Institut für Informatik
Technische Universität München

Planning for Human Robot Interaction

Dissertation

Thibault Kruse

TECHNISCHE UNIVERSITÄT MÜNCHEN und
UNIVERSITÉ TOULOUSE III PAUL SABATIER

Lehrstuhl für Informatik IX
Intelligente Autonome Systeme

Planning for Human Robot Interaction

Thibault Kruse

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. Daniel Cremers
Prüfer der Dissertation:	1. Univ.-Prof. Michael Beetz, Ph.D., Universität Bremen
	2. Univ.-Prof. Dr.-Ing. Darius Burschka
	3. Prof. Dr. Mohamed Chetouani, UPNC Paris / France
	4. Dr. Christian Laugier, INRIA Grenoble / France

Die Dissertation wurde am 04.02.2014 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.07.2014 angenommen.

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier) et la Technische Universität München

---

Présentée et soutenue le *14/01/2015* **par :**

THIBAULT KRUSE

Planning for Human Robot Interaction

JURY

PROF. DR. DANIEL CREMERS	Professeur d'Université	Président du Jury
PROF. DR. DARIUS BURSCHKA	Professeur	Membre du Jury
PROF. DR. MOHAMED CHETOUANI	Professeur	Membre du Jury
DR. CHRISTIAN LAUGIER	Directeur de Recherche	Membre du Jury
PROF. DR. MICHAEL BEETZ	Professeur d'Université	Membre du Jury
PROF. RACHID ALAMI	Directeur de Recherche	Membre du Jury
DR. ALEXANDRA KIRSCH	Maitre de Conference	Membre du Jury
PROF. PATRICK DANÈS	Professeur d'Université	Membre du Jury

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle
Unité de Recherche :
Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)
Directeur(s) de Thèse :
Dr. Alexandra KIRSCH et *Prof. Rachid ALAMI*
Rapporteurs :
Prof. Dr.-Ing. Darius Burschka, *Univ.-Prof. Dr. Mohamed Chetouani* et *Dr. Christian Laugier*

Contents

Abstract

The recent advances in robotics inspire visions of household and service robots making our lives easier and more comfortable. Such robots will be able to perform several object manipulation tasks required for household chores, autonomously or in cooperation with humans. In that role of human companion, the robot has to satisfy many additional requirements compared to well established fields of industrial robotics.

The purpose of planning in robotics is to achieve robot behavior that is goal-directed and establishes correct results. But in human-robot-interaction, robot behavior cannot merely be judged in terms of correct results, but must be agreeable to human stakeholders. This means that the robot behavior must suffice additional quality criteria. It must be safe, comfortable to human, and intuitively be understood. There are established practices to ensure safety and provide comfort by keeping sufficient distances between the robot and nearby persons. However providing behavior that is intuitively understood remains a challenge. This challenge greatly increases in cases of dynamic human-robot interactions, where the actions of the human in the future are unpredictable, and the robot needs to constantly adapt its plans to changes. This thesis provides novel approaches to improve the legibility of robot behavior in such dynamic situations. Key to that approach is not to merely consider the quality of a single plan, but the behavior of the robot as a result of replanning multiple times during an interaction. For navigation planning, this thesis introduces directional cost functions that avoid problems in conflict situations. For action planning, this thesis provides the approach of local replanning of transport actions based on navigational costs, to provide opportunistic behavior. Both measures help human observers understand the robot's beliefs and intentions during interactions and reduce confusion.

Résumé

Les avancées récentes en robotique inspirent des visions de robots domestiques et de service rendant nos vies plus faciles et plus confortables. De tels robots pourront exécuter différentes tâches de manipulation d'objets nécessaires pour des travaux de ménage, de façon autonome ou en coopération avec des humains. Dans ce rôle de compagnon humain, le robot doit répondre à de nombreuses exigences additionnelles comparées aux domaines bien établis de la robotique industrielle.

Le but de la planification pour les robots est de parvenir à élaborer un comportement visant à satisfaire un but et qui obtient des résultats désirés et dans de bonnes conditions d'efficacité. Mais dans l'interaction homme-robot (HRI), le comportement robot ne peut pas simplement être jugé en termes de résultats corrects, mais il doit être agréable aux acteurs humains. Cela signifie que le comportement du robot doit obéir à des critères de qualité supplémentaire. Il doit être sûr, confortable pour l'homme, et être intuitivement compris. Il existe des pratiques pour assurer la sécurité et offrir un confort en gardant des distances suffisantes entre le robot et des personnes à proximité. Toutefois fournir un comportement qui est intuitivement compris reste un défi. Ce défi augmente considérablement dans les situations d'interaction homme-robot dynamique, où les actions de la personne sont imprévisibles, le robot devant adapter en permanence ses plans aux changements. Cette thèse propose une approche nouvelle et des méthodes pour améliorer la lisibilité du comportement du robot dans des situations dynamiques. Cette approche ne considère pas seulement la qualité d'un seul plan, mais le comportement du robot qui est parfois le résultat de replanifications répétées au cours d'une interaction. Pour ce qui concerne les tâches de navigation, cette thèse présente des fonctions de coûts directionnels qui évitent

les problèmes dans des situations de conflit. Pour la planification d'action en général, cette thèse propose une approche de replanification locale des actions de transport basé sur les coûts de navigation, pour élaborer un comportement opportuniste adaptatif. Les deux approches, complémentaires, facilitent la compréhension, par les acteurs et observateurs humains, des intentions du robot et permettent de réduire leur confusion.

Chapter 1

Introduction

Imagine a robot acting as a butler or assistant, an intelligent being that performs daily chores in the household. This vision has driven robotics research for decades, with many challenges to solve. And so each generation of robotics researchers produces ever more capable robots. In contrast to industrial robots, household robots will interact with humans in numerous ways. So household robot platforms need to be adapted to the special needs of human robot interaction (HRI).

One benefit of that introduction of robots into homes is an improved quality of life for disabled people, such as allowing them to live more independently in their homes. Another benefit is the ability to provide services less expensively through robots and thus make those services available to a larger group of people.

A robot is a machine that can perceive its environment, move around in the environment, and perform further manipulation actions. Many robot models are already in productive use at the time of this thesis, in industrial settings like factories or warehouses. Yet those robots are not suitable for acting around humans in domestic environments.

Contemporary examples of research platforms usable in HRI are ASIMO [124], ARMAR3 [4], HRP2 [68] and PR2 [145]. They offer a roughly humanoid design which allows suitable operation in environments created for humans. This shape also allows a large diversity of actions, as opposed to robot designs that are intended for single purposes.

The difference of assistive Human-Robot Interaction (HRI) to other robotics research domains is that the main purpose of the robot is to act around humans in a human-centric environment, as opposed to factory settings. Since the constraints of autonomous action in cooperation with humans are not well researchable without observing humans and robots acting together, the process of creating human-aware robots is an iterative one. Starting with robots and approaches that have successfully been deployed in other domains, limitations are observed, leading to improvements, as a starting point for further experiments. This leads to structural adaptations in robot hardware and software.

In hardware, a main advance towards human-friendliness of robots is the introduction of compliant actuators, meaning arms that do not respond stiffly to contacts. Another advance in hardware are new sensors like the Kinect [69] camera which allows real-time motion capture at little costs.

In software, a similar adaptation is required from rigid sequences of planning and execution to frequent and reactive reconsideration of plans under the influence of human actions. All research on improving planning algorithms for HRI need to take into account this form of software compliance to human cognition.

1.1 Example Scenario

Here is a possible scenario that shows how a robot can act as an assistant in a household. The example is produced to show the necessity of plan-driven behavior to make decisions. For now, consider planning as the act of thinking several steps into the future to make sure the next action leads towards a long-term goal.

This scenario describes a robot named B21 working in a household, while humans are present. In this scenario the robot will help with kitchen chores. B21 has a mobile base, and two arms with grippers to pick up certain items. B21 is also equipped with visual sensors to identify items and persons, as well as a communication interface. The household belongs to a fictional couple named John and Jane. In the situation of the scenario, John and Jane want the help of B21 with setting a table for lunch, while food is also being prepared in the kitchen. This is

a regular activity, that still has a lot of variation from day to day. Setting a table in itself requires planning, as the table potentially needs to be emptied from items that do not belong on a table for lunch, cleaned, a tablecloth then needs to be placed, and after that the dishes required for lunch have to be put on the table in the right places. It would not be valid or helpful if B21 started by placing dishes on the table before the other prerequisites are satisfied. If B21 was left to do all the tasks by itself in the absence of John and Jane, the only challenge would be to achieve the desired state as efficiently as possible. With John and Jane acting and deciding individually at the same time, the challenge for B21 is to help John and Jane with those activities, without discomforting them, and blending in with their own actions and decision making.

