and Margaret E. Jefferies. *Computing a Representation of the Local Environment.* Artificial Intelligence, vol. 107, no. 2, pages 265–301, 1999. (Cited in page 62.)

[Zaraki 2014] A. Zaraki, D. Mazzei, M. Giuliani and D. De Rossi. *Designing and Evaluating a Social Gaze-Control System for a Humanoid Robot.* IEEE Transactions on Human-Machine Systems, vol. 44, no. 2, pages 157–168, April 2014. (Cited in pages 70 and 153.)

[Zhou 2017] Allan Zhou, Dylan Hadfield-Menell, Anusha Nagabandi and Anca D. Dragan. *Expressive Robot Motion Timing.* In Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, HRI '17, pages 22–31, New York, NY, USA, 2017. ACM. (Cited in pages 12 and 130.)

[Zickler 2009] Stefan Zickler and M Veloso. *Efficient Physics-Based Planning: Sampling Search via Non-Deterministic Tactics and Skills.* Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, pages 27–34, 2009. (Cited in pages 8 and 127.)

[Zucker 2013] M. Zucker, N. Ratliff, a. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. a. Bagnell and S. S. Srinivasa. *CHOMP: Covariant Hamiltonian Optimization for Motion Planning.* The International Journal of Robotics Research, vol. 32, no. 9-10, pages 1164–1193, August 2013. (Cited in pages 12, 94, and 130.)

Appendix A

Multiple Handover Task Solver Algorithm

The Multiple Handover Task solver is presented in the Chapter 3. We provide here more technical details on its implementation.

To avoid errors, I prefered to provide portions of the code as it is integrated in the system that ran the live demonstration. It is written in C++. The most important parts are presented here. The complete code is available under open-source license as part of *move3d*[1].

```
1  void run() {
2    computeGrid();
3    _grid->searchAllFrontiers();
4    if (_check_frontier_before) {
5      checkAllMetaFrontiersWithObject();
6    }
7    computeAgentsGraph();
8    searchSolutionLWAstar();
9  }
10
11 void searchSolutionLWAstar() {
12   double time_limit{};
13   Robot *first = getFirstAgent(), *target = getTargetAgent();
14   Node *start, *goal;
15
16   computeLongestSolutionDistance();
17   start = new Node(getOrigin(first), first);
18   goal = new Node(getTarget(target), target);
19   for (Agent a : _agents) {
20     if (useCost(a) <= 1 && haveFinalPosition(a)) {
21       _grid->computeAltDistancesForAgent(getAgentTargetCell(a), a);
22     }
23   }
24   deque<Node *> path = computeLWAstar(start, goal);
25   if (path) { // a solution is found to LWA*
26     // we will build the complete solution, with all motions
27     MultiHandOverSolution solution;
28     solution.object = _object;
29     Node *prev = nullptr;
30     Path currentPath; // a sequence of 2D cells (positions)
31     solution.addNextAgent(first);
32     for (Node *n : path) {
```

1. https://www.openrobots.org/wiki/move3d

```
33        if (solution.currentAgent() == n->agent()) {
34          // incrementally build the navigation path
35          currentPath.push_back(n->cell());
36        } else {
37          // the holding agent changes (handover)
38          solution.addNavigation(currentPath); // previous agent path
39          currentPath.clear();
40          solution.addNextAgent(n->agent()); // new holding agent
41          // add the trajectory for the new agent to come to the HO cell from
42          // its initial position
43          solution.addNavigation(getPrecomputedPath(
44              n->agent(), n->agent()->origin(), n->cell()));
45          // add the handover
46          solution.addHandOverAction(getHandoverBetween(
47              prev, n)); // the handover have already been computed
48        }
49        prev = n;
50      }
51
52      // compute and add to the solution the navigations for agents to go
53      // back to their target position, if any:
54      addFinalNavigations(solution);
55      // compute the solution cost from all the navigation and handover
56      // actions:
57      computeSolutionCost(solution);
58
59      // post-process to solve collisions between agents
60      if (usePostProcess()) {
61        postProcessSolution(solution); // not documented here
62      }
63    }
64  }
65
66  void computeLongestSolutionDistance() {
67    double dist(0), time(0), use_cost(0);
68    // traj is at most dmax and the last if it goes back to initial is dmax
69    // (for each agent)
70    int max_paths_per_agent = 2;
71    // if go back to init, max_paths_per_agent=3
72    for (uint i = 0; i < getAgents().size(); ++i) {
73      if (getAgentUseCost(i) < numeric_limits<double>::infinity()) {
74        // 2*dmax is an upper bound for the real Dmax from any point
75        double tmax, cmax;
76        double dmax = max_paths_per_agent * 2 *
77                      _grid->getAgentStructAt(i)->getMaxDist();
78        tmax = dmax / getAgentSpeed(i);
79        cmax = tmax * getAgentUseCost(i);
80        dist += dmax;
81        time += tmax;
82        use_cost += cmax;
83      }
84    }
85    _max_dist = dist;
86    _max_time = time;
```

```
87    _max_use_cost = use_cost;
88  }
89
90  void computeGrid(void) {
91    if (getGridCellSize() != _grid->getCellSize().x() ||
92        getGridCellSize() != _grid->getCellSize().y()) {
93      _computeGrid(); // initializes the grid with corresponding dimensions
94    }
95    _grid->computeAccessAndDistCombined();
96  }
97
98  void checkAllMetaFrontiersWithObject() {
99    map<MultiAgentGrid::MetaAgentInGrid::s_ptr, uint> meta_agents;
100   for (uint a1 = 0; a1 < _agents.size(); ++a1) {
101     meta_agents[_grid->getAgentStructAt(a1)->getMetaAgent()] = a1;
102   }
103   for (map<MultiAgentGrid::MetaAgentInGrid::s_ptr, uint>::iterator a1 =
104            meta_agents.begin();
105        a1 != meta_agents.end(); ++a1) {
106     // find frontiers with all other meta agents and itself
107     for (map<MultiAgentGrid::MetaAgentInGrid::s_ptr, uint>::iterator a2 =
108              a1;
109          a2 != meta_agents.end(); ++a2) {
110       checkMetaFrontiersWithObject(a1->second, a2->second);
111     }
112   }
113 }
114
115 bool checkMetaFrontiersWithObject(uint a1, uint a2) {
116   bool ret = false;
117   MultiAgentGrid::MetaAgentInGrid::s_ptr
118       m1 = _grid->getAgentStructAt(a1)->getMetaAgent(),
119       m2 = _grid->getAgentStructAt(a2)->getMetaAgent();
120
121   vector<MultiAgentCell *> &fcells = m1->frontiers(m2);
122   for (uint i_cell = 0; i_cell < fcells.size(); ++i_cell) {
123     MultiAgentCell *c = fcells[i_cell];
124     vector<MetaFrontierPtr> &mfrts = c->getMetaFrontier(a1, m2);
125     for (uint i_frts = 0; i_frts < mfrts.size(); ++i_frts) {
126       if (checkMetaFrontierWithObject(a1, a2, mfrts[i_frts])) {
127         // the frontier is ok
128         m1->frontiers_ok(m2).push_back(
129             static_cast<MetaFrontierPtr>(mfrts[i_frts])->cellOf(m1));
130         m2->frontiers_ok(m1).push_back(
131             static_cast<MetaFrontierPtr>(mfrts[i_frts])->cellOf(m2));
132         ret = true;
133       }
134     }
135   }
136   return ret;
137 }
138
139 bool checkMetaFrontierWithObject(uint a1, uint a2, MetaFrontierPtr mf) {
140   bool st = false;
```

```
141   // FrontierPtr f=mf->getCellA()->createRealFrontierFor(a1,a2,mf);
142   FrontierPtr f = mf->cellOf(_grid->getAgentStructAt(a1)->getMetaAgent())
143                         ->createRealFrontierFor(a1, a2, mf);
144   st = _tool_set->frontierTester->test(f, FrontierCells::FCS_OBJECT);
145   // st=checkFrontierWithObject(*f);
146   if (st) {
147     mf->setCheckedOk(true);
148     mf->setObjectPosition(f->object_position);
149   }
150   return st;
151 }
152
153 double computeSolutionCost(MultiHandOverSolution *solution) {
154   double time_nav_cost(0), time_ho_cost(0), distance(0), time_nav(0),
155       time_ho(0), obj_time(0), hri_cost(0), cost(0);
156   // solution->avail_delay.assign(solution->agents.size(),0);
157   for (uint i = 0; i < solution->actions.size(); ++i) {
158     assert(solution->actions[i].getSolution() == solution);
159     MultiHandOverSolution::Action &act = solution->actions[i];
160     // nav start date of the 1st agent: when it starts to move, its
161     // immediately after the end of the previous HO [or 0 if first action]
162     act.setStartDate(0, obj_time);
163     act.setNavTime(0, act.dist(0) / act.agent(0)->speed);
164     act.setNavTime(1, act.dist(1) / act.agent(1)->speed);
165     obj_time +=
166         act.dist(0) / act.agent(0)->speed; // tps parcourru par le holder
167     double waiting = act.dist(1) / act.agent(1)->speed -
168                      obj_time; // time the holder waits for the other agent
169     if (waiting > 0) {         // the holder waits
170       obj_time +=
171           waiting; // the object is then delayed of that waiting time
172       if (i == 0) {
173         act.setStartDate(
174             0, act.startDate(0) +
175                    waiting); // delay the start of the first agent
176         waiting = 0;
177         // the first agent can wait at its starting position
178       }
179     } else {
180       // nav start time of the second agent: when it should start moving
181       // to be at the HO point on time
182       act.setStartDate(1, -waiting);
183       waiting = 0;
184     }
185     assert(abs(act.startDate(1) + act.navTime(1) - obj_time) < 0.00001);
186     assert(abs(act.startDate(0) + act.navTime(0) - obj_time + waiting) <
187            0.00001);
188     // start date of the HO is at the end of both nav
189     act.setStartDateHO(obj_time);
190     obj_time += solution->handover_times[i];
191     act.setEndDateHO(obj_time);
192     for (uint a = 0; a < 2; ++a) {
193       distance += act.dist(a);
194       time_nav += act.dist(a) / act.agent(a)->speed;
```

```
195         time_nav_cost +=
196             act.dist(a) / act.agent(a)->speed * act.agent(a)->use_cost;
197         time_ho_cost += solution->handover_times[i] * act.agent(a)->use_cost;
198       }
199       time_ho += solution->handover_times[i];
200       hri_cost = std::max(hri_cost, solution->handover_hri_costs[i]);
201     }
202     // for the paths to target positions
203     for (uint i = 0; i < solution->nav_dists_to_target.size(); ++i) {
204       double d = solution->nav_dists_to_target[i];
205       double time = d / solution->agents[i]->speed;
206       double start =
207           solution->actions[min((int)i, (int)solution->actions.size() - 1)]
208               .endDateHO();
209       // [min()] : the frist and last agents go to their targets after their
210       // 1st (and unique) HO, unlike others
211       solution->nav_start_end_times_to_target[i] =
212           make_pair(start, start + time);
213       distance += d;
214       time_nav += time;
215       time_nav_cost +=
216           d / solution->agents[i]->speed * solution->agents[i]->use_cost;
217     }
218     // and the last agent is holding the object when going to its target so
219     // add the corresponding time to the solution duration
220     if (solution->nav_dists_to_target.size() &&
221         solution->nav_dists_to_target.back() > 0) {
222       obj_time += solution->nav_dists_to_target.back() /
223                   solution->agents.back()->speed;
224     }
225     solution->time = obj_time;
226     solution->cumul_time_nav = time_nav;
227     solution->hri_cost = hri_cost;
228     double norm_dist, norm_obj_time, norm_use_cost;
229     norm_dist = distance / _max_dist;
230     norm_obj_time = obj_time / _max_time;
231     norm_use_cost = (time_ho_cost + time_nav_cost) / _max_use_cost;
232     cost = _f_dist * norm_dist + _f_time * norm_obj_time +
233            _f_agent_cost * norm_use_cost + _f_hri * hri_cost;
234     assert(std::abs(cost - (_f_dist * solution->getTotalDist() / _max_dist +
235                             _f_time * solution->time / _max_time +
236                             _f_agent_cost * solution->getTotalUseCost() /
237                                 _max_use_cost +
238                             _f_hri * solution->hri_cost)) < 0.00001);
239     if (isnan(cost)) {
240       cost = numeric_limits<double>::infinity();
241     }
242     solution->cost = cost;
243     return cost;
244   }
```

Appendix B

Résumé Français: Une Formulation Nouvelle et des Schémas de Résolution pour la Planification de Tâches Collaborative Humain-Robot

---

Nous présentons ici un résumé en langue française de ce document. Nous nous attardons essentiellement sur la présentation des contributions plus que sur leur implémentation et les résultats. Ce résumé ne saurait être une vue exhaustive de la version originale, et n'est là que pour permettre au lecteur non-anglophone de saisir la teneur de nos travaux.

Sommaire

B.1 Introduction

B.1.1 Contexte et Tâches

La collaboration Humain-Robot est un domaine portant de nombreux défis et tout autant de manières de les aborder. Au cours de la préparation de ma thèse, je me suis concentré sur certaines tâches nécessitant d'être traitées par des approches spécifiques. Ces tâches étaient liées aux projets Européens finançant mes travaux :

— ARCAS (Aerial Robotics Cooperative Assembly System [1] : assemblage de structures par des robots aériens coopératifs) ;
— SAPHARI (Safe and Autonomous Physical Human-Aware Robot [2] : prise en compte de l'Humain par un robot assistant d'atelier) ;
— MuMMER (MultiMedia Mall Entertainment Robot [3] : robot de divertissement pour un centre commercial).