As a concrete situation, imagine John is preparing sandwiches, while Jane sets the table. Jane says to B21 "Please help me set the table for me and John, we will have sandwiches". Ideally this information is sufficient such that the table can be set without requiring further details to be communicated explicitly. To start the scenario, there is a book on the table and a flower bouquet. For the sake of the scenario, assume that B21 cannot move the flower bouquet from the table because it could easily damage the flowers, but B21 is capable of picking up the book. So B21 picks up the book, while Jane picks up the flowers. The next two steps are to clean the table and put a tablecloth on it. Jane might in this case pick up a sponge to clean the table. In the meantime, thinking ahead, B21 can go fetch the tablecloth, even if the table is not cleaned yet. This coordination of activities does require a plan for B21 to know it is valid and useful to fetch a tablecloth at that time. At a lower level, both B21 and Jane have been moving around in the household. The minimal solution for B21 to get to a goal is to approach the goal while evading obstacles. However, with Jane also acting in the household, B21 needs to consider where Jane will be moving, and choose a path and poses that do not unnecessarily obstruct the paths Jane takes. This also requires thinking ahead, and thus planning. The scenario continues with Jane setting the table and B21 picking up dishes to put on the table. B21 has to consider several things. Maximum efficiency is not required, but wasting time may cause annoyance to John and Jane. So B21 may consider picking up more than one item at a time to bring from the kitchen to the table. At the same time, John and Jane are acting, John is preparing sandwiches in the kitchen and requires kitchen tools and access

to food items in the process. So B21 ideally observes John, thinking ahead what locations in the kitchen are best to pick up items without disturbing the workflow of John, and choosing to pick up items accordingly. This process is very unpredictable, as John may move unpredictably in the kitchen, spontaneously deciding what items to use next. So B21 must at any given time be prepared to change its mind and pick up a different item next, or to stand and wait until John moves on from a given location.

This scenario shows how a robot may provide help with daily activities of humans. To expand the scenario, John and Jane can be imagined to be either physically or mentally challenged, to increase the value of the robot assistance. The scenario only shows challenges related to avoiding discomfort to humans. But many other new challenges arise when cooperating with humans, as an example, B21 could attempt to set the tablecloth together with Jane, requiring both dexterity and interaction. Also Jane might have specific wishes for B21, requiring more communication, which in itself poses many challenges, such as the communication medium. The scenario does not include such challenges as this thesis focuses on the impact of HRI on the planning algorithms.

1.2 Human-robot interaction

Human-robot interaction (HRI) combines robotics and psychological research. Several surveys exist on social robots and human-robot interaction [31, 35, 48, 66]. As Feil-Seifer [31] puts it: "Human-robot interaction (HRI) is the interdisciplinary study of interaction dynamics between humans and robots." There are many possible types of interaction between robots and humans, which depends largely on the robot design and capabilities.

This thesis uses a very broad concept of interactions describing any change in a human or a robot as a consequence of the presence of the other. So a human being distracted by a present robot is already an interaction. In this thesis the distinction between explicit and implicit interaction is made: Explicit interaction happens when an action is made for the purpose of achieving a reaction by the partner; implicit interaction happens when actions are modified due to the (potential) presence of a partner, but no reaction is intended.

In robot action planning, explicit interaction happens during dialog, or when an object is being handed over between a human and a robot. In navigation, explicit interaction may happen for guiding missions. Implicit interaction happens all the time, whenever the robot is navigating or manipulating in the presence of humans, or even just standing in a special location. This thesis deals mostly with issues related to implicit interaction, in order to improve the robot task execution behavior such that present humans are not discomforted by the robot decisions.

1.2.1 Compliant interaction

A general definition of compliance is: "Compliance refers to the act of responding favorably to an explicit or implicit request offered by others." [24]. The concept of compliance can be applied to robots as a whole or to just parts of robots. Compliant robot arms do not react stiffly to contact, as an example. The contact can then be seen as an implicit request. In a household setting, the first requests to a robot that come to mind are explicit requests. The robot may be requested to clean a room, fetch an item, etc. Thus robot behavior is usually implemented this way:

1. A robot is given some task to execute
2. The robot collects the necessary information to select a plan
3. The robot finds a plan to execute the task
4. The robot executes the plan to achieve the task

This approach follows the philosophy of the Sense-Plan-Act (SPA) cycle that has been a common approach in robotics in static domains. There is nothing wrong with the SPA cycle in its general form as a description of data flow. However as such it only considers explicit requests to respond to by acting. But during plan execution, humans have many implicit requests to the robot, such as not being disturbed or distracted while they also perform activities of their own. To comply with implicit requests a robot has to act safely, legibly, efficiently and comfortably adapting all the time to the environment.

As a special challenge in HRI, a robot needs to observe humans in its environment, their visible position and their state. There is no time where the robot can ignore the human. At any time what the human does, even if the human does not move or speak, is relevant to interaction, and this means it must have an impact on the robot behavior. What is needed in HRI are architectures that appropriately react to all perceptions, even the event of nothing happening, in a way that is comfortable to the human. This is compliant interaction, because only this way act favorably to the implicit wishes of humans for comfort.

Planning is one approach to control robot behavior that is adequate for several problems typically arising in human environments. Planning means in simple terms to think several steps ahead before acting. Planning does not solve all control challenges of robotics or HRI, only some of those. While planning is already well researched as a topic of its own, the particular challenges in HRI are the relative lack of structure and robot knowledge in human environments, and the high degree of unpredictable dynamics when humans act in the same environment as the robot.

1.3 Planning for robotic behavior

The aim of this thesis to improve the planning process of robots in human environments, so what do we regard as "planning"? Planning solves the problem of short-term action selection for a long-term goal. If a robot has to achieve some goal that cannot be achieved in a single action, selecting the next action to take requires reasoning several steps ahead, such that the next action is a step towards the long term goal. Planning is thus one approach for problem solving, where a problem is a situation for an agent where a desired state of affairs cannot be achieved (sufficiently well) in an already known procedure.

In planning the agent performs actions virtually on a mental model of the situation, searching for a procedure that in the model leads to the desired state of affairs. If a procedure is found, the robot may then execute the procedure, with a high degree of confidence that by doing to so the desired state will be achieved in the real world. Besides planning, other forms of behavior control exist for robots.

There are learning-based techniques, and purely reactive approaches. This thesis however focuses only on planning to control behavior.

The result of planning is usually called a plan. The format of a plan can vary widely, so we'll for now just say that a plan is something that a robot can use to quickly and correctly select the next small action to perform. This way, a long-term goal can be achieved without the robot having to spend a lot of time after each action to reason about what actions are valid or not given the long-term goal. As long as a plan remains valid, it can be followed without additional planning.

Most autonomous robots use several separate planning algorithms to achieve individual purposes, to perform base motions, arm motions, communications, and so on. This thesis focuses on HRI planning algorithms to solve indoor navigation problems under social constraints, and action selection problems in the presence of humans.

1.4 Challenges when planning for HRI

There are specific challenges when planning in HRI that need to be solved to achieve acceptable robot behavior when using planning techniques.

- HRI happens in worlds with unpredictable dynamics
- Human-centered behavior quality required
- Planning with interactive actions
- Timely responsible are essential

The first challenge of unpredictable dynamics arises in comparison to more traditional domains for robots. In risk management, one kind of dynamics are called "contingencies", those are propositions about the future that can become true or false. A contingency plan is a plan for the case that something goes wrong. Hence a contingency represents knowledge of uncertainty about the future. However in HRI, dynamics do not merely refer to things that could go wrong, but many things that can merely change, requiring just a slight adaptation to the behavior to behave compliantly.

The second challenge is about defining the desired properties of plans when a robot should act around humans. Mere energy efficiency seems like an inadequate measure for which planning should optimize plans. Instead, the comfort and well-being of humans need to be considered.