Les deux tâches principales étudiées sont le problème de Transport Multi-Agent et une tâche liée à l'indication de chemin. La première consiste à transférer un objet (handover : transfert de main à main) entre plusieurs agents coopératifs (*c.à.d* robots et humains) afin qu'il atteigne une position ou un agent donné. La seconde tâche consiste à donner des indications de chemin à des humains en utilisant la parole et les gestes, où l'interaction peut être améliorée en se déplaçant avec les personnes vers un endroit où les indications seront plus simple, grace à une perspective permettant de pointer du doigt des amers pertinents. Des tâches de manipulation en présence de l'humain ou avec sa collaboration ont aussi été étudiées ; essentiellement des tâches de saisie et pose (pick and place) avec un bras robotisé ou un robot humanoïde. Toutes ces tâches ont servi de moteurs et de tests pour nos principales contributions :

— MHO (Multiple HandOver) est une solution permettant de résoudre le problème de Transport Multi-Agent en considérant les préférences de l'Humain ;
— le planificateur SVP Planner (Shared Visual Perspective : Perspective Visuelle Partagée) correspond à une part de la tâche de guide, et est intégrée dans un système développé par notre équipe planifiant et exécutant ce type de tâches ;
— développement de move4d, un logiciel de calcul de mouvements pour les tâches collaboratives humain-robot ;
— contribution à GTP (Geometric Task Planner : Planificateur Géométrique de Tâches), qui est un élément de move4d.

Ces contributions ont en commun leur objectif d'améliorer la résolution et l'exécution des tâches collaboratives Humain-Robot, en termes de qualité ou de coût de calcul.

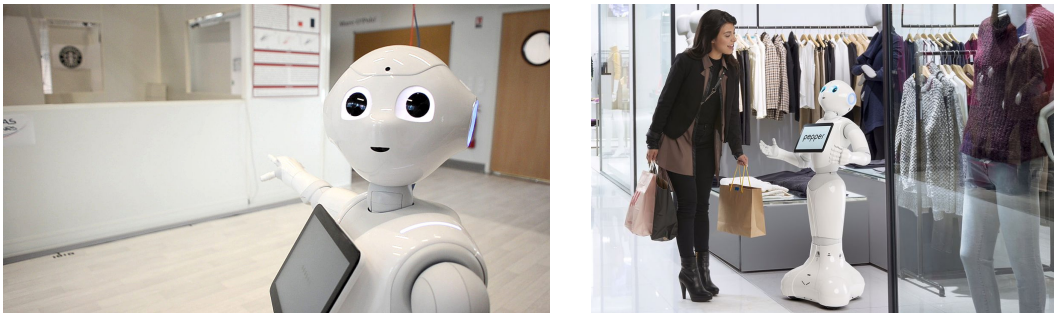


FIGURE B.1: Le robot Pepper de SoftBank Robotics est un humanoïde de la taille d'un enfant de 8 ans (1.21 mètres), sur une base mobile roulante. (À droite une image promotionelle de SoftBank Robotics.)

B.1.2 Design de Tâches Collaboratives Humain-Robot

Un robot interagissant avec des humains dans une tâche collaborative doit afficher diverses capacités, notamment la perception, la planification d'activités de haut niveau et la supervision, l'estimation d'état mental et le mouvement. Pour que le robot présente un intérêt quelconque pour ses partenaires humains, il doit s'avérer utile pour assurer l'engagement des partenaires dans la tâche. Les principaux aspects de l'utilité sont l'efficacité et l'adaptation aux besoins.

De nos jours, et probablement pour quelque temps encore, les plates-formes robotiques ont des capacités très limitées comparées à (la plupart) des humains, mais peuvent nous surpasser dans certains domaines spécifiques (précision, force, mémoire, compliance). Pour que ces plates-formes soient utiles, les concepteurs et les développeurs doivent adapter leurs comportements à leurs capacités et les capacités attendues aux capacités réelles. Ne pas le faire peut engendrer des malentendus, des illusions ou de l'impatience si le robot n'est pas capable de faire la tâche de manière satisfaisante tout en prétendant (explicitement ou non) qu'il est capable de les accomplir.

A titre d'illustration, considérons le robot Pepper de SoftBank Robotics, illustré à la figure B.1. Son design ressemble à un enfant de huit ans avec une base mobile et une tablette sur le torse. Une telle apparence induit une attente inconsciente chez l'humain, qui peut se sentir dérouté pendant l'interaction lorsque le robot s'avère moins puissant, adroit ou sensible qu'attendu. De plus, si le développeur programme le robot pour une tâche de collaboration pour laquelle il est inefficace, les partenaires humains s'attendent à ce que le robot soit utile dans l'interaction dans laquelle ils se sont engagés et seront de nouveau déçus, ce qui pourrait entraîner un désengagement de la tâche.

Lors de la conception d'un robot à des fins commerciales, il s'agit bien sûr d'une question importante de succès. Mais dans la recherche IHR (Interaction Humain-

1. http://www.arcas-project.eu/
2. http://www.saphari.eu/
3. http://www.mummer-project.eu/

Robot), des humains pleinement engagés sont également nécessaires, car les études ou les expériences des utilisateurs impliquant des robots collaboratifs ne doit pas être dérangé par une deception du partenaire humain. Dans cet esprit, nous allons construire et programmer un robot afin qu'il se révèle utile et qu'il partage efficacement le travail en fonction des capacités de tous les partenaires.

Lors de la construction d'un robot pour une application, les concepteurs doivent comprendre les motivations et les besoins de l'homme et y répondre avec les solutions robotiques (lors de la conception d'une étude utilisateur, ils peuvent vouloir se concentrer sur les motivations et les attentes, car les volontaires ne viennent généralement pas avec un besoin).

Nous avons essayé de garder cela à l'esprit, afin que les solutions développées puissent s'adapter aux capacités des robots, aux capacités humaines et aux besoins. Le solutionneur de la tâche de transfert multiple peut s'adapter à différentes vitesses de robot, à la disponibilité humaine ou à l'engagement dans la tâche, tout en laissant les humains à leurs propres activités lorsque l'on peut se passer de leur aide.

La tâche de pointage s'adapte aux capacités de navigation du robot en limitant la distance parcourue pour montrer la cible. Dans certains cas, le système peut décider de ne pas la pointer du doigt, mais plutôt de donner des instructions pour y arriver. Ainsi, il ne sera pas demandé à l'humain de s'engager dans une tâche collaborative pour laquelle il pourrait trouver le robot inapproprié (naviguer dans un environnement encombré tout en guidant un humain, dans le contexte du projet MuMMER, peut être une tâche difficile pour le robot Pepper).

B.1.3 Planification pour des équipes d'Humains et de Robots

Les tâches collaboratives abordées dans cette thèse sont considérées comme des actions conjointes, c'est pourquoi nous formulons l'hypothèse d'un objectif commun pour l'équipe de robots et d'humains. Cette hypothèse nous permet d'en formuler une autre : que les partenaires humains auront tendance à faire ce qu' ils doivent faire. Ensuite, nous pouvons approcher les tâches collaboratives d'une manière similaire à la planification de tâches multi-agents, où le planificateur calcule les actions à effectuer pour les agents robotisés comme pour les humains. Dans la plupart des cas, nous n'avons pas besoin de planifier précisément pour les humains, car ils ne peuvent être contrôlés comme un robot ;mais les actions peuvent être envoyées à des humains, finalement après vérification de leur faisabilité.

Lorsque les robots sont au service des humains avec lesquels ils interagissent, il est important que tous les planificateurs prennent en compte l'objectif de l'humain. Contrairement à de nombreuses approches découplées d'affinement de plans, nous pensons qu'un meilleur comportement robotisé est réalisé lorsque tous les niveaux de planification partagent les informations sur ce qu'est la tâche, et donc sur quel est l'objectif global.

Prenons l'exemple d'un robot de service dans un hôpital pour aider le personnel en transportant des objets (appareils, médicaments, nourriture, etc.). Lorsqu'il est nécessaire d'apporter un objet spécifique à quelqu'un, le robot peut soit le remettre

à la personne, soit le placer à proximité. Dans chaque cas, il doit décider où se rendre pour la remise ou le placement de l'objet, planifier comment s'y rendre, s'y déplacer, planifier les mouvements de bras et les exécuter tout en contrôlant quand libérer l'objet (particulièrement important lors d'une remise). Ici, nous voulons simplement que le lecteur ait l'intuition que chacune de ces six fonctionnalités doit être consciente de certaines informations sur la tâche globale, et pas seulement de leur propre sous-ensemble du problème. Sinon, des comportements du robot inadaptés au contexte immédiat peuvent causer des erreurs plus ou moins sévères.

C'est évident pour la plupart d'entre nous, les roboticiens mettent en oeuvre des systèmes capables de livrer en toute sécurité un bol de soupe chaud à une personne affaiblie ou de naviguer dans des environnements encombrés. Mais quand il s'agit de créer un système polyvalent évoluant dans des contextes variés, nous devons nous assurer que le robot utilise la bonne méthode au bon moment, et cela implique de partager des informations entre des composants très différents qui fonctionnent généralement sur des types d'informations très distincts et communiquent mal, voire pas du tout.

B.1.4 Résumé des Contributions

Nous avons commencé par travailler sur une tâche de transfert d'objet entre plusieurs agents (Multiple Agent Handover, Chapitre B.3), en relation avec de nombreux sujets de recherche en robotique : collaboration homme-robot au niveau de la tâche et du mouvement, transfert, navigation considérant l'humain. Le planificateur prend alors en compte de nombreuses informations sur le scénario et les partenaires pour trouver un plan conforme aux utilisateurs. La planification est en fait divisée en plusieurs niveaux avec différents espaces de recherche : niveau tâche, navigation, manipulation, planification des mouvements. Bien que l'approche globale soit satisfaisante et donne de bons résultats, les algorithmes de mouvement devaient être améliorés. Nous nous sommes concentrés pendant un certain temps sur la planification optimale de mouvements, en nous concentrant sur l'amélioration de la dynamique du mouvement des robots et sur l'adaptation des mouvements, en ligne et en temps réel. Tout cela dans le but de rendre le mouvement des robots plus acceptable pour les humains (Chapitre B.7). Revenant à la planification des tâches et des mouvements, j'ai contribué à une structure logicielle permettant une meilleure connexion entre les planificateurs symboliques et les raisonneurs géométriques (donc les planificateurs de mouvements), tout en ayant pour objectif d'améliorer l'acceptabilité et l'efficacité des robots dans les tâches collaboratives humains-robots (Chapitre B.6). Dans le même esprit que la tâche de transfert entre plusieurs agents, nous nous sommes intéressés à une tâche d'orientation (guidage) dans laquelle des actions pour les humains et les robots sont planifiées, et les mouvements sont générés en tenant compte de la tâche globale (Chapitre B.4 & B.5).

B.1.5 Liste des Publications

B.1.5.1 Publiés

— Jules Waldhart, Mamoun Gharbi, and Rachid Alami. Planning Handovers Involving Humans and Robots in Constrained Environment. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6473–6478, Hamburg, Germany, September 2015. IEEE.

— Jules Waldhart, Mamoun Gharbi, and Rachid Alami. A Novel Software Combining Task and Motion Planning for Human-Robot Interaction. In *AAAI Fall Symposia*, Washington D.C., 2016.

B.1.5.2 Soumis

— Jules Waldhart, Aurélie Clodic and Rachid Alami. *The SVP Planner : Planning Human-Robot Shared Visual Perspective to Provide Route Direction.* Submitted to HAI '18 the 6th International Conference on Human Agent Interaction. ACM, 2018.

Acceptés

— Jules Waldhart, Aurélie Clodic and Rachid Alami. *Planning Human and Robot Placements for Shared Visual Perspective.* In Robotic Co-workers 4.0 : Human Safety and Comfort in Human-Robot Interactive Social Environments Workshop at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018

B.2 État de l'Art

Nous présenterons un aperçu des travaux liés à la planification et à l'exécution de tâches collaboratives impliquant des humains et des robots. Nous examinerons d'abord les contributions dans les domaines de la planification : planification des mouvements, planification des tâches et leur fusion. Les travaux spécifiques à l'interaction homme-robot seront ensuite présentés, ce qui nous conduira à une troisième partie où un aperçu des techniques de planification des tâches collaboratives homme-robot sera fourni.

B.2.1 Planification de Tâches et de Mouvements

Les systèmes robotisés intégrés dans notre société en tant que robots de service ou assistants doivent impérativement gérer la complexité et la variabilité de leur environnement, tout en exécutant leurs tâches de façon sûre et robuste. Ces robots doivent planifier et raisonner à un niveau élevé pour décider des actions à entreprendre en fonction de leur objectif et de leur environnement. Mais ils doivent aussi vérifier la faisabilité de leurs plans concernant les collisions, la stabilité, la dynamique, les échéances, et ainsi de suite. Habituellement, la planification de haut niveau est résolue par la recherche d'une série d'actions, dont les conditions préalables et les effets sont déjà connus, qui transforment l'environnement en un état qui satisfait un objectif [Ghallab 2016]. D'autre part, la planification au niveau physique, lorsque des mouvements réalisables sont générés, lie généralement deux états physiques entièrement définis avec un mouvement [LaValle 2006]. Cela laisse un écart entre les actions symboliques trouvées au niveau de la tâche (haut niveau), comme "*saisir le tournevis*" ou "*sortir de la pièce*", et les demandes de communication à la planification des mouvements, qui seraient "*déplacer le gripper de A à B, puis fermer le gripper*" ou "*naviguer de* (x, y) *à* (x', y')". Ce problème de recherche d'une proposition de spécification d'action de haut niveau s'appelle *raffinage*. Le problème souvent appelé "Task and Motion Planning" (TMP, planification des tâches et des mouvements) traite de la question générique de la recherche d'un plan de mouvement réalisable qui réponde à un objectif symbolique.

L'intégration de différents niveaux de raisonnement est essentielle pour faire face à des systèmes, environnements et tâches de plus en plus complexes. L'augmentation récente du nombre de publications traitant de ce problème particulier montre qu'il suscite un vif intérêt. Ce problème a été nommé et résolu par une multitude de méthodes, [Erdem 2016] et [Gharbi 2015a] présentent un aperçu de la littérature dans ce domaine et proposent des classifications de ces méthodes. Nous allons brièvement passer en revue certaines des techniques proposées pour résoudre le problème de TMP. Certaines approches affinent le plan symbolique jusqu'au niveau géométrique par un parcours en profondeur [Dornhege 2012]. D'autres approches consistent à rechercher d'abord plusieurs plans de haut niveau, puis à les vérifier par rapport aux contraintes géométriques pour finalement produire un plan de mouvement réalisable pour l'un des plans de haut niveau [Şucan 2012, Lagriffoul 2014].