While humans often move and act in ways that are difficult for a robot to predict, human actions are of course not random. Instead, human actions are mostly goal-directed, and humans are usually able to spontaneously create a team to cooperate jointly on a goal. So a challenge is how to turn a robot into a suitable partner for such interaction.

The last challenge dictates that robot response times to events must be small. Any large delay between an event and the robot reaction decreases the acceptability of the robot. This is also true without the social aspect, as Gat puts it: "A mobile robot must operate at the pace of its environment. (Elevator doors and oncoming trucks wait for no theorem prover.)" [41]. But with human stakeholders near the robot, there is a less obvious necessity of the robot to react quickly to events, that of human comfort. A robot acting a person, must avoid long stretches of time where it appears stalled or performing wasteful actions, in order not to draw attention unnecessarily. This challenge is somewhat opposed to that of the necessity of plan-driven behavior, because planning as an activity is known to scale badly. In many domains outside HRI, unexpected changes are less frequent, less diverse, and a timely reaction is usually only required where an obvious temporal constraint exists in the environment. So HRI implies a stronger necessity to deal with unexpected changes in the world.

The following chapters analyze these challenges in more depth and show how planning techniques HRI can be improved to generate more compliant robot behavior.

1.5 Contributions Overview

The aim of this thesis is to advance robotic task and navigation planning towards enabling robots to cooperate with humans in scenarios such as given in the introduction section 1.1. Enabling robots to do so still requires research and

improvements in a lot of areas, this thesis only looks at research in planning and plan execution.

The main problem for planning in HRI identified in this thesis are the unpredictable dynamics of human actions. Unless sleeping or severely handicapped, humans must be assumed to always be active in one way or the other. While planning is essential to goal-directed behavior, lengthy planning and inflexible plan execution generate robot behavior that is confusing and wasteful in dynamic situations. So the research done for this thesis focuses on specific dynamic situations, and how common planning approaches fail to produce behavior that is acceptable in HRI.

The key aspect of robot legibility to be improved by the contributions of this thesis is legibility of robot behavior in dynamic situations. Legibility means that the internal state of the robot (e.g. its beliefs, intentions and plans) that are relevant to its behavior can be easily and early estimated from mere observation of the robot. In contrast to other works, this description of legibility is not restricted to estimating the immediate intention of the robot. Very often in HRI, a human stakeholder does not need to know what the robot will do ahead of time, but rather what things the robot will *not* do. So it is important for observers to know the robot will not next crash into a wall, fall down stairs, or hurt a person. Correspondingly observers have questions like: "Does the robot see me? Will the robot hurt me?". So the primary goal of legible behavior should be to dissipate fears and doubts about the robot, a secondary goal is to allow joint planning. For the secondary goal early estimation is important so that persons can adapt to what the robot intends to do.

Figure 1.1 indicates the modules affected by results of this thesis. It is based on a sketch of robot software architectures explained in more detail in Figure 2.7. The figure shows functionalities a robot typically requires to operate, grouped by similarity. Functionalities closer to the top operate on abstracted knowledge, functionalities to the bottom on the mass of raw sensor data and motor commands.

Two areas of planning for HRI are affected, action selection and navigation. In action selection, chapter 4 shows improvements to planned robot behavior when

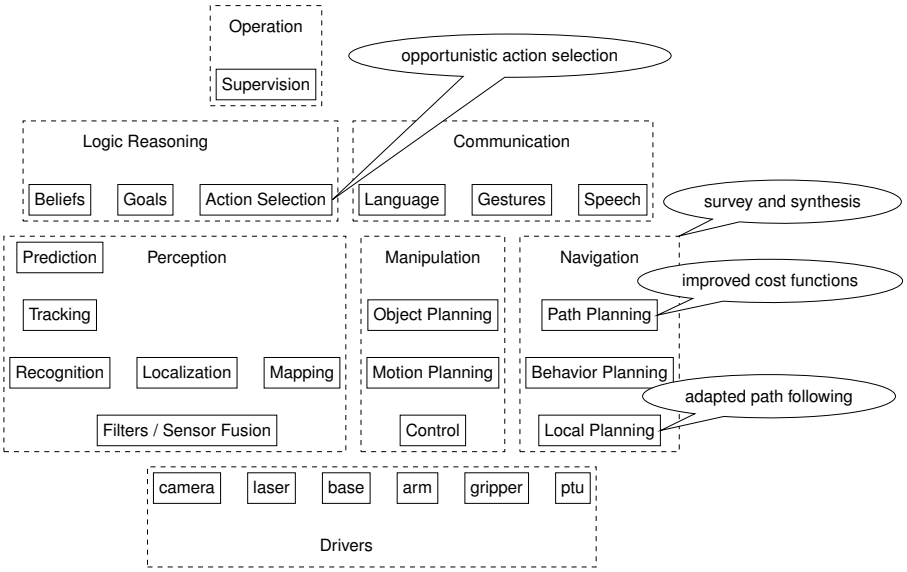


Figure 1.1: This thesis technical contributions as part of an overall robot architecture

applying local planning principles to structured reactive controllers. In navigation, section 2.4 has provided a survey of HRI navigation, which is synthesized into the sketch of a comprehensive framework in section 3.1. Chapter 3 goes into details about the research on improving the cost function in path planning for HRI.

Chapter 2

Concepts

To present the contributions of this thesis, several formal concepts of planning are required. This chapter introduces these concepts.

2.1 Plans

This thesis will mention plans and planning a lot, later using very specific representations and algorithms. In this conceptual section, plans and planning are introduced first from a very abstract point of view, that should be applicable to many other plan representations.

The prior research on planning techniques in general is too vast to present as a whole, the books by Latombe [92] and Lavalle [93] summarize the state of the art.

Planning in simple terms means thinking several steps ahead before acting. So planning is an activity that is based on the notion of actions. To perform actions, at least one acting entity is required, this can be a software process, or a single motor, but in HRI usually humans and robots are considered as atomic acting entities. Teams as collective entities can however also be considered as acting entities. From now on the term "agent" will be used implying a robot or human. Thinking ahead means reasoning about how the environment will change in the future, and how actions can influence this change. This leads to a most generic definition of a plan for an agent:

Definition 2.1. *A plan is a model of future states or events under the influence of agent actions.*

Definition 2.1 is true for many kinds of plans, for the purpose of planning in robotics, more restrictive definitions are often useful. To generate plans for agents at runtime, the model of the future must allow reasoning about it in short time, and it must be creatable based on the knowledge and perception available to a robot. A plan must also be tailored to the abilities of a robot.

In order to create a plan for an agent, the following things need to be considered:

- The environment state, or how to get information about it
- The goal environment state
- The agent abilities
- Quality measures for behavior

Based on the current agent knowledge about the environment, a plan is built by selecting actions that are expected to change the environment until it reaches the desired goal state, if possible. Depending on the domain, a plan might list actions one by one as a list, or be constructed in an algorithmic way. Usually the aim of planning is not just to find any working plan, but a plan that is optimal in some respects, such as the quickest or the most efficient plan.

The quality measures are necessary to favor solutions of higher benefit or reduced negative impact. Those can be implemented as e.g. cost functions or as reward functions. In this thesis only cost functions are being used, to reduce undesirable robot behavior. Cost functions define values for physical or abstract attributes of actions, such that actions and plans can be compared during planning, and cheaper plans be preferred. Risks of failure can be modeled as costs as an example for an abstract attribute. In HRI the expected discomfort to a person is often used as a cost function.

A plan is useful to agents only if the agents can use the plan, resulting in plan-driven agent behavior. And similarly an agent like a robot only produces useful behavior if it follows a plan, such as in Definition 2.2.

Definition 2.2. *A plan-driven agent is an agent that, presented with a plan, reliably produces behavior that satisfies that plan*

Humans can follow a plan this way, though a person may of course chose to act in violation of that plan by mistake or intentionally for other reasons. Humans may act as plan-driven agents if they wish to.

A common problem with plans for the real world is to decide their validity. Commonly in theoretic computer science, plans are validated during construction, such that their validity at creation time is proven. And in otherwise static environments, the validity of a plan remains intact unless unexpected events happen, which is assumed to happen rarely. So in classical planning domains, the validity of a plan does not need to be re-evaluated often in the common flow of events. However in domains with unpredictable dynamics, such as in HRI, plenty of unexpected events happen all the time, many of which have no impact on the validity of a plan, but some which may have an impact. As a particular problem, in HRI plans are often used to produce robot behavior that not just produces a correct outcome, but does so using acceptable manners. The inability to identify that a plan has become invalid causes robots to appear less intelligent. The most simple example is taking away an object a robot is about to grasp. Many prototype robots in research will not adapt to the sudden absence of the object, and continue the grasp motion plan, grasping just thin air. So in general checking the validity of a plan is a difficult chore in robot control that designers will often rather circumvent by making the robot environment as static as possible.

Definition 2.3. *A plan is* valid *for given agents as long as it can be expected that the goal state will be achieved if the agents follow the plan*

Classical planning theory follows practices that grant validity of plans under the assumption that the model used represents reality well enough. In certain controllable domains, this certitude can be achieved by formal proof. However the domain of human-centric environments is difficult to formalize, and in an environment of constant change, it may be required to constantly verify plan validity, which can render the robot behavior too slow for the robot to be a useful actor.

Non-classical approaches may deviate from formal proof of validity, and instead rely on long-time learning and continuous improvement of plans to make

plans that are robust against variations and changes in the environment. The concept of Structured Reactive Controllers [7] is an example for such an approach.

Besides the classical concept of plan validity, in HRI it is necessary to consider plan acceptability. When a robot acts near humans, its actions may be acceptable or not. A plan for HRI can be constructed using cost functions in order to favor more acceptable plans over less acceptable plans. However, such cost functions are subject to unexpected changes as much as plan validity is. This means that after a change in the environment, a plan may still be valid, but not sufficiently acceptable anymore. And in HRI it is not feasible to circumvent this problem by keeping the environment static, because persons near the robot must be free to move freely to feel comfortable.

To make things worse, the acceptability of a plan cannot be expressed as a mere function of its costs, because often a plan with high social costs (regardless of how they are defined) can be acceptable if there is no better alternative, but not acceptable if there is a better alternative. So theoretically the acceptability of a plan is a function of its costs and of the presence of alternative plans with less costs.

The pragmatic solution for these problems is again to modify the robot environment rather than the robot, such as instructing persons near the robot about its planning limitations.

In any case, a plan-driven robot in HRI will often need to adapt to unexpected changes by abandoning the currently selected action sequence in favor of a different action sequence, either by selecting a different plan or by selecting a different set of actions allowed by the current plan.

Which of those strategies can be used depends on the plan representation, which can be rigid or flexible. The next section presents different ways to represent plans.

2.1.1 Plan representation

As said before, plans are models of future states or events. Plans come in various shapes even in daily life, like navigation routes, cooking recipes, off-the-shelf furniture setup instructions, business models, dance lessons, career plans, project plans. Some plans focus on describing actions that have to be performed, others focus on intermediate states before reaching a final state. With respect to individual actions, some plans are totally ordered, some are partially ordered, some plans allow parallel actions, and some plans require synchronized and timed action. Some plans explain every detail while others leave it up to the agent to choose how to perform a step.

The same kind of differences exist for plans in robotics. This section discusses how HRI influences plan representations for robots.

Commonly the first priority of plans is to define world state changes, without necessarily specifying *how* a given agent shall bring about such states. However this has to be refined to agent actions either during planning or during execution when the plan-driven agent attempts to execute the plan. Thus depending on the robot software architecture, plans may specify robot actions or not. In HRI, specifying robot actions in plans plays a more important role than in other robotics domain as the acceptance of robot behavior during interaction depends not just on the result, but also on how it is achieved. At the end of the table setting scenario, the table needs not only to be properly set, but also during the interaction the persons must not have been annoyed or harmed. So in HRI, plans have to not just consider world states to change, but the actions to bring about such changes and their impact on nearby humans.

A distinction that we will need in this paper is between concurrent and non-concurrent plans. In a concurrent plan, the actions represented can happen at the same time if the plan defines this. In non-concurrent plan, only one action can be performed at the same time. As an example, driving home and making a phone call on a mobile phone are two actions that can be performed with an overlap of time, rather than having to finish one before starting the other. The navigation plan itself, such as what turns to make at each intersection on the road map, should be totally ordered.

The importance of concurrent plans in HRI stems from the fact that any desired goal state can be tried to be achieved by a human and a robot together. If the robot tries to achieve a goal using a plan, humans cooperating might be following the same plan, or a different plan. To be useful, a robot need to be able to adapt to this situation, by understanding whether an action by a person nearby contributes to a plan the robot is following itself.

Simple demonstration scenarios may be presented like this: A person asks a robot for help with a task, the robot plans actions for both itself and the persons involved, tells all persons what they should do and when, and this team then acts by the robot leadership. However this does not merely pose ethical problems, but also practical problems like requiring humans to remember and strictly follow a plan. Besides, in a household situation humans may become distracted, or follow some other goals in parallel about which the robot does not know, so it is not convenient to require humans to strictly follow a robot-made plan in their own homes. So in HRI, plans need to allow for concurrent actions, and for some freedom about which agent performs what step in the plan.

With respect to the detail of planning, HRI also has an influence on planning. Classically when faced with a task like setting a table, a planning algorithm decides from the available dishes which dish will go where on the table. When enacting a scenario with a human, this kind of planning would produce an error if the human as an example brings plates to the table that are not those that the robot had chosen. To some degree this problem can be reduced by late commitment, meaning that a plan contains variables for items like plates which are only bound to real world-entities at execution time. However in HRI, this is insufficient, as there is no point in time where a decision can be made definitely, as a human may switch plates at any given time. So HRI also requires planning decisions to remain flexible up to the end of execution.

In sum, the specific challenges of HRI adds to the requirements of plans:

*Req*1 Specify not just world states, but also robot actions, based on social costs

*Req*2 Allow for concurrent enactment by robot and humans without prior task assignments

*Req*3 Allow to flexibly react to decisions by humans contradicting current plan decisions

This thesis focuses on action and navigation planning for robots. The requirements can be applied to both these planning domains.

Research on planning has produced a large number of plan representation formats, suitable for solving specific problems. In abstract action planning (as opposed to e.g. motion planning), a planning language commonly defines both the input to a planning algorithm as well as the output.

A planning language that emerged rather early in planning history is the STRIPS language [33]. In STRIPS planning, a plan is a partially ordered structured of atomic actions, meaning beginning at a start state, an agent may have several next actions to chose from, but for any action, all prerequisite actions must have been executed before starting the action, and typically only one action can be executed at a time. STRIPS plans are created by adding actions to an initially empty plan until the sequence of actions achieves the transition from an initial state to a goal state. A popular alternative planning approach are hierarchical task networks (HTN), such as used by the SHOP2 planner [109], in which constraints between tasks can be specified to guide the planning process. Some work exists on adapting such planning algorithms to produce acceptable action plans in the context of HRI, such as HATP [1].

In this thesis, the plan representation used for action plans (not navigation plans) is the Reactive Plan Language (RPL) [7]. In RPL, a plan is a algorithm that uses actions in the same way as it invokes functions. It will be introduced in greater detail in Section 2.6. RPL extends the Common LISP [136] programming language, thereby offering algorithmic freedom to integrate the additional requirements of HRI listed above. So *Req*1 can be resolved easily by adding robot actions to plans. *Req*2 is also possible with RPL, as RPL provides strong support for concurrent programs. Finally *Req*3 can also be implemented with RPL by making decisions revertible, and later sections will show how.

With respect to the specific plan requirements in HRI, navigation planning is also affected. Addition of robot actions based on social costs as required by *Req*1 can relate to prompting gestures, as well as behaviors such as moving on the

right side of a corridor. Concurrent enactment as in $Req2$ is often not considered in navigation planning, as planning for the robot alone allows for a lot of support scenarios.

Advanced topics like guiding, following or moving in formation with humans however may require some planning for concurrent enactment. Reverting decisions quickly in navigation plans if humans move unpredictably as in $Req3$ in navigation planning may be less problematic, given that the navigation planning allows fast replanning, but it will be shown later see that replanning alone does not produce human-friendly behavior in all cases.