Une approche similaire est d'abord de rechercher un plan symbolique, puis de le raffiner au niveau géométrique, et lorsque des actions sont impossibles, utilisez cette information pour guider la recherche au niveau symbolique pour produire un nouveau plan [Lozano-Pérez 2014, Srivastava 2013, Caldiran 2009]. Dans certains travaux, dont [Zickler 2009, Nedunuri 2014, Garrett 2014, Plaku 2010, Barry 2013], la recherche d'un mouvement se fait directement au niveau géométrique, utilisant des informations symboliques ou un plan pour guider la recherche. Enfin, [Hauser 2009, Cambon 2009] proposent des solutions où la recherche se fait simultanément aux deux niveaux.

Notre travail tient compte de ces questions et nous nous sommes inspirés de certaines techniques utilisées pour résoudre le problème de l'intégration de la planification des tâches et des mouvements. En effet, nous voulions développer des "*solveurs de tâches*" qui seraient en mesure d'affiner une demande symbolique à une solution physique et réalisable, ce qui est une question centrale de la planification des tâches et des mouvements. Mais dans notre cas, nous nous sommes concentrés sur des tâches spécifiques et proposons des solveurs spécialisés (solveurs MHO et planificateur SVP, respectivement présentés dans les chapitres B.3 et B.4), qui pourraient cependant être intégrés dans des cadres TMP plus génériques.

B.2.2 Interaction Homme-Robot

Outre la planification de l'action dans des environnements complexes et pour des tâches complexes, le futur robot développé pour partager l'environnement humain doit se comporter de manière acceptable. Le domaine de l'interaction homme-robot (HRI) traite de nombreux sujets liés à la façon dont un robot interagissant avec les humains doit bouger, prendre des décisions, communiquer, etc. La HRI est très lié à la psychologie humaine car nous devons comprendre les comportements et les attentes des humains pour créer des robots engageants qui sont efficaces dans les interactions sociales. Nous allons d'abord présenter dans ces sections des travaux d'étude des mécanismes que les humains utilisent pour atteindre des objectifs communs en tant que collaborateurs : théorie de l'action conjointe. Ensuite, nous nous concentrerons sur les travaux lié à la génération de mouvement robotisé pour de telles tâches de collaboration avec les humains.

Certains aspects plus spécifiques de la littérature seront traités par chapitre : le transfert homme-robot est traité dans B.3.2, et le travail lié à l'orientation et à la fourniture de directions d'itinéraire dans B.4.3.

B.2.2.1 Théorie de l'Action Conjoint chez l'Humain

Comprendre la façon dont les humains réalisent des actions conjointes peut nous aider à donner au robot la capacité d'effectuer une telle interaction collaborative avec les humains. Nous utiliserons la définition de l'action conjointe proposée par [Sebanz 2006] :

> Joint action can be regarded as any form of social interaction whereby

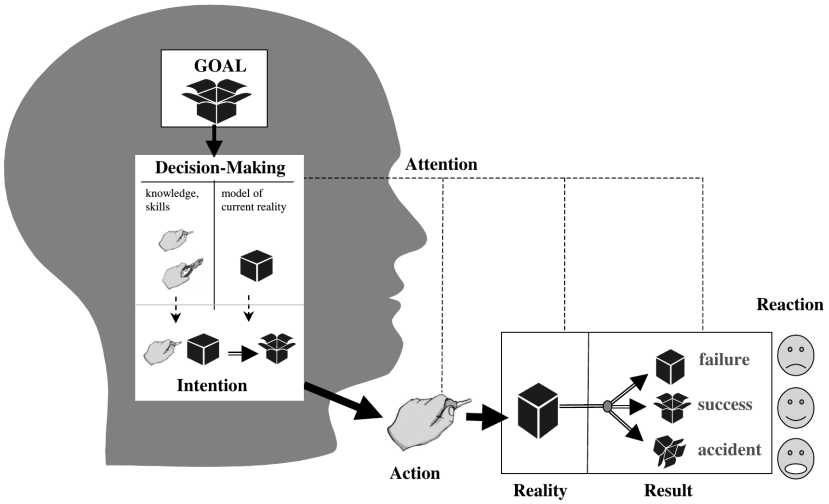


FIGURE B.2: Exemple tiré de [Tomasello 2005]. La personne forme une intention à partir de son but, de ses connaissances et de ses compétences.

> two or more individuals coordinate their actions in space and time to bring about a change in the environment.

("L'action conjointe peut être considérée comme n'importe quelle forme d'interaction sociale par laquelle deux individus ou plus coordonnent leurs actions dans l'espace et dans le temps pour apporter un changement dans l'environnement.")

Pour parvenir à une action commune, les individus doivent s'entendre sur ce "changement dans l'environnement" qu'ils souhaitent apporter, comment ils le feront et dans quelles conditions ils restent engagés dans la tâche. Ceci est lié à l'engagement que nous allons développer dans la Section B.2.2.1. Comme il est indiqué dans la définition ci-dessus, les agents doivent se coordonner entre eux, ce qui exige à leur tour de percevoir et de prévoir les actions de leurs coéquipiers. Nous allons développer ceci dans la Section B.2.2.1.

Engagement Pour qu'une action soit menée par un agent, ce dernier doit avoir un *objectif*, c'est-à-dire une représentation de l'état souhaité du monde, et une *intention* pour y parvenir. Pour un seul individu, [Tomasello 2005] présente une tâche dans laquelle une personne doit ouvrir une boîte. L'*objectif* est la boîte ouverte, et l'*intention* est d'utiliser une clé pour ouvrir la boîte (Figure B.2).

La définition de l'intention dans [Cohen 1991] diffère de celle de Tomasello *et al.* : ils la définissent comme l'engagement de l'individu pour atteindre l' objectif. Toutefois, dans leur définition, l'intention reste liée à l'objectif.

Nous pouvons maintenant étendre ces définitions de l'action individuelle à l'action conjointe. Une définition de l'*intention conjointe* est proposée par

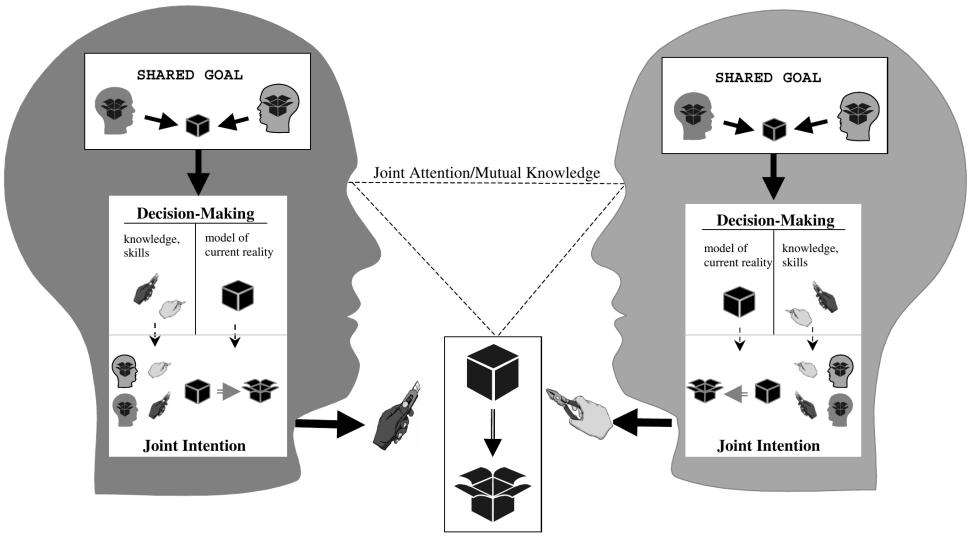


FIGURE B.3: Extension de l'exemple de Figure B.2 [Tomasello 2005] pour une action collaborative.

[Bratman 1989] :

> Nous avons l'intention de J si et seulement si :
>
> 1. (a) J'ai l'intention que nous fassions J et (b) tu as l'intention que nous fassions J.
> 2. J'ai l'intention que nous fassions J conformément à et en raison de 1a, 1b, et des sous-plans concordants des points 1a et 1b ; tu as l'intention que nous fassions J conformément à et en raison de 1a, 1b des sous-plans concordants de 1a et 1b.
> 3. 1 et 2 sont des connaissances communes entre nous.

[Tomasello 2005] étend l'exemple de la boîte à cette défintion (Figure B.3). Un agent tient la boîte avec une pince et l'autre l'ouvre avec un cutter.

L'*objectif partagé* est la représentation d'un état désiré et du fait qu'il sera atteint en collaboration avec d'autre(s) personne(s), et une *intention conjointe* est un plan collaboratif dans lequel les agents s'engagent afin d'atteindre l'objectif partagé et qui prend en compte les plans indiviuels des deux agents.

Pour [Cohen 1991], il y a *intention conjointe* quand tous les membres savent qu'il veulent tous atteindre l'objectif. Leur définition ne considère pas *comment* les agents vont atteindre l'objectif, mais pour eux les agents doivent s'informer mutuellement de l'état d'avancement de l'objectif (atteint ou non, ou sans rapport avec la tâche).

En ce qui concerne la manière d'atteindre l'*objectif partagé*, la notion de *plan partagé* est introduite par [Grosz 1988, Grosz 1999]. Dans leur définition, il existe un plan pour atteindre l'objectif commun, mais aucun individu ne connaît néces-

sairement l'ensemble du plan commun, chaque individu n'a besoin de connaître que sa partie du plan commun.

Coordination Les psychologues étudient les mécanismes et les signaux à l'appui de la coordination, [Vesper 2017] est une revue récente des travaux dans ce domaine.

En ce qui concerne notre travail, la théorie de l'action conjointe peut nous aider à :

1. concevoir un comportement de robot qui répond au besoin d'information des coacteurs ;
2. surveiller et comprendre les signaux humains ;
3. concevoir des systèmes efficaces qui tirent profit des capacités élevées d'action conjointe de l'homme pour accroître l'efficacité de la tâche et la satisfaction et le confort des utilisateurs.

B.2.2.2 Générer des Mouvements pour l'Action un milieu Humain

Générer un mouvement robotisé est une tâche complexe, même lorsque seuls des mouvements fonctionnels sont nécessaires : [LaValle 2006] fournit un bon aperçu des difficultés et des approches existantes du problème de planification des mouvements. Les difficultés incluent l'espace dimensionnel élevé, les collisions, les limites de couple, les contraintes de dynamique [Boeuf 2014, Li 2015] et souvent contrainte de pose d'effecteur final [Stilman 2007, Koga 1994, Ahuactzin 1999, Yao 2005]. Cependant, les récentes améliorations des processeurs et des algorithmes d'optimisation permettent maintenant de générer des mouvements optimisant les coûts [Jaillet 2008, Devaurs 2013, Zucker 2013]. Dans notre cas, le coût du mouvement refléterait les propriétés requises pour générer le mouvement en fonction de ce que les humains attendent d'un agent robotique.

Il existe de nombreuses contributions concernant les actions de navigation [Kruse 2013, Rios-Martinez 2015]. En ce qui concerne la manipulation, les contributions se limitent souvent à une action spécifique telle que le transfert [Cakmak 2011b, Huang 2015, Kupcsik 2016] ou à quelques propriétés du mouvement de manipulation telles que sécurité, confort [Sisbot 2012], lisibilité [Dragan 2013a] , visibilité [Sisbot 2007b], timing [Zhou 2017]. Passons en revue certains des critères importants de la génération de mouvements pour l'interaction avec l'homme et de la façon dont ils sont traités dans la littérature robotique.

Sécurité La sécurité est la principale préoccupation lors de la conception de robots qui travaillent dans l'environnement humain. Les approches communes comprennent le respect d'une certaine distance de sécurité avec l'homme, éventuellement en prévoyant leur mouvement pour planifier en conséquence [Kruse 2013, Rios-Martinez 2015] ; la distance se rapporte aussi à la théorie de la *proxémie* [Hall 1969], qui peut être utilisée pour augmenter le confort des humains. La plupart des planificateurs de mouvements et de navigation pour l'interaction avec l'humain

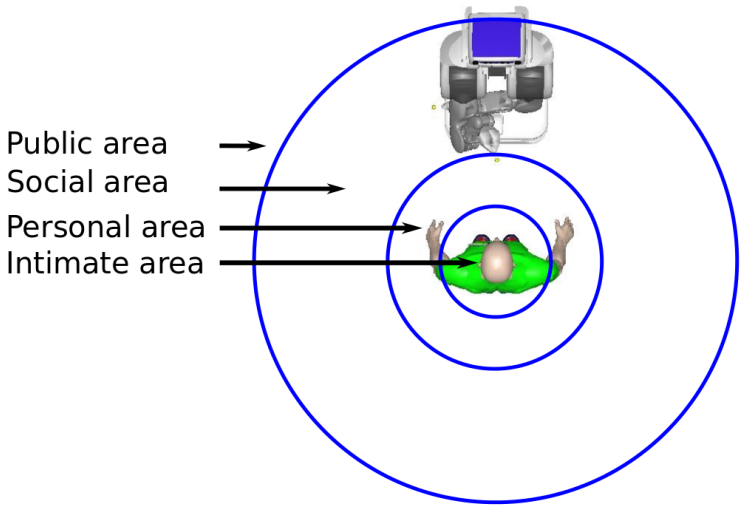


FIGURE B.4: les quatre zone de *proxemie* autour de l'humain comme définit par Hall [Hall 1969]. Un robot interagissant avec l'humain doit rester dans la zone "Sociale".

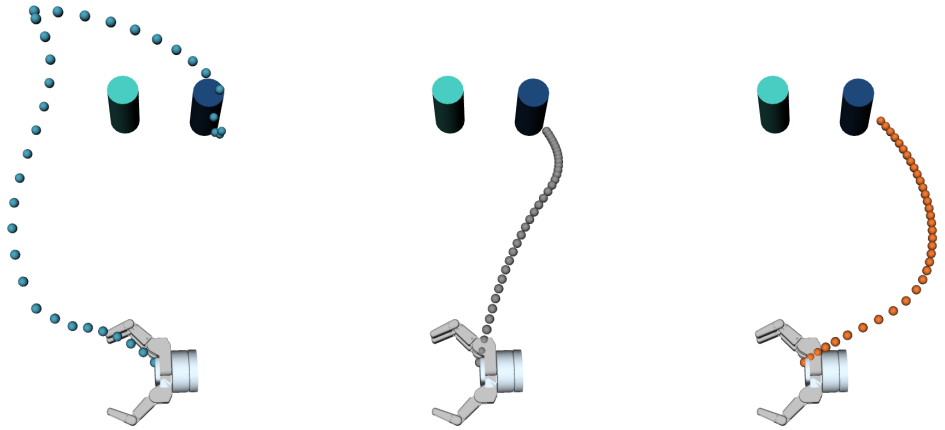
sont encore construits à partir d'un coût proxémique. Mais les planificateurs récents explorent les autres aspects importants des mouvements collaboratifs.

Lisibilité et Prévisibilité du Mouvement Robotique Les mouvements d'une tâche collaborative peuvent transmettre de l'information, même lorsque leur objectif premier n'est pas la communication. [Dragan 2015] défini trois attributs du mouvement : fonctionnalité, prévisibilité et lisibilité.

La figure B.5 montre trois mouvements où un robot doit saisir l'objet sur la droite, chaque mouvement est représenté par les points de son chemin et chacun d'eux présente l'un des attributs ci-dessus. Dans le mouvement fonctionnel, la pince effectue quelques courbes dans l'espace de travail avant d'atteindre l'objectif ; le mouvement prévisible est plutôt direct vers l'objectif ; et le mouvement lisible exagère sa courbe de sorte que l'observateur peut faire une distinction entre les deux cibles dès le début de l'exécution du mouvement.

[Kruse 2012] présente un planificateur de navigation inspiré des comportements humains afin d'amener le robot à avoir un comportement plus lisible lorsqu'il croise le chemin des humains.

Visibilité du Robot en Mouvement Lorsqu'il se déplace, un robot ou une pièce de robot peut devenir visible ou cachée à ses partenaires. L'état de visibilité est important à prendre en considération lors du déplacement, mais le changement d'état de visibilité a une incidence importante sur l'acceptabilité du mouvement. [Koay 2007] est une étude qui montre qu'un robot doit éviter de se rapprocher de l'homme sans être visible par lui, comme quand il s'approche de derrière ou quand



Functional Predictable Legible

FIGURE B.5: illustrations de [Dragan 2015] montrant les différences entre mouvements fonctionnels, prévisibles et lisibles.

il apparaît soudainement de derrière un obstacle proche de l'humain. [Sisbot 2007b] implémente un planificateur de navigation qui prend explicitement en compte la visibilité du robot du point de vue du partenaire humain.

Attention Partagée entre l'Humain et le Robot La coordination dans le cadre d'une action commune est en partie soutenue par une attention commune. Des systèmes permettant aux robots de partager leurs points d'attention avec les humains ont été développés et étudiés. Par exemple, dans [Marin-Urias 2009], ils peuvent calculer ce que l'humain regarde pour que le robot le regarde aussi, créant ainsi une attention partagée initiée par l'humain.

Des robots orientant l'attention des partenaires humains grâce à sont propre regard ont été mis en {oe}uvre avec succès, comme dans [Mwangi 2018].

B.2.3 Plans Impliquant le Partenaire Humain

Quelques études récentes se concentrent maintenant sur les planificateurs capables de générer des plans pour l'équipe, ce qui signifie que le robot crée un plan où les humains sont assignés à des tâches spécifiques nécessaires pour atteindre l'objectif commun.

L'idée de planifier non seulement pour les robots, mais aussi pour les humains qui coopèrent avec eux n'est pas intuitive. En effet, le système ne peut pas contrôler les humains. Cependant, la théorie de l'action conjointe nous enseigne sur l'objectif commun, l'intention commune et le plan partagé, et concernant la robotique [Clodic 2017] étudie quelles sont les caractéristiques essentielles nécessaires aux collaborations homme-robot, y compris le raisonnement sur les plans partagés. L'action conjointe existe parce que les agents sont prêts à partager l'effort, ce qui signifie que

les partenaires humains agissent aussi. Lorsqu'ils établissent leur propre partie du plan commun, les agents tentent alors de prévoir ce que les partenaires feront, grâce à leurs connaissances sur leurs compétences, leurs connaissances et leurs possibilités. Cela a été fait au niveau de la tâche par [Alili 2009, Lallement 2014, Lallement 2016] avec le *Hierarchical Agent Task Planner* (HATP). Associé au *Geometric Task Planner* (GTP) [Gharbi 2015b], le système planifie le mouvement des partenaires pour la tâche qui leur est assignée par HATP. Les systèmes fonctionnels basés sur le planificateur HATP sont présentés dans [Lemaignan 2017, Devin 2016].

Sur une tâche plus spécifique, [Mainprice 2012] propose un planificateur qui prend en compte le partage de l'effort pour construire un plan. Ils cherchent le meilleur endroit où aller pour remettre un objet à un partenaire humain, en tenant compte de l'effort qu'il est disposé ou capable de fournir pour accomplir la tâche. Ils tiennent compte de la distance parcourue par l'humain et du confort de la posture de transfert. Leur planificateur peut alors produire des plans très différents en fonction de l'urgence de la tâche ou de la condition physique humaine ou du désir d'être assisté ou d'être proactif. La figure B.6 montre un exemple, où nous pouvons voir que pour partager l'effort, leur planificateur planifie un mouvement pour le partenaire humain, afin d'évaluer sa faisabilité et son coût.

Concernant la navigation, un robot peut proposer de façon proactive une solution de co-navigation à l'humain. [Khambhaita 2017] montre que la navigation homme-robot peut être vue comme une action conjointe dans certaines situations contraintes. Ils proposent un planificateur qui génère des trajectoires pour les deux partenaires dans un scénario de croisement dans un couloir. La trajectoire est conçue pour être lisible, de sorte que l'homme peut comprendre la solution proposée par le robot. La pertinence de ce travail a été récemment confirmée par une étude d'utilisateurs menée dans notre équipe, dont les résultats seront bientôt publiés.

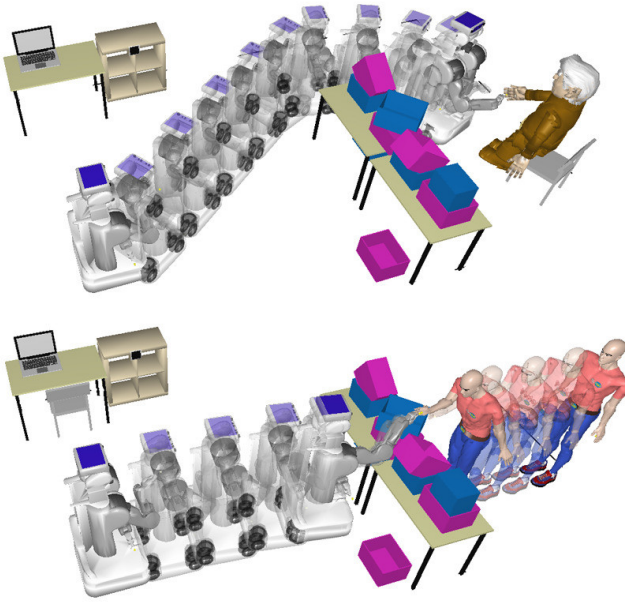


FIGURE B.6: Illustration issue de [Mainprice 2012] : pour la même tâche, le planificateur produit deux solutions très différentes ; dans le premier cas, la personne à qui donner l'objet est un vieil homme assis, le robot fait tout son possible pour lui apporter l'objet ; dans le deuxième cas, la personne est déjà debout et est mobile, l'effort peut être partagé.

B.3 Tâche de Multiples Transferts d'Objet

Planifier où effectuer les transferts successifs pour résoudre un problème de transport impliquant robots et humains.

Présenté dans le Chapitre 3

B.3.1 Introduction

Nous – les humains – manipulons avec efficacité les objets de notre environnement, que nous passons souvent de mains en mains. Cette action est appelée *handover* dans la littérature anglaise, et semble être une fonctionnalité clé pour des robots devant agir avec les humains. Nous allons présenter ici des travaux liés aux tâches de transport, qui peut impliquer de multiples agents, humains et robots. C'est une tâche où un objet doit être déplacé d'un agent à un autre, éventuellement avec l'aide de plusieurs autres agents présents. La figure B.7 montre un exemple de ce problème, les agents étant séparés par un mur où fenêtres et comptoirs permettent le transfert de l'objet. Nous avons traité les problèmes de décision du choix des agents à impliquer dans la solution, des lieux où les objets sont transférés, en préservant le confort des humains. Dans l'exemple de la figure B.7, une solution serait de faire se rencontrer les humains bleu et rouge pour se donner directement l'objet. L'utilisation du robot dans la solution réduit les efforts fournis par les humains.

La contribution présenté ici est un planificateur entrelaçant des plusieurs modèles, des niveaux abstraits (tâche) aux niveaux géométriques. Ce planificateur doit être capable de trouver efficacement des solutions de bonne qualité en considérant de nombreux critères liés aux normes sociales et au confort de humains. Le système a été testé en simulation puis déployé sur deux robot PR2 interagissant avec deux humains dans un environnement encombré.

B.3.2 Travaux Liés

B.3.2.1 Transfert d'Objet

Des difficultés apparaissent pour les transferts entre l'homme et le robot, concernant la sécurité humaine, le confort et d'autres règles sociales, pour qu'il soit aussi naturel que possible [Cakmak 2011a, Cakmak 2011c]. [Moon 2014] explore l'utilisation d'indices non verbaux fournis par le regard. Le problème de la détection du moment où un humain a l'intention de donner un objet à un robot a été abordé dans [Pan 2017, Pan 2018]. De même, il faut décider quand l'objet doit être libéré par le robot[He 2015]. Pour un transfert entre deux robots, la synchronisation est essentielle [Quispe 2014].

B.3.2.2 Navigation Adaptée à l'Humain

La tâche de transport dans notre contexte exige que les robots naviguent dans un espace occupé par les humains, qui peuvent être occupés. Dans ce contexte, le

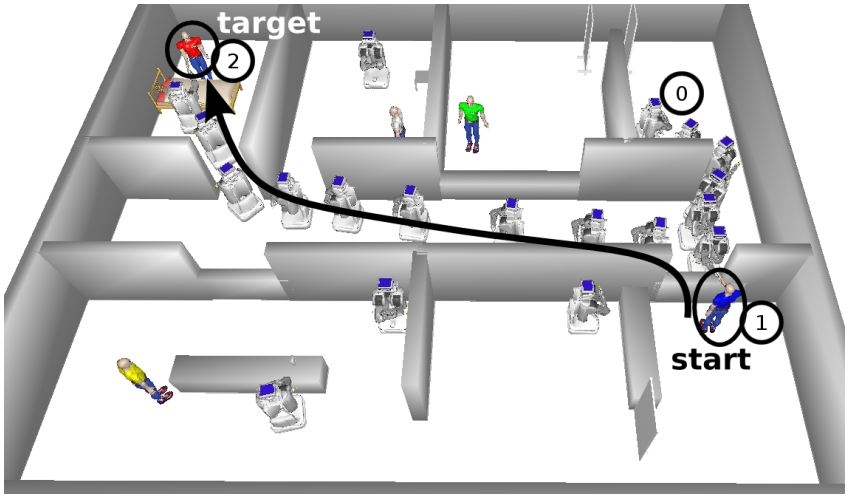


FIGURE B.7: Exemple dans un environnement où les agents sont séparés en deux zones navigables reliées par des fenêtres et des compteurs. L'homme bleu est en possession d'un objet dont le rouge a besoin. La solution trouvée par notre algorithme est la suivante : le robot 0 navigue jusqu'à l'humain bleu, prend l'objet à travers un transfert sur le comptoir, puis navigue jusqu'à l'humain rouge et le lui donne.

robot devra naviguer parmi eux, tout en créant le moins de perturbations possible. L'état de la technique de navigation est présenté dans le Chapitre B.2.

B.3.2.3 Planification de Manipulation

La planification de manipulation [Simeon 2004, Barry 2013], une extension de la planification de mouvement classique basée sur l'échantillonnage [LaValle 2006] pourrait résoudre le problème du transport. Néanmoins, il ne serait pas en mesure de traiter efficacement le complexe les coûts liés à la tâche et la très grande dimension de la espace de recherche.

B.3.2.4 Problème de collecte et de livraison

Le problème de transport est proche du problème de collecte et de livraison (PDP : Pick-up and Delivery Problem). Dans [Savelsbergh 1995], les auteurs présentent une revue sur les types de problèmes et les méthodes de résolution pour les PDP. Plus récemment, le lien entre PDP et le transfert d'objet a été souligné avec un algorithme où les robots se transfèrent des objets pour optimiser un plan PDP [Coltin 2014] (Figure B.8).

B.3.2.5 Multiples Transferts d'Objets

Un travail récent lié au problème de transfert d'objet entre plusieurs agents utilise une heuristique pour guider la recherche dans les transferts possibles

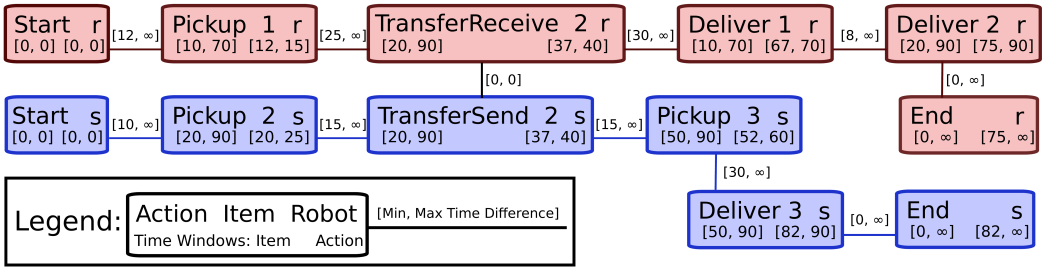


FIGURE B.8: Illustration tirée de [Coltin 2014] où ils planifient plusieurs livraisons et collectes où les objets à livrer peuvent être transférés entre robots pour optimiser le plan. L'image montre un réseau temporel de tâches avec deux robots, trois éléments à livrer et un transfert.