In navigation planning, a global plan is usually a sequence of waypoints to follow in cartesian space. While in industrial motion planning such a sequence usually must be followed with high precision, in HRI the waypoints are often just a coarse guideline to allow the robot finding a path without running into dead ends and by taking an optimal route, with the exact states between any two consecutive waypoints being free for a local planner to chose.

2.1.2 Planning Layers for robots

Robots are designed to be able to execute given plans. In many domains it is also necessary or useful for a robot to itself create new plans to execute, based on current knowledge of the environment. The formulation "the robot itself plans" is an anthropomorphism, treating the robot as a living creature. In reality the planning process we talk about may well happen in a different location, on a computer physically distant from the robot body. The point here is rather that planning may happen during runtime, while the robot is in the middle of a situation with persons acting in the same environment. Planning during such situations requires the planning processes to be short, and to easily adapt to unexpected changes. Using layers of abstraction in planning helps to achieve responsive planning behavior.

When a robot plans its own actions, there is one or more processing structures responsible for creating plans. The planning problems to solve are both high-level decision such as which item to grasp, and low-level decisions such as the

speed of a motor in an elbow joint. Designing the software for these separate problems usually requires separate subject-matter experts. Additionally deciding whether to change a plan (e.g. grasp a different item, or move the elbow joint at a different velocity) must both be done at all times, so it is practical to organize the software with independent processes. It is also common that the same solver for abstract problems is deployed on different types of robot requiring each specific planners for low-level planning problems. So in order for planning processes to be reusable it helps separating them into robot specific and robot-agnostic modules. All these circumstances lead to a common split of robot planning structures into separate modules for separate layers of data abstraction. The output of one planning layer becomes the input of other planners.

This works under the assumption that for every plan that one planner proved to be correct, the following planners can always also find a valid plan. This assumption is often valid, but can break. Humans know this problem when driving a car: A car parking space may seem large enough for the car, so a plan may be formed early to move the car into the space, get out, and do some shopping. Having accepted this plan and starting to execute it, another more detailed plan is required to get the car into the parking space without collisions. This means planning and performing several steering motions to move the car into the vacant space, and in this process a human may find out that due to the constraints of the car steering, the car can actually not be moved into this vacant space. The observable behavior of drivers attempting to park and then giving up shows the separation of planning into several independent planning processes, and the problem that can arise if secondary planners cannot find solutions to satisfy earlier plans of higher-level of abstraction.

Still the separation of planning into separate planning processes is valid as long as the assumption holds that later planners will find solutions. For this thesis, we will look at two separate levels of planning: Action selection and navigation. Action selection considers actions like “Go to position X” or “pick up item Y”, but not finer-grained decisions like how fast to move, which hand to use for picking up, and so on. Navigation is a lower level of abstraction that is traditionally concerned with finding a path on a map and controlling the base motion of the robot.

There are other planning processes typically running on robots that are outside the scope of this thesis. As examples, there is grasp planning, multi-robot coordination and planning for legged walking. Those topics may each also be analyzed with respect to acting in presence of humans, such as grasping an object in a way that makes it easy for a later handover step. As an example a knife can be handed over grip fist or tip first, with very different effects on human comfort. But this thesis is limited to navigation and high-level action planning.

2.2 Planning in worlds with unpredictable dynamics

So how do planners create plans? All planning involves representing the evolution of world states given actions. The world changes due to the actions an actor performs. However the world also changes due to actions other agents perform, and the world even changes due to natural events (gravity pulling objects, water streaming out of a tap, or even weather conditions, as examples). Some actions may also have effects that are difficult to predict exactly.

Early planning frameworks in Artificial Intelligence created plans for practically static worlds, and actions with predictable effects. "Practically static" means that while time still passes and earth rotates, in the space considered for planning, the solid objects that can be observed do not change position or state noticeably, unless the robot performs an action on them.

Such simple domains offer advantages for research in planning. One advantage is that during the planning process, the world does not change in a relevant way, so once the planning process has finished, the world state is still the same as before planning, and the plan is still valid. Another advantage is that the future world change after each action could be predicted with high confidence. To some degree this still holds for worlds with predictable dynamics, where there are independently dynamic objects, but those objects change state in a predictable way.

The theoretical approach to planning is to build a model representing the future as a function of actions. The planning process starts with a model where the agent remains passive, not actively influencing the future of the world. Then

the planning process adds actions to this model with the agent influencing the future. Each time an action is added, this results in a new model of the future. Since alternative actions could be added, a planner considers a tree of models with the initial model at the root, and branches being modifications to existing models. Once a suitable model is found that leads to a desirable future, the planning process can stop, and the model be used as a plan for the robot.

A key challenge is to avoid having to try out all possible combinations of actions, because this set grows exponentially with the abilities of an agent, the items in the world and their states. So most of the research in planning goes into structuring the planning problems and processes such that only very few combinations have to be tried out to find a solution. The techniques for achieving this vary a lot between the planning problems to solve, leading to specific planning strategies for specific problem domains.

The HRI domain is very different from static domains, in that it constantly changes, and it changes in unpredictable ways very often. If a planner cannot reliably predict the world state five seconds into the future, it can also not reliably check whether the consequences of an action that takes five seconds will satisfy constraints.

So the general planning strategy described above needs to be extended in every domain of unpredictable dynamics.

The following strategies exist to plan in unpredictably changing worlds:

- Replan
- Predictions and temporal planning
- Local planning
- Generic plans
- Stochastic planning

The following subsections explain each of those strategies in more detail.

2.2.1 Predictions and temporal planning

Predictions and temporal planning are suitable techniques when the domain changes a lot and gradually, but the changes are mostly predictable by nature. For changes to be predictable, they need to be non-chaotic, and well-observable. Non-chaotic means that for small variations of the original state or actions, the result also varies only a little, and not drastically.

Temporal planning means that plan actions and other events are modeled with durations, and the plan describes not just what actions can be done according to the plan, but *when* they can be done. This allows for many more plan variants where actions are scheduled at different times. The increased search space causes planning time to grow quickly, making a robot less reactive to events. For interaction, such delays can be unacceptable for HRI. Also such delays make the resulting plan less robust. The more robust a plan is, the more variation in real world events it can support without failure. The planning time itself can cause the resulting plan to be outdated when finished, and invalid.

A strict temporal plan becomes invalid if there are any unexpected delays. Planning with durations and times is also often called "scheduling", and a common approach to plan more flexibly with durations are variations of the so called "Critical Path Planning" method [70], used predominantly in operations research and supply-chain management.

The other aspect of temporal planning is predicting how long events will last that are outside the control of the robot. For people, attempts at prediction can be made regarding what a person will do, how it will do it and when and for how long a person will be doing something. This can be done regarding human manipulation actions, such as what item a human will pick up or use, regarding communication, or regarding navigation, such as what path a person will take. As for most prediction processes, short term predictions are easier to make than long term predictions, and at higher levels of abstraction predictions are more robust than at lower levels. As an example, when an empty-handed person approaches a fridge, the abstract prediction that the person will take some items is more likely to occur than a more specific prediction that the person will pick up a bottle of milk. In navigation, an abstract prediction that a walking person

is going to his desk is more likely to occur than a prediction of what path the person will be taken and what velocity and acceleration the person will apply. For temporal planning, predictions need to be reliable with respect to the space and time of actions, and

Due to these vulnerabilities, the strategy of predictions and temporal planning alone is insufficient to deal with unpredictable worlds in general.

2.2.2 Replanning

Replanning is a common technique to deal with dynamic worlds. Of all strategies it is easiest to implement. Replanning means that while a robot executes a plan, a different plan is generated. Then the current plan is interrupted and the new plan is executed. While this is always necessary when the current plan has failed, replanning can also be used when the current plan is still possible, but a better plan has become available. In general it is difficult to know whether unexpected changed circumstances allow for a better plan to exist or not. If a robot moves through an office building to a specific room, and a door that was assumed to be closed is unexpectedly open, it is a challenge to estimate without replanning whether this change allows for a better route to the goal or not. This is a variation of the so called "frame problem" [101], a philosophical dilemma of all classical artificial intelligence approaches. To keep things simple, replanning can be applied on every unexpected change, or even simpler in regular intervals, saving the effort of designing a correct and robust detection and classification of unexpected events. This works when the replanning effort is sufficiently low.

Replanning is a suitable technique when the duration of the planning effort is sufficiently short, and the unexpected world changes are sufficiently seldom and of short durations.

Replanning can be combined with the so called "Anytime Planning" strategy. In anytime planning, the idea is to have a valid but possibly suboptimal plan early, and then searching for better plans only as much as time allows it. This prevents the robot from being stuck without a plan when a plan is required, but as a consequence the plan the robot follows may be of bad quality. So anytime

planning is a compromise between optimal behavior and reactivity, and it can be applied to most planning techniques and domains, such as action and navigation planning [125, 108].

2.2.3 Local planning

A common strategy for agents in dynamic environments is to decide on actions in a local context rather than the global context. That way a global goal state cannot reached by following a single plan from start to goal, but by iteratively planning a small time into the future to approach the goal. If the iterations overall approach the goal, the goal will likely be reached eventually, though it is then unknown how many steps will be required until the goal is reached. A common local planning strategy is to make an agent strictly converge towards a goal, meaning every action reduces the distance to the goal. The word "distance" here can be applied to any measure of difference between the current state and the desired state of the world. When clearing a table of all items, a local planning strategy is to remove the items one by one, thus the distance to the goal (empty table) can be measured as the number of items left on the table.

Local planning can be combined with abstract global planning. This gives rise to the notion of global and local planners, as action selection algorithms.

Definition 2.4. *A global planner produces a plan as a complete solution transforming the current state of the world into a goal state, satisfying global properties.*

Global properties could be e.g. correctness or optimality. In most domains where global planners are being used, the size of the shortest valid plan and the time required to produce it are not bounded, they scale with the size and complexity of the problem. This makes replanning impractical in domains, where a timely response is required, such as for navigation.

This can also quickly become an issue in HRI, as a person cooperating with the robot has to wait for a planned action, becoming impatient and inconvenienced.

If the global plan however has degrees of freedom, then it may not become invalid by unpredicted changes. A local planner then has to change lower-level decisions even when the environment slightly changes without the need to globally replan.

Definition 2.5. *A local planner produces a plan that transforms the current state of the world into a target state which is closer to the goal state, and satisfies local properties.*

Local properties could be e.g. collision-safety or bounded jerk. Local planning means that the planning process does not necessarily find a complete plan towards a goal, but only a plan towards a subgoal. This allows for the planning horizon to be small, and thus the unexpected changes in the world to have only little impact on robot behavior. While in navigation outside HRI the subgoals are commonly selected to strictly converge towards the global goal, it is conceivable in HRI to temporarily allow for subgoals that do not converge towards the goal, but provide polite behavior. As an example a robot might stop and step back to let somebody else pass through, even if this local plan temporarily moves the robot further away from its goal.

In navigation such a subgoal is usually a location around the robot that is closer to the goal than the robot. In action planning, when the goal is to clear all items from a table, a subgoal can be to remove one item from the table.

While in HRI strict convergence towards a global goal may not provide the best robot behavior at all times, eventual convergence is still required for the robot to reach the global goal by iterative local planning. So the local planning strategy requires a distance measure by which a suitable subgoal and local plan can be selected that converges towards the global goal. The computation of that distance measure has to be significantly cheaper than creating a global plan, else the strategy offers no advantage over global planning. One way to achieve this is by having a global planner create a coarse plan in such a way that the local planner can use states in the global plan as subgoals with a simple distance measure. So this strategy requires a contract between a local and a global planner that the global plan always defines subgoals that can be approached using the distance measure of the local planner.

It is noteworthy that the output of a local planner is often not called a "plan", as it often produces just a single action to execute. Local planners are also often called "reactive planners". Also, the word "planner" alone is in literature often synonymous with what we call "global planner" above, meaning that in literature, when talking about a "planner", an algorithm satisfying the definition of "global planner" above is meant. This is due to historic reasons and in particular the separation of research on symbolic planning from the research activities in robotics.

Finding suitable subgoals without a global plan defining them may be impossible due to local minima and inevitable failure states. A local minimum is a subgoal from which no other subgoal can be selected within planning range which approaches the final goal. In navigation that is typically a cul-de-sac where the robot would have to backtrack away from the goal for some time before choosing a different path at an intersection. In action planning, a constructed example is if the robot has the goals of putting an item into the fridge, keeping the fridge door shut when not interacting with the fridge. An unsuitable local minimum can occur if the robot opens the door in preparation of putting the item in, then closes the door to conserve energy, and does this in an infinite loop.

An inevitable failure state is a state where the final goal can not be achieved anymore, even if there are still valid actions that converge towards the goal state. In navigation, this is usually a temporal goal, such at arriving at the train station before 9am. If one attempts to approach the train station in iterative steps (following sign posts) and still needs 10 minutes to reach it at 08:55am, then no action can achieve the goal. Another class of inevitable failure sates in navigation are inevitable collision states such as presented by Fraichard et al. [38], where a future collision cannot be avoided anymore.

In action planning, an example for an inevitable failure state is an irreversible action, such as cracking an egg. If the final goal requires an uncracked egg, then cracking the egg makes achieving the final goal impossible.

In summary, global planners are typically characterized by:

- Planning duration not restricted
- Produces start-to-end sequence of states from start state to goal state

- Can generally grant achievement of predicted global behavior properties

Local planners on the other hand typically have the following traits

- Satisfies planning duration constraints
- Produces only next *n* commands, not a complete model of future states
- Cannot generally grant achievement of global behavior properties

It is noteworthy that while global planners can grant achievement of global properties of behavior under the assumption of a predicted situation, this grant does not extend to unpredicted changes in the environment. In other words, the more dynamic a situation is, the less reliable global plans become in predicting global properties of the behavior.

In some specific domains it may be possible and useful to unify global and local planner, however in the general case this is not the case. In such specific domain the maximal global distance the robot needs to travel might be bounded, and the required reaction times to expectable unpredicted events might be large enough.

Even in such domains where global and local planner could be implemented as one, it might be a good architectural decision to split them into separate modules following the principle of separation of concerns. A system might also have more than one global and one local planner.

2.2.4 Generic plans

A generic plan is a plan that is known to be valid for numerous unexpected world state variations. This would allow a robot to still use the same plan, even after the world has changed in unexpected ways. This can be achieved in several ways.

- prepared event responses
- degrees of freedom

A plan having prepared event responses defines valid actions for the robot to take after certain events. The Structured Reactive Controller Framework [7]

gives examples for this. Such events need not happen very likely, but a plan may created to provide for the possibility of such events. As an example, a plan may require a robot to grasp an object, lift it, then move it to a new place. If the object slips during the lifting, the plan would have failed, and a new plan would be required. This unless the plan already had a prepared response for the slipping event, such as re-trying the grasp and lift.

Plans may be or may not define degrees of freedom. A degree of freedom means that a robot may select more than one action according to the plan at a time during the execution of the plan. Such degrees of freedom can allow selection techniques that provide suitable behavior of the robot under a broad range of unexpected future developments.

However this requires the control algorithm that executes the plan to make decisions that the planning left open, and of which the planning process has no control. Conceptually this causes problem if the planning construction process aims to improve quality of behavior according to a quality measure. If some decisions are only taken during execution time, the planner might not be able to well-predict the quality of a plan, given that different decisions at execution time can yield different behavior quality.

2.2.5 Stochastic planning

The goal of stochastic planning is to find a global plan that has the best estimation of success given a known likelihood of certain future events. While this reduces the likelihood of unexpected events, it does not produce a strategy for handling unexpected events happening despite of low probability.

A common stochastic planning approach are variants of Markov Decision Processes (MDP). This thesis will not go into more detail about this planning approach. The overview by Boutilier et al. [12] explains the concepts and highlights the challenges of using MDPs for planning. A main challenge is to define a suitable space of world states to represent the environment and possible actions. With an arbitrary number of humans being able to perform arbitrary actions in the same space of the robot, the number of possible world states becomes too

large to handle. So in HRI, MDPs are suitable only where most variations can be abstracted. MDPs then allow creating so-called optimal policies, defining for each state the action that most likely leads to a state with the greatest reward. However, another challenge of MDPs is that this computed optimal policy is a black box to researchers, meaning it cannot easily be analyzed for flaws that are due to the construction of the state space. So for the scenarios and planning challenges of this thesis, MDPs were not considered, while they certainly can be adequate solutions in other HRI scenarios.