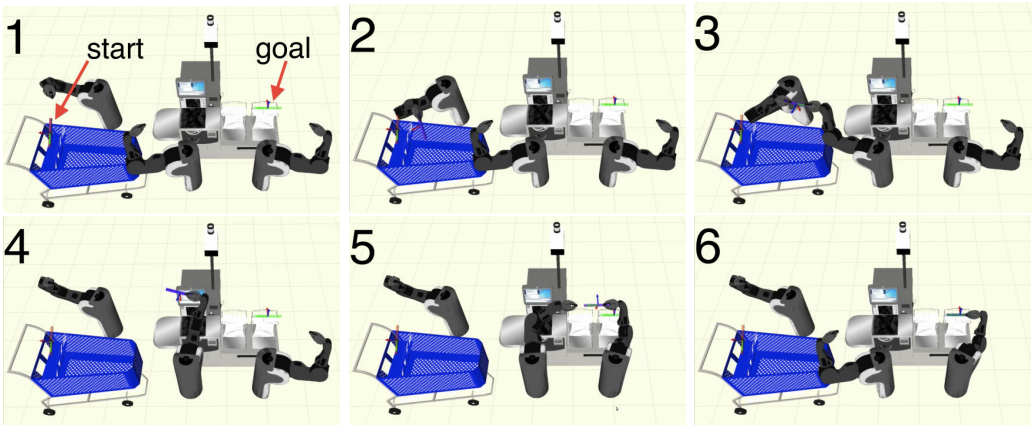


FIGURE B.9: Images de [Cohen 2014] : leur planificateur trouve une séquence de transferts pour n bras pour amener un objet à une position de but. Les images ci-dessus montrent les étapes successives de la solution : initiale, collecte, premier transfert, transfert, deuxième transfert, dépose.

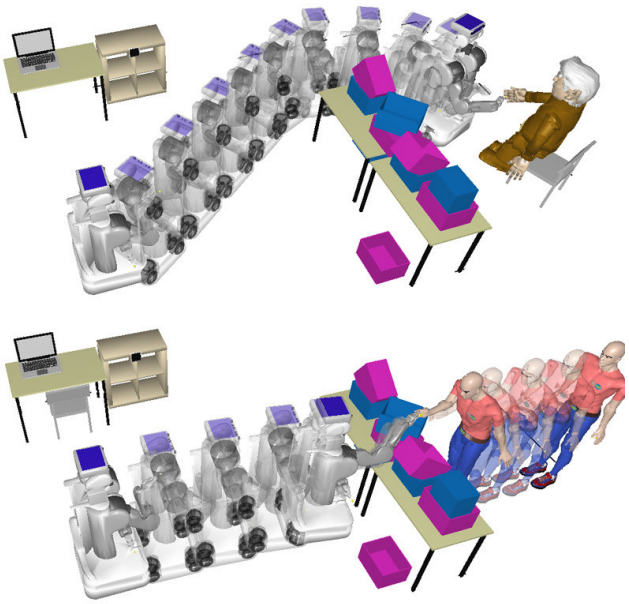


FIGURE B.10: Illustration de [Mainprice 2012] : pour la même tâche, le planificateur produit deux solutions très différentes, en prenant en compte un paramètre *mobilité* pour chaque agent qui reflète l'effort que l'agent peut faire. Dans l'image du haut, le vieil homme a une faible mobilité, il est donc *servi* par le robot, tandis que dans l'image du bas, l'homme a une mobilité similaire à celle du robot, donc l'effort est partagé pour accomplir la tâche.

[Cohen 2014] (voir Figure B.9). À la différence de l'approche précédente, les transferts sont nécessaires pour accomplir la tâche, et ne résultent pas d'une optimisation, car les robots sont des bras de manipulateur fixés. L'heuristique utilisée prend en compte l'estimation des coûts des transferts et une approximation du nombre restant les transferts qui seront nécessaires. Ils introduisent aussi une variante paresseuse de A* pondéré (*Lazy Weighted A*).

Dans [Mainprice 2012], les auteurs présentent un algorithme capable de partager les efforts entre un humain et un robot selon les préférences humaines. Dans certains cas, l'humain préférera peut-être se diriger vers le robot, alors que dans d'autres cas il préférera attendre que le robot vienne jusqu'à lui. Même dans ce dernier cas, si aucune solution n'est trouvée, le robot peut demander à l'homme de se déplacer pour effectuer la remise avec succès. Cette approche est limitée aux transferts uniques et ne s'étend pas à une série de transferts. Nous proposons une approche similaire adaptée à une séquence de transferts lorsqu'il est important de choisir les agents à impliquer et l'endroit où transférer l'objet. Nous utilisons le même concept de *mobilité* qu'ils introduisent : la mobilité est un facteur scalaire utilisé pour pondérer la quantité d' efforts que nous voulons qu'un agent fournisse par rapport à l'autre. Un agent à faible mobilité (proche de zéro) sera principalement servi par d'autres agents, fournissant un minimum d'effort, tandis qu'un agent à mobilité élevée fournira la majeure partie de l'effort (voir Figure B.10).

B.3.2.6 Justification

Nous présentons ci-dessous un outil qui produit un plan de résolution d'un problème de transport pouvant impliquer plusieurs agents hétérogènes, robots et humains. Le plan doit tenir compte des contraintes liées à la sécurité humaine et à l'acceptabilité. L'efficacité de cette tâche dépend de la capacité des robots à communiquer avec l'homme, en particulier en fournissant des indices non verbaux, de sorte que le plan doit être généré avec un algorithme centré sur l'humain. Le plan devrait optimiser les efforts et le confort de l'homme tout en respectant les contraintes de tâche.

Ce chapitre présente un système fonctionnel élaboré dans ce but. Le solutionneur de tâches prend explicitement en considération les contraintes et les objectifs liés à l'homme, mais n'a pas pour but de trouver des solutions à des sous-tâches telles que la navigation et le transfert ; il s' appuie sur des outils externes spécifiques à ces fins, qui sont appelés pendant la recherche. Nous présentons ici le planificateur de haut niveau et son lien avec les autres composants spécifiques principalement liés aux mouvements collaboratifs homme-robot.

Nous insistons sur le fait que ce travail ne se concentre pas sur l'algorithme pour décider comment effectuer un transfert, mais plutôt un algorithme qui sélectionne où effectuer des transferts et avec quels agents. Cependant, il prend en compte *comment* le transfert est effectué, mais grâce à des composants externes. Comme nous le présenterons plus tard, le planificateur décide également *quand* effectuer les transferts.

Pour résumer, notre planificateur décidera :
— *où* effectuer des transferts successifs ;
— *comment* les exécuter ;
— *quand* ils seront exécutés ;
— *comment* les transferts sont effectués, grâce à des outils externes.

B.3.3 Modèle

Nous représentons le modèle avec un *graphe principal* G, dont les nœuds représentent des états du monde et les arcs des transitions entre ces états ; nous chercherons le plus court chemin dans ce graphe en utilisant une heuristique et un coût représentants les différents paramètres de la tâche. Un nœud contient l'identifiant de l'agent portant l'objet et sa position : $[a_{id}, \{x, y\}]$. Deux actions permettent de changer de nœud : le déplacement élémentaire et le transfert d'objet. Deux nœuds sont reliés par un arc lorsqu'il existe un action permettant de passer d'un état à l'autre. L'arc et pondéré par le coût de l'action.

Une heuristique guide la recherche au travers des rencontres possibles entre agents. Nous utilisons un graphe secondaire G_A (pour graphe d'agents), où les nœuds sont les agents et les arcs existent lorsque les agents peuvent se rencontrer pour se transférer l'objet. Le graphe est initialisé avec des estimations optimistes des coûts des transferts possibles, et mis-à-jour durant la recherche lorsque des transferts s'avèrent impossibles.

Une grille 2D permet de calculer rapidement les déplacements des agents possibles dans l'environnement (algorithmes Dijkstra et A*).

Les positions et mouvements des transferts d'objets sont générés et validés avec des algorithmes de planification de mouvement classique reposant sur un modèle géométrique de l'environnement.

Le coût utilisé pour évaluer une solution est le suivant :

$$\begin{aligned} c = f_{util} \cdot \sum_{a \in A} t_{util}(a) \cdot f_d(a) + f_{temps} \cdot (t_{fin} - t_{début}) \\ + f_{HRI} \cdot max(c_{HO}(0), \ldots, c_{HO}(n)) \end{aligned}$$

$f_d(a)$ est un facteur appliqué à chaque agent a, $t_{util}(a)$ est le temps durant lequel un agent a est mobilisé par la tâche, $t_{début}$ et $tfin$ sont les dates de début et de fin de la tâche. Les c_{HO} sont les coûts des transferts d'objets.

B.3.4 Résolution

B.3.4.1 Algorithme de Recherche

Notre problème aillant un grand nombre d'états non nécessaire à la solution et des coûts parfois long à évaluer, nous avons choisi un algorithme *A* pondéré paresseux* (LWA, Lazy Weighted A**) [Cohen 2014]. Cet algorithme garanti une

sous-optimalité bornée par le facteur de pondération (il peut donc être optimal si le facteur et égal à un).

B.3.4.2 Tests Géométriques

Afin de vérifier la possibilité d'un transfert d'objet, les algorithmes de planification de mouvement peuvent être très coûteux en temps de calcul, en particulier dans les situations où il n'existe pas de solution. Nous préférons donc détecter ces cas impossibles le plus tôt possible avec des solutions moins coûteuses.

B.3.4.3 Ordonnancement

La sortie de l'algorithme LWA* ne représente pas les positions de tous les agents à chaque instants. Une étape de post-traitement est nécessaire pour ajouter toutes les actions de navigation sans l'objet. Ces actions ont déjà été calculées et prises en compte dans le coût de la solution, mais ne sont pas représentées dans le chemin renvoyé par l'algorithme de recherche.

Cette nouvelle solution n'est pas ordonnée. Nous utilisons le formalisme STN (*Simple Temporal Networks*) [Dechter 1991] pour ordonnancer toutes les tâches.

B.3.4.4 Traitement des Collisions

Les navigations générées à l'étape précédente peuvent créer des collisions entre les agents. Si une collision entre agents est détectée, le chemin de l'agent est modifiée pour contourner "l'obstacle". Si cela est impossible, une solution impliquant de déplacer les deux agents est recherchée. Cela modifie la qualité de la solution sans borne sur son évaluation, mais la modification reste locale et aussi minime que possible.

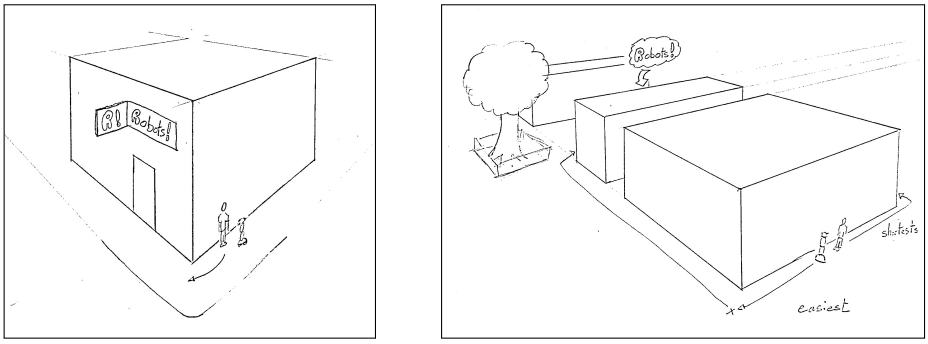
B.3.5 Expérimentation

Nous avons testé notre approche dans plusieurs scénarios, incluant un environnement réel aillant donné lieu à une implantation sur deux robots PR2 collaborant avec deux humains. Les temps de calcul obtenus sont assez satisfaisant et permettent un usage en ligne dans des instances de taille raisonnable (bureaux, petite industrie, maison). La variante pondérée de A* permet d'accélérer les calculs par un facteur allant jusqu'à 10 tout en conservant expérimentalement un coût inférieur à 2 fois le coût de la solution optimale. Des résultats détaillés sont présentés dans la version complète en anglais.

B.3.6 Conclusion

Le planificateur de tâche présenté tente de montrer une approche possible à la résolution de tâche collaboratives humain-robot, avec des agents aillant des engagements et des capacités différentes, mais qui peuvent partager l'effort quand cela

est nécessaire. Notre solution a aussi un coût en temps de calcul acceptable pour une utilisation en robotique.



(a) Le robot peut seulement dire où se trouve le magasin, ou bien passer l'angle avec l'humain pour lui montrer le magasin.

(b) Il y a deux chemin menant au magasin. Le plus long peut être meilleur car un repère saillant proche de l'objectif (l'arbre) est visible plus rapidement. Le robot peut accompagner l'humain là où l'arbre est visible, ou seulement donner des indications se basant sur ce repère.

FIGURE B.11

B.4 Guidage et Indications de Direction

Décider où aller pour donner des indications de chemin

B.4.1 Introduction

Le développement de fonctionnalités pour un robot de service et de divertissement pour les centres commerciaux, dans le cadre du projet MuMMER, nous avons étudié une tâche de guidage et d'indication de directions. Après une approche naïve et peu satisfaisante, nous avons considéré que cette tâche nécessitait de la planification et constituait une action jointe où l'objectif commun est que le partenaire humain atteigne son objectif. Cette tâche peut être résolue de plusieurs manières. Le robot peut simplement pointer vers la direction à prendre et donner quelques instruction ("Par là, prenez la première à droite,..."). Dans certains cas, il peut pointer directement la destination ("c'est juste ici."). Mais dans certains cas il peut être pertinent de se déplacer avec le partenaire vers une position où un point de repère (précédemment caché à la vue) devient visible ("c'est à droite quand vous arrivez à l'arbre", cf. Figure B.11b). Enfin, le robot pourrait accompagner le partenaire jusqu'à sa destination.