2.2.6 Summary on unpredictable domain planning

As shown in the previous subsections, several strategies exist to use a planning approach in domains with unpredictable dynamics. In the context of HRI, each strategy may have its place in specific robot applications. This thesis takes a closer look at replanning and local planning for navigation, and combines local planning and generic plans for task planning.

Based on those strategies, a robot becomes able to handle certain unpredictable dynamics that have to be expected when acting around humans. As a last aspect of planning for HRI, the next section looks at how planning can be directed to produce more acceptable robot behavior.

2.3 Human-centered Behavior Quality

In HRI, robot behavior needs to satisfy the demands of human stakeholders involved in the robot activity, such that persons may accept a robot in their living environment.

Where the robot behavior is controlled by a plan, this requires the plan-driven behavior to satisfy human-centered quality criteria. During plan generation, a quality measure is required that distinguishes between better and worse plans. So when a task like setting a table can be correctly be solved by different plans, such a quality measure defines an ordering on these plans. In general robotics, the most commonly used quality measures are time or energy efficiency, meaning

the robot should use as little time or energy to solve a task. In HRI, the common assumption is that efficiency of behavior of the robot is not the dominant factor in how humans rate the quality of the robot. As an example, instead of planning to minimize its own effort, a robot may plan altruistically such as to minimize the effort of nearby humans, not itself. Other factors are to act avoiding risks and acting respectfully.

So for planning in HRI, different quality measures are required to allow creation of plans that generate acceptable robot behavior. A lot of effort in HRI research goes into finding suitable quality measures. A common way to define such quality measures are cost models that define social costs of robot actions. Social costs consider a virtual effort assigned to some action, such that algorithms that already optimize plans based on costs models can be applied. A social cost model thus assigns higher costs for one action than another if the first action is assumed to be less desirable than the latter.

Compared to the challenge of finding suitable quality measures and proving them to be indeed suitable, it seems trivial to apply such cost functions to any existing planning framework. However, as we have seen in section 2.2, robots acting around humans often have to replan during the execution of a plan. This has a quality impact as well. If a robot in his behavior switches between different plans multiple times, then even if the quality of each of the individual plans had been optimized, the resulting behavior of the robot may become bad. This is not well researched, and a key conceptual contribution of this thesis, as shown for the navigation planning case in section 3.3.

2.3.1 Social cost models

Cost and utility models are ubiquitous in computer science as a method to create appropriate behavior of systems. For human-centered behavior, it is possible to attach higher costs to worse behavior to allow planning to increase behavior quality by reducing costs.

Among the main issues in HRI around cost functions are these:

- Measuring human preferences over robot behaviors
- Finding suitable cost models for individual aspects
- Combining several cost models

Cost functions are trying to model the way a person will judge events as satisfying or dissatisfying. This is of course personal and context-dependent, but preferences may still be measured by exposing humans to different robot behaviors in experiments, and retrieving a measure of satisfaction by observing those persons or asking them to rate the experience. This is common practice as for other research than HRI. Yet measuring preferences via experiments in HRI faces several additional challenges:

- The situation of interaction with a robot is unusual
- The experimental situation is artificial
- Interactive situations play out differently every time
- A robot functioning robustly is required
- Robot research prototypes have safety risks
- Interaction situations are complex, minor variations can cause large differences
- Robot behavior is driven by a complex set of parameters, minor variations can cause large difference
- Most acceptable behavior in one context is not necessarily most acceptable in another context

All these challenges increase the effort of finding good cost models empirically and reduce the applicability of experimental results to a more general context.

The other main issue is that having several cost functions, there is so far no justified way of comparing them quantitatively. As an example: Assume that a robot moving close to a human starts producing negative reactions when approaching closer than a certain distance at a certain speed. We can empirically test how close a specific robot can move to humans in specific contexts, and derive a cost function from that that will make algorithms prefer path outside that distance. As another example, we can assume that the human satisfaction given a fetch task (e.g. the robot bringing coffee to the human) decreases with the time the robot spends on traveling (which may also cause the coffee to get colder). We

Designation	Specification	Reserved for ...
Intimate distance	0 - 45cm	Embracing, touching, whispering
Personal distance	45 - 120cm	Friends
Social distance	1.2 - 3.6m	Acquaintances and strangers
Public distance	> 3.6m	Public speaking

Table 2.1: Proxemic interpersonal distances as found by E. T. Hall [52]. How those categories and the values are to be adopted for the human-robot context is an open research question

can model a second cost function from that. But if the task of the robot is to fetch a coffee going past another human, then how are we to compare these two cost functions, of task duration and distance to a human? How much cost in the distance cost function should the robot tolerate to fetch the coffee in a shorter time? Generally humans can face the same challenge when trying to behave politely, it is sometimes difficult to judge beforehand which behavior will annoy other humans or not. A social solution to such problems is to ask forgiveness when in doubt, and such a solution may similarly make a robot more acceptable.

The following subsections explore the different kinds of cost functions that are common in HRI to produce more acceptable robot behavior.

2.3.1.1 Social spaces and Proxemics

There is a large amount of publications on the topic of robot navigation in human environments that focuses on avoiding undesirable locations. While there is more to navigation than just locations (such as direction, velocity and trajectory shape), the properties of locations is comparatively easy to investigate. A lot of this research is based on the concept of "Proxemics", which was invented by E. T. Hall [52]. He found different social distances humans choose for interaction depending on the relationship and intention as shown in Table 2.1.

Applying proxemics in HRI can still be challenging, as the distances given by Hall are only validated for a special context, two humans interacting with each other while standing. For cases of more humans, or humans moving in the same space without interacting, or interactions under special circumstances like constrained space, the proxemics model does not describe adaptation. Similarly, the model does not describe how two individuals settle for a distance. Nevertheless, work on HRI can still use the concept in HRI in building cost functions that at least make robots choose certain distances from humans.

A larger overview of cost functions for spaces has already been given by Jorge Martinez [120]. So here is only a shorter overview on the subject. Basically humans react to other agents taking up certain space in their vicinity, for various reasons. There are at several categories of spaces that humans tend to avoid:

- Personal space
- Interaction space
- Affordance space
- Reserved space

Personal spaces relate to locations around the body of a person, that people do not like being taken by other agents, for various reasons. The proxemics spaces are an example of this. When walking, people prefer the space in front of them to be free. Similarly, when walking people often prefer not to be closely tailed by other agents, as this is a threatening situation.

Interaction spaces exist where interaction happens, between two people, or a person and an object. With respect to interaction between people, the space taken by the interaction is labeled O-space in literature(because it is represented with an elliptic shape usually). Similarly there is a definition of a P-Space, which is an area around the O-Space where entering it will often be interpreted as a cue that the person entering it wishes to participate in the interaction. Where interaction requires a clear visual field, silence, or calmness, and robot intruding O-Space or P-Space may cause adverse emotional reactions.

Affordance spaces are spaces where interaction may happen in the future, so these are potential interaction spaces. Examples are space in front of information panels, vending machines and emergency exits. The difference to interaction

spaces is that in the absence of people, a robot using such space does may not be behaving against acceptance.

Reserved spaces are spaces that humans do not want to be taken for other reasons. As an example, people often feel reservation about strangers (or robots) entering their homes, regardless of whether they are present to witness this or not. Similarly humans are supposed to stay off the lawn in some places, and not stay on the road at a pedestrian crossing when the pedestrian lights turn red. So for various legal, security or ceremonial reasons, spaces may also be undesirable.

Several examples for the consideration of spaces in navigation is given in subsection 2.4.2. This thesis does not contribute further to the analysis of the specific spatial categories that exist.

However it is important to know that knowing about the potential places that a robot should rather avoid does not immediately help to design acceptable robot behavior. The main challenges are:

- space properties are context dependent
- space properties can quickly change over time
- space properties are personal and subjective
- no natural priority exists for multiple overlapping spaces

To give some examples: In an almost empty train, when there is just one seat taken by a person, choosing the seat next to the person is intuitively a gesture of familiarity, that can be met by different kind of emotional reactions by the person there. If the train car were full except for this place, taking the same place will not result in the same reaction.