Le scénario implémenté au sein du projet MuMMER peut se résumer ainsi :

— un robot placé près d'un point d'information est disponible pour donner des informations et des indications de chemin ;
— il peut se déplacer de quelques mètres avec son interlocuteur humain vers un lieu où il peut pointer des points de repères et donner des indications de chemin ;
— le robot n'accompagne pas les personnes vers leur destination mais les aide à trouver un chemin.

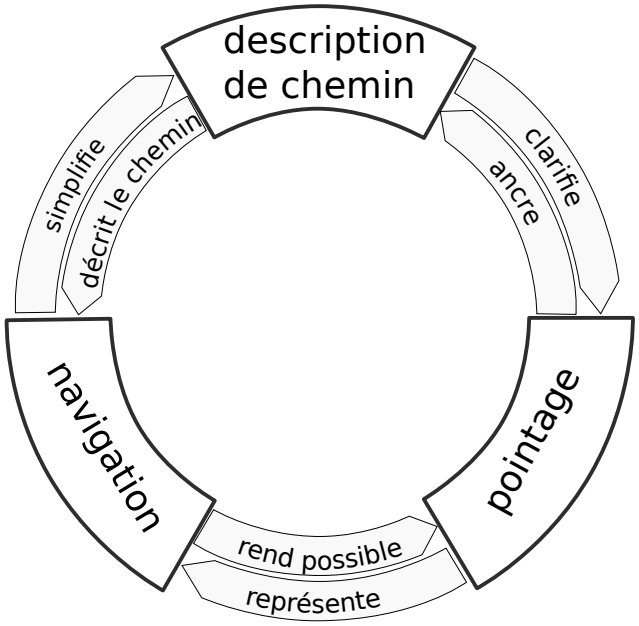


FIGURE B.12: Représentation schématique des liens entre les trois modalités de la tâche.

Il est intéressant de noter que la littérature abonde en contributions liées à des aspects spécifiques de la tâche, mais nous n'avons trouvé aucune traitant le scénario complet présenté ci-dessus. Les systèmes existants se distinguent entre :

— les guides robotisés qui accompagnent ou ouvrent la route aux personnes ;
— les systèmes fournissant uniquement des itinéraires ;
— les systèmes fournissant des directions de route en utilisant des repères où l'on suppose que les repères pertinents à pointer sont déjà visibles du point de vue humain.

B.4.2 Définition du Problème

Le problème que nous traitons est lié à trois modalités : parole, geste (y compris déictiques), et navigation. Ces modalités sont fortement liées car elles se supportent les unes les autres au sein de la tâches, cf. Figure B.12. Le robot utilise la parole pour décrire une trajectoire de navigation que le visiteur doit suivre pour atteindre sa destination ; les gestes améliorent la parole en l'associant à des repères ou en décrivant les actions ; le robot peut naviguer avec l'humain pour se rapprocher de la cible, ce qui simplifie la description verbale, et le déplacement peut aussi permettre des gestes déictiques différents (meilleurs) en atteignant une perspective plus appropriée.

Ainsi, il y a une continuité entre la description d'itinéraire allant du point de départ à l'accompagnement du visiteur vers sa destination, y compris la navigation vers un bon *point de vue* pour fournir des indications. Pour résoudre cette tâche, il faut décider quelle solution choisir dans ce continuum. Nous présentons ici un

planificateur qui décide *où* se placer pour donner les indications de route, et quel point de repère sera indiqué par le guide, au regard d'une évaluation des coûts dédiés.

Le planificateur de perspective visuelle partagée (SVP Planner) que nous avons développé et présentons dans les sections suivantes considère une tâche qui est une séquence des étapes suivantes :

— le robot guide - accompagne physiquement - le visiteur à un endroit ;
— il pointe un objet ;
— il fournit des indications de chemin en basées sur l'objet pointé ;
— l'humain continue en ayant une meilleure connaissance de la route qu'elle doit emprunter pour atteindre sa destination ;

dans cet ordre, mais chaque étape étant optionnelle.

Le planificateur SVP choisit l'étendue de chacune de ces étapes en choisissant l'endroit où les directions seront fournies. Il recherche une position pour le visiteur et le guide, où celui-ci peut pointer un ou plusieurs points de repère, et d'où le visiteur peut atteindre sa destination en suivant un chemin avec une description simple fournie par le robot. Il est important de noter que ces quatre étapes sont prises en compte par le planificateur du SVP pour évaluer les solutions de la tâche dans son ensemble. Pour ce faire, il faut disposer d'informations appropriées sur la tâche, le partenaire et l'environnement.

B.4.3 Travaux Liés

B.4.3.1 Utilisation et sélection de repères pour les indications de chemin

[Daniel 2003] discute de l'usage de points de repères visuels dans les indications de chemin, et montre l'importance des propositions liant les actions à faire et ces points de repère (par exemple, “au parking, tournez à droite”). [Nothegger 2004], propose un système de sélection des points de repère basé sur leur pertinence et leur salliance. [Clark 1983] étudies comment les humains produisent et interprètent des références démonstratives (incluant le regard et le pointage) et l'utilisation du dialogue pour lever les ambiguïtés. [Allen 2000] proposes des recommandations pour le choix des indications de chemin.

B.4.3.2 Pointage

[Clark 2005] classifie les signaux physiques du dialogue situé en deux catégories : “diriger l'attention vers” et “placer l'objet pour capter l'attention”. [Sauppé 2014] présente une étude montrant la grande variété de mouvements déictiques et la nécessité de les adapter aux contextes physiques, environnementaux et de la tâche. [Holladay 2014] s'intéresse à la lisibilité des mouvements déictiques. [Allen 2003] montre que les mouvements déictiques sont fréquents et généralement liés à un élément du dialogue immédiat.

[Hato 2010] étudie les modèles cognitifs utilisés par les humains pour pointer des régions de l'espace et pour leur interprétation, et propose une implantation d'un tel modèle.

La synthèse d'un système fonctionnel utilisant d'une combinaison de parole et de gestes afin d'obtenir une référence déictique détaillée aux objets de l'environnement avec les humains a été discutée dans [Brooks 2006].

B.4.3.3 Placement pour partager la perspective visuelle

[Wong 2010] donne une analyse des éléments importants à la réussite d'un mouvement de pointage. Ils mentionnent le besoin pour l'observateur de voir en même temps le mouvement et l'objet, ainsi que la nécessité de maintenir le mouvement tant qu'il n'est pas clair que l'observateur ait bien saisi ce qui est pointé.

[Kelly 2004] met en avant le rôle de "l'espace visuel partagé" ainsi que les difficultés des humains à estimer la perspective des autres.

En ce qui concerne la planification du placement, il y a un certain nombre de résultats sur le placement de capteurs (par exemple [Chen 2004]) ou de planification de position pour un robot afin de partager l'attention et le point de vue avec un humain [Marin-Urias 2009], mais nous n'avons trouvé aucune contribution sur la recherche d'une perspective partagée pour l'humain et le robot à la fois.

Nous n'avons pas non plus trouvé de références sur la manière dont un locuteur et un observateur se positionnent et éventuellement se déplacent pendant des explications de chemin. C'est la raison pour laquelle une étude préliminaire a été menée dans le projet MuMMER, sur les intéractions humain-humain [Belhassein 2017].

B.4.3.4 Indication de Chemin

Les indications de chemin sont des indications fournies par le guide au visiteur pour qu'il atteigne la destination désirée. Elles sont principalement constituées par le dialogue, et sont souvent améliorées par des gestes, comme nous l'avons vu ci-dessus [Daniel 2003, Allen 2000, Allen 2003]. Les itinéraires sont communiqués avec succès aux visiteurs lorsqu'ils les comprennent et peuvent se rendre à leur destination.

Certaines études montrent que les humains utilisent des gestes pour représenter ce que le visiteur verra ou fera, et que de tels gestes facilitent la communication d'informations spatiales. Le haut-parleur peut représenter des informations spatiales depuis une *perspective de chemin* ou une *perspective d'ensemble*. C'est-à-dire soit en se projetant dans une visite mentale, ou bien sur une carte (vue du ciel). [Alibali 2005], par exemple, traite de l'usage des gestes dans la cognition et la communication spatiales. Les gestes aident les orateurs à réfléchir à l'espace et à décomposer leur discours en unités temporelles, ils améliorent aussi efficacement la communication de l'information spatiale. Ils font également état de la variabilité des productions de gestes selon les individus et les tâches.

Des systèmes capables de fournir des indications de directions ont été largement étudiés par T. Kanda et ses co-auteurs [Okuno 2009, Kanda 2010,

Matsumoto 2012, Satake 2015a, Morales Saiki 2011, Satake 2015b], ainsi que d'autre auteurs [Bohus 2014, Chen 2017]. Ils traitent tous un seul des aspects de notre tâche.

B.4.4 Modèle

B.4.4.1 Modèle Physique de l'Environnement

L'environnement est représenté en trois dimensions, il est utilisé pour la validation de la navigation et les calculs de visibilité.

B.4.4.2 Modèle Symbolique de l'Environnement

Dans notre implémentation, le modèle symbolique se limite à la liste des repères visuels à pointer et une évaluation de la complexité et de la durée des indications basées sur chacun de ces repères. Cependant, [Morales 2015] présente un modèle conçu pour cette tâche. Nous présentons dans le Chapitre~B.5 un composant similaire développé conjointement au SVP Planner dans notre équipe.

B.4.4.3 Modèle de l'Humain (visiteur)

Le guide doit s'adapter aux capacités hétérogènes des visiteur, pour que notre système soit accessible et ne soit pas discriminant envers certaines personnes en ignorant leurs spécificités. Il doit aussi pouvoir s'adapter à une variété de cas d'utilisation. Pour adapter les solutions, notre système utilise des informations liées à :

- vision :
- hauteur des yeux du visiteur, pour calculer sa perspective ;
- acuité visuelle, pour utiliser des repères mieux visibles ;
- capacités de déplacement (vitesse) pour pénaliser plus fortement les longues routes ;
- urgence, pour déterminer l'importance de la durée de la solution par rapport aux autres critères.

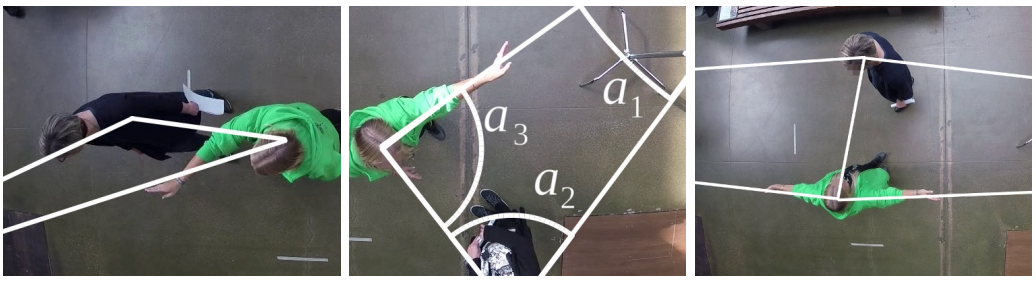
Ces informations sont prise comme entrées, elles doivent être acquises ou inférées par le dialogue et la perception.

B.4.4.4 Modèle du Robot (guide)

Le robot peut naviguer, le planificateur doit prendre en compte la distance maximale qu'il peut parcourir ; nous utilisons aussi une estimation de sa vitesse "de croisière". Notre approche peut prendre en compte des capacités de robots différents en ajustant des paramètres de pondération liés aux différentes étapes de la tâche.

B.4.4.5 Paramètres

Certains paramètres sont définis en fonction du domaine. Le guide peut se voir accorder un temps limité pour servir chaque visiteur ou, au contraire, pour aider



(a) (b) (c)

FIGURE B.13: Trois examples de pointage réel. La personne portant le haut vert (clair) est le guide pointant un repère (deux dans (c)) ; en blanc sont représenté les triangles guide-visiteur-repère. (Image : [Belhassein 2017])

chaque visiteur engagé autant que possible, *i.e.* fournir le maximum d'efforts pour résoudre une demande une fois qu'il est engagé dans une tâche.

B.4.5 Evaluation des Solutions

La décision est basée sur l'estimation et la comparaison de plusieurs possibilités de solutions à la tâche. Cette évaluation doit prendre en compte :

chances de succès : plus les indications sont simple, plus il y a de chances que l'humain s'en souvienne et atteigne sa destination ;

optimalité pour les visiteurs en termes de durée et d'effort ;

optimaliité pour le guide en fonction de ses objectifs globaux – servir le plus de personnes possibles ou aider le plus possible l'utilisateur en cours.

Notre solution évalue la conformation (les positions relatives du robot, de l'humain et de la cible à pointer, cf. Figure B.13) ; la navigation (durée, distance) ; et la complexité des indications.

B.4.6 Implémentation

Le problème est traité comme un problème d'optimisation non-linéaire sous contraintes. Les contraintes sont utilisées pour réduire l'espace de recherche, dans lequel une solution optimale est recherchée, après discrétisation du problème.

B.4.7 Conclusion

Nous développé un cadre original pour permettre à un robot de donner des indications de direction à des utilisateurs d'un espace public. L'implémentation a donné lieu à des expérimentations sur des environnements fictifs et réels, et à une intégration dans un système complet et fonctionnel, dans le cadre du projet MuMMER (voir section suivante).

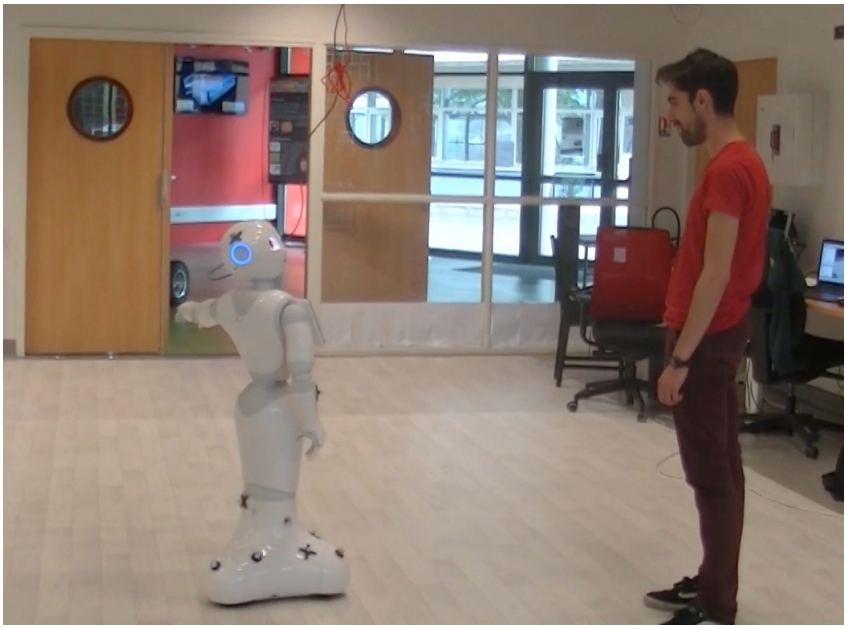


FIGURE B.14: Le robot indiquant un chemin dans l'environnement.

B.5 Le Robot Guide

B.5.1 Introduction

Dans le chapitre précédent, nous avons présenté un planificateur conçu pour fournir des indications de chemin, le robot pouvant pointer des points de repère, les utiliser lors des indications de chemin, et navigant avec l'humain vers des endroits où les repères sont visibles. Nous nous sommes concentrés sur la définition des tâches et sur le planificateur lui-même. Nous avons montré que le planificateur SVP est au cœur de cette résolution de tâches, mais qu'il dépend de l'information sur la route, sa description et les repères qui pourraient être utiles à pointer, ainsi que du modèle de l'environnement. De plus, l'implantation sur un vrai robot nécessite une perception de l'homme, de la localisation, de la navigation, du mouvement et un superviseur dédié.

Nous présentons l'intégration du planificateur SVP dans un système complet, dont les composants sont développées par certains de nos partenaires dans le projet MuMMER ou par des collègues. Nous présentons brièvement ces composants et nous nous concentrons sur l'architecture et les résultats des tests en laboratoire.

Les composantes que nous avons développées et intégrées à ce jour sont au nombre de six :

— Un module d'évaluation de la situation qui maintient ce que le robot et les humains savent sur la scène afin de raisonner sur les croyances ;
— Le Planificateur SVP qui calcule l'endroit où le robot et l'humain doivent être placés pour indiquer et décrire la route vers certains points de repère ;
— Un module de navigation ;

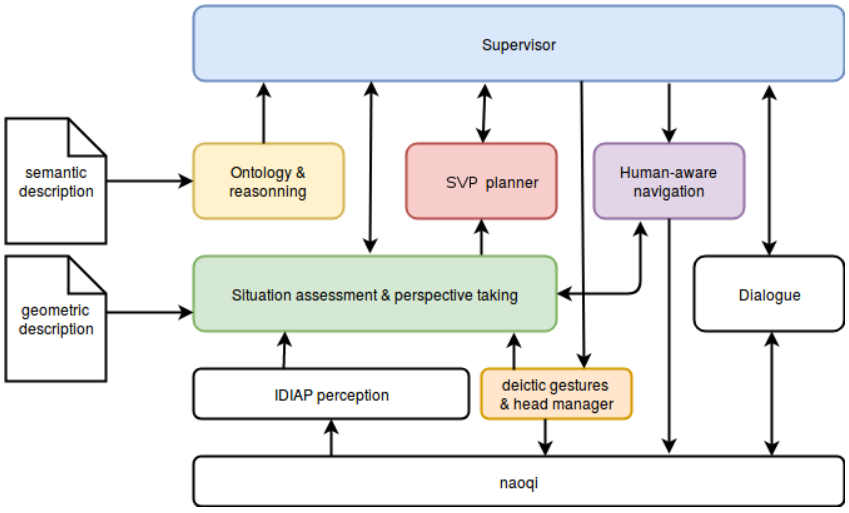


FIGURE B.15: L'architecture générale du système.

— Un module de description de route sémantique qui produit une description sémantique qui doit être verbalisée par le robot ;
— Un module de gestion de la tête pour gérer l'endroit où le robot doit regarder, selon le signal reçu des autres composants ;
— Un superviseur chargé de coordonner les modules ci-dessus, de surveiller l'interaction et de réparer ou de récupérer les problèmes d'exécution.

Ici, nous ne présentons volontairement pas la perception et les parties de dialogue car elles ont été traitées par d'autres partenaires du projet MuMMER. La tâche a été mise en œuvre dans un laboratoire adapté pour imiter un centre commercial. Cependant, l'architecture est assez générique pour être étendue à d'autres environnements.

B.5.1.1 Reconnaissances

L'architecture et l'intégration présentées dans ce chapitre sont des travaux d'équipe, et les différents composants autres que le planificateur SVP présenté précédemment sont développés sous la supervision de Dr Rachid Alami par leurs auteurs respectifs :

Navigation par Guilhem Buisan ;

Supervision par Amandine Mayima ;

Évaluation de Situation par Yoan Sallami ;

Modèle Sémantique et Ontologie par Guillaume Sarthou ;

Gestion de la tête et mouvement de Pointage principalement développé par Guilhem et Yoan.

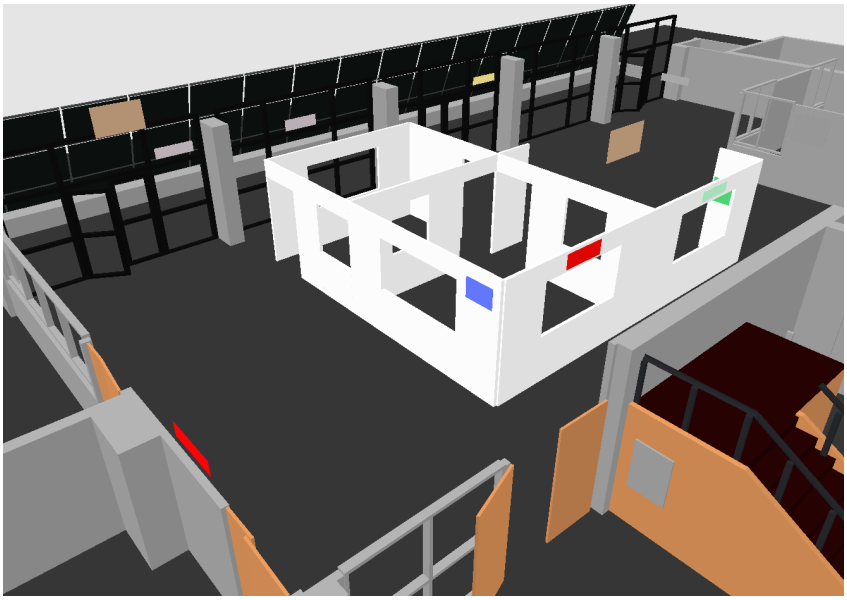


FIGURE B.16: Modèle 3D de notre laboratoire avec des panneaux ajoutés pour servir de repères

B.5.2 Architecture

Nous considérons la tâche comme étant une activité jointe, nous basons nos travaux sur des architectures pré-existantes dédiées à l'action jointe [Lemaignan 2017, Devin 2016]. Voir figure B.15.

B.5.3 Description des Composants

B.5.3.1 Modèles Symboliques et Géométriques

Nous utilisons un modèle géométrique (mailles 3D) de l'environnement pour les calculs de collision pour les mouvements et la navigation et pour les calculs de visibilité (voir figure B.16).

Les chemins sont calculés grâce à une représentation topologique, contenants des zones (régions), liées par des interfaces (portes, escaliers, ascenseurs,...) et des couloirs, comme illustré par la figure B.17. La topologie est gérée par une ontologie qui a également connaissance des magasins et de ce qu'ils vendent.

B.5.3.2 Description de Chemin

Une ontologie est utilisée pour représenter les liens sémantiques et spatiaux entre les éléments de l'environnement, ainsi que leur signification et comment ils peuvent être utilisé dans la tâche. Par exemple l'ontologie peut représenter qu'un panneau indique la direction vers un lieu, qu'un élément représente un escalier et ne peut donc pas être emprunté par certains individus. Ces informations sont utilisées pour choisir la meilleure route possible vers la destination, en prenant en compte

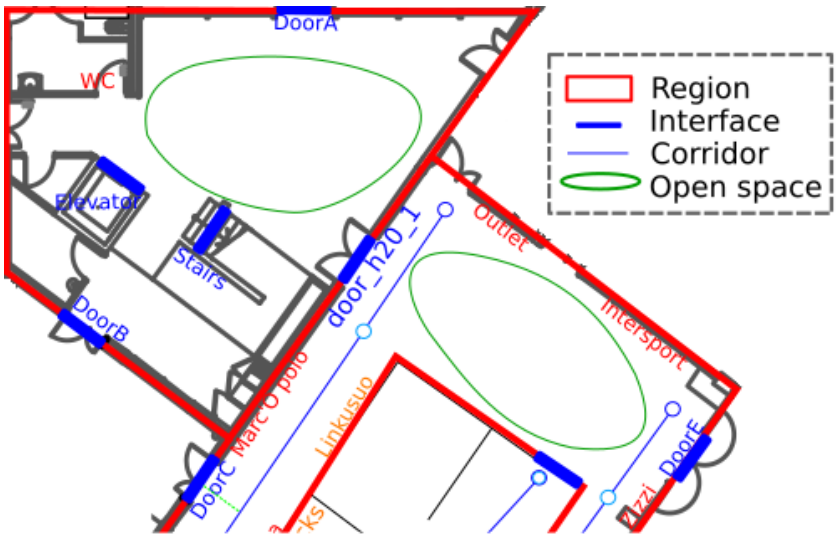


FIGURE B.17: Représentation du modèle sémantique de l'environnement utilisé pour la description de chemin.

la saillance des éléments, l'accessibilité de la route, le confort, l'explicabilité et la durée, et la sensibilité de la personne à chacun de ces critères. Une fois la route sélectionnée, l'ontologie est capable de fournir deux repères à utiliser pour pointer lors des explications : l'un représentant la destination elle même et l'autre le premier passage à emprunter pour l'atteindre, le cas échéant.

B.5.3.3 Évaluation de Situation

Ce composant, basé sur Underworlds [Lemaignan 2018], assure la collecte de données des modules de perception et produit en continu des information géométriques et symboliques sur l'environnement, permettant ainsi de raisonner sur le contexte. Il maintient également plusieurs modèles de connaissances représentant la théorie de l'esprit de plusieurs chaque agent. Chacun de ces modèle étant un modèle géométrique de la scène et un historique de prédicats, pouvant donc représenter les croyances du robot sur l'environnement et sur les croyances des humains (théorie de l'esprit de premier niveau).

B.5.3.4 Supervision

Ce composant assure l'orchestration des différents composants, la cohérence de l'interaction, notamment par le suivi de l'engagement de l'humain dans la tâche et la réparation de plan. Ce composant s'appuie sur tous les autres pour remplir ses fonctions de gestion de la tâche au plus haut niveau.

Session d'interaction Le module de supervision sélectionne l'humain avec qui interagir et quand se termine la session d'interaction. Durant la session, la personne

engagée peut discuter avec le robot, et le superviseur pour déclencher d'autres activité en fonction.

Tâche de description de chemin Ce composant contrôle l'ensemble de la tâche, en appelant les autre composants et en relayant les données entre eux.

La tâche est modélisée par des machines à états exécutées par SMACH [Bohren 2011].

Le flot typique d'exécution peut se résumer ainsi :

1. Sélectionne une route parmi celles produites par le composant de description de chemin pour la tâche en cours ;
2. Si une route est trouvée, le robot demande si la personne souhaite qu'elle lui soit expliquée ;
3. Si oui, le planificateur SVP est invoqué
4. Si le plan de SVP implique de déplacer l'humain, le robot lui indique où il doit se placer ; si besoin le robot se déplace aussi ;
5. Le superviseur s'assure que les positions correspondent au plan avant de donner les indications en pointant éventuellement les repères indiqués dans le plan du SVP.

Le superviseur s'assure notamment que chaque étape soit réussie avant de passer à la suite.

B.5.3.5 Navigation

Ce composant est un travail basé sur HATEB (Human-Aware Time Elastic Bands) [Khambhaita 2017], avec deux approches liées à notre tâche :

— **Approcher un humain** : le planificateur adapte dynamiquement sa cible quand l'humain se déplace (il se déplace souvent vers le robot) ;
— **Se déplacer vers un emplacement** : navigue dans l'environnement en prenant en compte les humains présents.

B.5.3.6 Gestion de la Tête

Le regard du robot est utilisé pour améliore l'attention, l'engagement et montrer l'attention [Mutlu 2006, Zaraki 2014, May 2015]. Dans notre scénario, il est utilisé pour améliorer la lisibilité des déplacements et diriger l'attention vers les repères pointés. Mais la tête sert à la fois pour ces signaux et pour la perception ; elle est donc une ressource partagée critique, nécessitant un composant dédié pour imposer une politique de partage. Le gestionnaire reçoit des requêtes de différents composants, et arbitre grâce à des informations émises par le superviseurs en fonction de la situation.

B.5.4 Expérimentations

Nous avons testé l'ensemble du système présenté dans notre laboratoire, dans un environnement reproduisant un centre commercial, avec des repères réels, dans des situations variées afin de démontrer les capacités et limites du système.

B.5.5 Perspectives

Le système présenté est entièrement fonctionnel. L'intégration de chaque composant au sein de ce système permet d'obtenir un retour direct sur les choix de conception et permet de les améliorer continuellement, en particulier pour le planificateur SVP afin qu'il évalue la tâche plus précisément.

Le besoin de partage de l'information nécessaire à l'accomplissement de tâches collaboratives avec l'humain est rendue possible par une intégration avancée des différents composants du système.

Nous continuons ces travaux dans le cadre du projet MuMMER.

B.6 Planification d'Actions pour des équipes Humain-Robot

B.6.1 Introduction

La plupart des tâches courantes, y compris celles étudiées dans cette thèse, se basent sur une recherche de positions intermédiaires avant de calculer les mouvements les reliant. Dans le cas d'un robot partageant l'espace et des tâches avec des humains, ces mouvements et actions doivent observer des propriété supplémentaires, qui dans nos travaux sont représentés par des fonctions objectifs à optimiser pouvant varier en fonction de la tâche et du contexte. De plus il peut être nécessaire de traiter chaque individu de manière spécifique (comme c'est le cas dans nos deux contributions principales).

Notre idée est que dans ce contexte, il est important pour tous les planificateurs, du symbolique au géométrique, de partager des informations sur la tâche planifiée, afin de prendre les meilleures décisions au bon niveau d'abstraction, avec des algorithmes et fonctions objectifs appropriés.

Le projet initié par Mamoun Gharbi sous la supervision de Rachid Alami, le Planificateur Géométrique de Tâches (GTP) [Gharbi 2015b], un outil reliant la planification symbolique et géométrique, est à l'intersection des deux espaces de planification. Nous avons développé une fonctionnalité qui configure les algorithmes de planification des mouvements et les étapes de raffinement en fonction de la tâche en cours d'exécution.

Ce chapitre présente d'abord les bases de GTP, puis montre comment nous l'utilisons pour planifier des tâches collaboratives.

B.6.2 Concepts

B.6.2.1 Actions

Une action est définie par de symboles comme dans les planificateurs de tâches [Ghallab 2016, Lozano-Perez 1987, Alami 1998]. L'objectif est rarement connu sous forme d'un état entièrement définit, mais est plutôt un ensemble de propriété et relations symboliques à atteindre, qui peuvent être représentées comme des contraintes (souvent non-linéaires) dans l'espace de travail cible.

Nous découpons ces actions en étapes élémentaires pouvant être représentées comme un problème de planification de mouvement [LaValle 2006], ce qui nécessite de trouver des états initiaux et cible pour chaque sous-action.

B.6.2.2 Action Request (input)

Nous définissons une façon de décrire une action de manière générique, afin d'avoir un format de requête uniforme ; il est formé uniquement de symboles. Les éléments de la requête sont les suivants :

— Type d'action : la catégorie d'action que nous voulons calculer

— Agent principal : l'agent exécutant l'action
— Agent(s) cible(s) : autres agents à prendre en compte
— Objet principal : objet manipulé
— Objet(s) de support : les supports que nous utilisons (p. ex. pour les placements)
— Objet cible : une cible pour les actions de navigation
— Bras : bras, main ou gripper de l'agent principal à utiliser
— Coordonnées de la cible
— Contraintes sur l'état final : faits symboliques qui doivent être observés à l'état final (voir ci-dessous)

Chaque type d'action peut utiliser un sous-ensemble de ces paramètres, certains pouvant être optionnels (et donc le planificateur est libre de les choisir).

la planification se base sur un état du monde entièrement défini, qui peut être un état courant (obtenu via des senseurs) ou un état futur planifié.

B.6.2.3 Solution d'Action (sortie)

Une solution d'Action est un mouvement modifiant l'état du monde pour satisfaire les objectifs de la définition de l'action. GTP est capable de diagnostiquer certaines causes d'échec lors de la planification, et calcule également des faits sur l'état final de l'action.

B.6.2.4 Faits

Les faits sont des prédicats exprimant des relations entre les objets ou les agents dans un état du monde donné. Ils sont utilisés pour définir les effets symboliques des actions, pour informer les niveaux de planification supérieurs. Ils peuvent également être donnés comme contraintes sur l'état final d'un action, par exemple pour préserver les effets de précédentes actions. GTP peut résoudre une tâche en respectant des contraintes additionnelles sous formes de faits.

B.6.2.5 Planification en Séquence

GTP peut résoudre une séquence d'actions. Dans ce cas GTP peut réévaluer (*backtrack*) des actions déjà résolues pour rendre le plan faisable ou l'améliorer.

B.6.2.6 Modèles

GTP est pensé comme un système générique, et la création de nouveaux domaines et l'utilisation avec de nouveaux robots sont censés être des tâches simples. Nous utilisons des modèles URDF pour les robots, et des fichiers textes contiennent les informations sur les éléments de planifications disponibles (bras, etc. . .), ainsi que les positions de saisies d'objet ou les surfaces sur lesquels ils peuvent être posés.

B.6.3 Algorithme

Nous abordons dans la version complète des détails d'implémentation de GTP, notamment la résolution de tâche, la réévaluation (*backtrack*), la recherche basée sur des coûts et leur utilisation dans une planification de tâche collaborative Humain-Robot, le calcul d'états intermédiaires pour les tâches de manipulation, et les calculs de faits.

B.6.4 Planification de Mouvements pour l'Action Joint Humain-Robot

GTP a été conçu pour résoudre des tâches avec d'importantes dépendances entre les niveaux symboliques et géométriques, avec plusieurs agents, dont des humains pouvant être impliqués dans le plan partagé. Les humains n'étaient pris en compte que comme des agents pour lesquels ont peut planifier, et finalement les mouvements sont générés en prenant en compte les préférences et la sécurité de l'humain.

Nous avons contribués à GTP sur les fonctionalités suivantes :

— optimisation des mouvements
— optimisation des motions entre 2 états, mais aussi des 2 états
— choix de l'objectif à optimiser en fonction du contexte.

B.6.4.1 Planification de Mouvement Optimal

Les mouvements sont calculés avec des algorithmes standards de, pouvant prendre en compte l'humain [Mainprice 2011] lorsque nécessaire.

B.6.4.2 Amélioration par de Meilleures Requêtes

Dans ces travaux, le planificateur de mouvement ne peut pas optimiser les états origine et cible, ils lui sont imposé par GTP. Il est donc nécessaire de les choisir avec attention. Nous avons rendu cela possible en échantillonnant plusieurs états intermédiaire et en les sélectionnant avec des fonctions objectifs reflétant les propriétés attendues des mouvements. Mais ces fonctions ne sont pas nécessairement celles utilisées lors de la planification de mouvement car ici le choix se fait à un autre niveau (par exemple le choix des états intermédiaire peut amener à utiliser telle où telle partie (bras) du robot, ce qui n'a pas à être pris en compte lors de la planification de mouvement).

GTP créé de meilleures solutions aux actions de manipulation en choisissant les parties de robots à utiliser, comment attraper les objets, en utilisant un planificateur de mouvement centré sur l'humain.

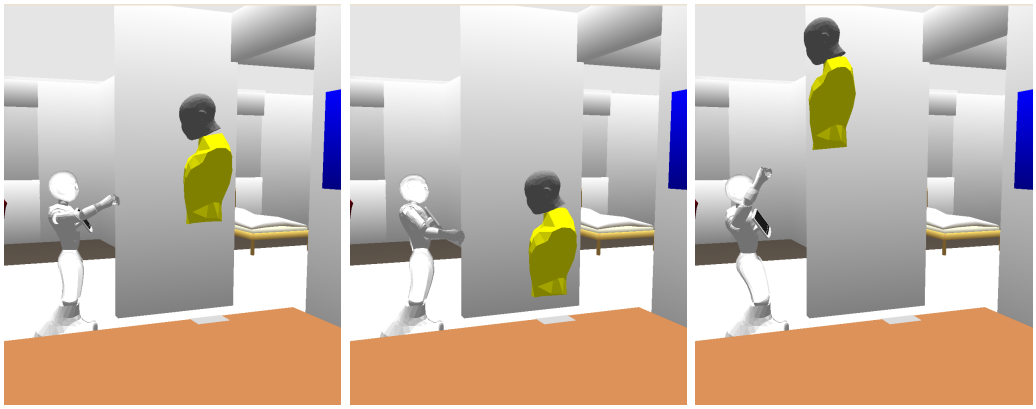
B.6.4.3 Fonction Objectif dépendante de la Tâche

L'utilisateur peut configurer les fonction objectifs à utiliser pour chaque étape de chaque action, car le comportement du robot doit être adapté à la tâche, et n'est pas nécessairement le même dans chacune.

B.6.5 Conclusions et Perspectives

Nous montrons dans ce chapitre comment GTP a été modifié pour améliorer sa prise en compte du contexte pour générer des plans plus acceptables.

Nous pensons que GTP est un outil versatile qui pourrait être augmenté par de nouveaux types de tâches, en commençant pars les deux solveurs de tâches présentés dans cette thèse.



(a) (b) (c)

FIGURE B.18: Exemple d'adaptation d'un mouvement intéractif. La position initiale est (a). Dans (b) et (c) la position est adaptée à un humain de taille différente à un emplacement différent. Cette méthode est adaptée pour des changement locaux ; si la situation est trop différente, une replanification plus complète est requise.

B.7 Conclusion : Vers des Tâches Collaboratives Humain-Robot Fluides

B.7.1 Introduction

Nos travaux présentés dans cette dissertation se concentrent sur la construction de systèmes ou de composants rendant les robots capables de répondre à des situations où ils doivent *planifier* pour des *tâches collaboratives*, en utilisant la théorie de l'*action jointe* pour construire des *plans partagés*.

Durant la conduite des travaux présentés dans cette thèse, nous avons rencontré un certain nombre de problèmes liés, et nous nous sommes intéressés à certains. Ils concernent principalement la planification de mouvement pour la collaboration humain-robot, nécessitant la prise en compte d'incertitudes, de la dynamique, et de multiples objectifs d'optimisation.

B.7.2 Prise en Compte de l'Incertitude par l'Adaptation en Temps-Réel

L'environnement évoluant rapidement, il est nécessaire de s'adapter aux changements pouvant avoir lieu entre la planification et l'exécution. Trois approches se distinguent : la re-planification, la planification en continu, et l'adaptation locale de solutions.

Nous avons notamment commencé le développement d'un solution inspirée par [Ho 2010] utilisant des relations spatiales pour adapter des mouvements à des environnements ou des acteurs différents ou aillant changé par rapport au mouvement initial. Un exemple de résultat préliminaire est montré par la figure B.18.

Contrairement aux approches courantes adaptant un mouvement en continu, nous avons aussi cherché à rendre le système capable de décider quelle méthode utiliser pour s'adapter à un changement dans l'environnement. Nous avons commencé à développer un outil capable d'évaluer les changements aillant eu lieu par rapport à l'environnement de planification, et de choisir un comportement parmi :

— continuer le mouvement en cours ;
— adapter localement le mouvement ;
— replanifier en réutilisant le matériel de planification (graphes de RRT,...) ;
— planifier à nouveau à partir du nouvel état du monde.

Cette approche tente d'utiliser le plus d'information (géométrique et symbolique) disponible possible, et se base naturellement sur GTP. Nous avons implémenté une notion de distance entre deux états du monde et nous utilisons cette métrique pour choisir quoi faire.

B.7.3 Conception des Coûts

La plupart des algorithmes de planification de mouvement pour l'interaction humain-robot sont basées sur des coûts, des fonctions objectifs à optimiser. La difficulté réside dans le fait que chaque coût reflète une propriété du mouvement et que l'assemblage de ces coûts n'est pas aisé. Ce problème est connu sous le nom d'optimisation multi objectifs (ou multi critères) et est étudié en économie comme en ingénierie. [Marler 2004] propose une revue et une discussion sur les approches de l'optimisation multi-critères.

Nous avons également établi que bien souvent, plutôt que d'établir des agrégations de coûts risquant de ne pas refléter la somme de propriétés souhaitées, il peut être pertinent de modéliser sous forme de contraintes. Cette approche a aussi l'avantage de rendre la planification plus simple en rejetant les portions de mouvement au plus tôt lors de la planification.

Abstract: When interacting with humans, robotic systems shall behave in compliance to some of our socio-cultural rules, and every component of the robot have to take them into account. When deciding an action to perform and how to perform it, the system then needs to communicate pertinent contextual information to its components so they can plan respecting these rules. It is also essential for such robot to ensure a smooth coordination with its human partners. We humans use many cues for synchronization like gaze, legible motions or speech. We are good at inferring what actions are available to our partner, helping us to get an idea of what others are going to do (or what they should do) to better plan for our own actions. Enabling the robot with such capacities is key in the domain of human-robot interaction.

This thesis presents our approach to solve two tasks where humans and robots collaborate deeply: a transport problem where multiple robots and humans need to or can handover an object to bring it from one place to another, and a guiding task where the robot helps the humans to orient themselves using speech, navigation and deictic gestures (pointing). We present our implementation of components and their articulation in a architecture where contextual information is transmitted from higher levels decision components to lower ones, which use it to adapt. Our planners also plan for the human actions, as in a multi-robot system: this allows to not be waiting for humans to act, but rather be proactive in the proposal of a solution, and try to predict the actions they will take.

Résumé : Les robots interagissant avec des humains doivent se comporter en adéquation avec certaines de nos règles sociaux-culturelles, qui doivent être considérées par chaque composant du robot. Lorsqu'il décide d'une action à faire et de comment l'exécuter, le système a besoin de communiquer l'information contextuelle pertinente à chacun de ses composants afin qu'ils puissent respecter ces règles. Il est essentiel que de tels robots puissent se coordonner sans accrocs avec leur partenaires humains. Nous humains utilisons de nombreux signaux de synchronisation notamment via le regard, la lisibilité de nos gestes ou par le dialogue. Nous inférons efficacement les possibilités d'actions de nos partenaires, ce qui nous aide à anticiper ce qu'ils vont ou devraient faire afin de mieux planifier nos propres actions. Dans le domaine de l'interaction Homme-robot, ces capacités sont essentielles.

Cette thèse présente notre approche pour résoudre deux tâches où humains et robots collaborent étroitement: un problème de transport d'objet où plusieurs robots et humain doivent ou peuvent se faire passer un objet de main à main pour l'amener d'un endroit à un autre, et une tâche de guide où le robot aide des humains à s'orienter en utilisant dialogue, navigation et mouvements déictiques (pointage). Nous présentons notre implantation de ces composants et de leur articulation dans le cadre d'une d'architecture où l'information contextuelle est transmise des plus hauts niveaux de décision vers les plus bas qui l'utilisent pour s'adapter. Le robot planifie aussi pour les actions des humains, comme dans un système multi-robot, ce qui lui permet de ne pas être dans l'attente des actions des humains, mais d'être proactif dans la proposition d'une solution, et d'anticiper leurs actions futures.

Keywords: motion planning, task planning, human-robot interaction

Mots clés : planification de mouvement, planification de tâches, interaction Homme-robot