A similar effect can be observed for shunning. If the places around one specific person are avoided far more than the places for other persons, this can be interpreted as shunning the respective person.

As another example, while people do usually not like strangers in their homes, they will usually accept firemen entering their homes in case of a fire. So all

spaces have mitigating circumstances where entering the spaces is acceptable or even desired.

Several spaces with social importance also change position and shape over time, making planning difficult. An affordance space immediately changes properties when a person wants to use it for an activity. And when humans walk, the position of personal spaces change obviously with the position of the human, the shape of the personal space may also easily change with more free space desired in front of a person as the person moves at a faster pace. When a person approaches the robot, without intent to interact with the robot, should the robot move out of the personal zone to behave acceptably, or is this a case with different rules?

To make matters worse multiple overlaying spaces are difficult to compare. Imagine a robot that has to cross a room with humans and objects, and there are multiple distinct paths which the robot can choose, yet each path passes through a space that has social properties. As an example a path p_1 may pass very close to a human, path p_2 may pass between two interacting people, path p_3 passes over freshly cleaned floor. Each path has properties that make it somewhat undesirable to have the robot follow through it, but how can these properties compared with each other to decide which path to take?

Similarly it is difficult to quantify how undesirable a space is, compared to benefits of actions. Imagine a robot has a choice of passing a person with just a 10 cm gap between robot and human, or choosing a detour that adds n seconds to the time of arrival at the robot's goal position. So for what values of n should the robot select the path close to the human, and for which values the detour? Can there be an absolute value, or does this depend on the total expected time for the robot to reach its goal? Does it also depend on the urgency of the robot task?

One reason that these problems exist is that the proxemics model describes only symptoms of human behavior, but not underlying causes. It describes how human will choose or prefer social distances, but not what biological or psychological processes produce this behavior, and how those processes can be described in terms of inputs and context. Thus a spatial cost model to direct robot

behavior does not directly reason about the effects of candidate actions on the human mood, fear, distraction, fatigue or stress.

The questions and concerns raised above show that categories of spaces are insufficient as specifications for robot behavior, and additional research work is required to produce a specification using such space semantics.

While a large amount of publications deals with social cost models for spaces the robot should prefer to avoid, social cost models related to other aspect of robot behavior than spatial presence are being researched. There is research on optimizing robot velocity and acceleration, gaze direction [17, 139], distance-taking when following or guiding a person [112], positioning for interaction like handover [97], and social preference order of actions [1].

Another group of social models are concerned with social protocols of human civilization. As an example, walking on the right side of a corridor, knocking at a door before opening, greeting each known person once a day, holding a door open after passing to somebody else who wants to pass the door, and many more examples. Social protocols are also culture-dependent.

Any social cost function may be context dependent, however costs may still be classified as to whether they model historic context or not. Without historic context, the current belief about the world serves as context, and two situations which are similar will have the same costs by the model.

With historic context however, a situation two situations which are exactly the same can have very different costs, if they have a different history. As an example, a robot might inform a person close by of its current intention or action verbally. The first time this happens, the cost of this action might be very low, the person might find that very useful. However, if the robot has done so a dozen times in the last minute, the person may become very annoyed. We can therefore expect that several social cost functions require a historic context factor.

This concludes the overview of approaches to human-centered behavior quality. Core to HRI research to make plan-driven robots more socially acceptable is to apply quality measures to actions during planning, and to use social cost models as quality measures. This causes plan-driven robots to follow plans with

reduced social costs, which has been shown to improve robot behavior in several experiments. There already exists a large set of individual social cost models solving specific situations, but a unifying theory of social costs and robot behavior is yet to be defined.

The last sections also shows that whatever cost model are applied during planning, they are prone to rapid changes during execution time of a robot, so to maintain social acceptability, replanning is required under changed social costs. And at the same time, repeatedly switching plans may also adversely affect robot behavior, so a cost model that has been show to be valid in static situations cannot generally be applied to dynamic situations by mere replanning.

2.4 Related work in Human-aware Navigation Planning

The work in this section has been partly published in Robotics and Autonomous Systems Journal [86].

This section gives an overview of previous work on human-aware navigation with a focus on architecture and software modules.

The software required to move robots among humans is typically divided into separate software modules executing separate tasks. A framework then is the composition of such modules. To classify related work, an overview of navigation frameworks used for human-aware navigation is given.

2.4.1 General navigation software modules

To grasp the diversity and common practice in navigation frameworks for autonomous robots, here are some overview figures taken from robotics publications of other researchers:

- Figure 2.1 an architecture deployed on the ROBOX robot
- Figure 2.2 the DIARC architecture deployed on several platforms
- Figure 2.3 an architecture deployed on the RACKHAM robot
- Figure 2.4 an architecture deployed on the JIDO robot

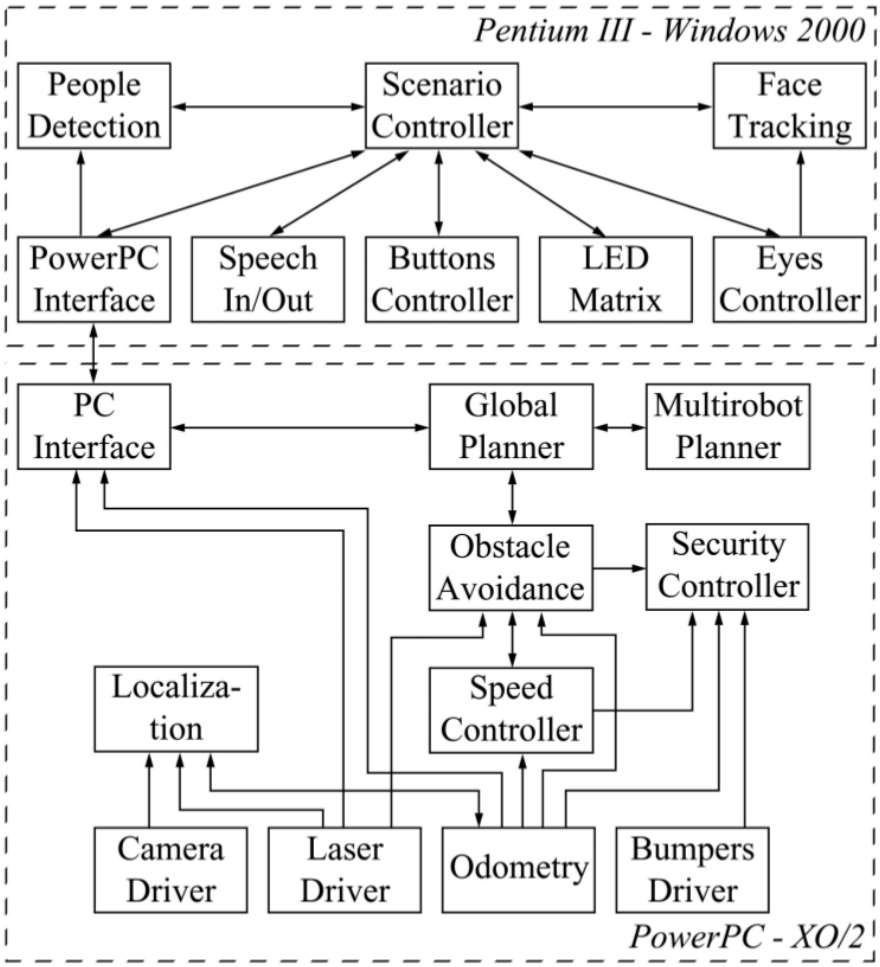


Figure 2.1: ROBOX Overview, Figure taken from [115]

- Figure 2.5 an architecture deployed on the TOOMAS robot
- Figure 2.6 an architecture deployed several robots (TUM-ROSIE, PR2)

All frameworks shown here were deployed as research platforms, not commercial platforms, which means the design followed the specific research focus and constraints of the respective research labs. The figures do not strictly follow any modeling language, so they can only be compared superficially.

The ROBOX architecture is presented in a split of two module sets running on different Computers for load balancing. ROBOX in the state presented in [115] is a mobile platform without arms, so no manipulation modules are part of the figure. A recognizable pattern to notice is the use of a top-level central action se-

lection unit called "Scenario Controller", and the split between "Global Planner" and "Obstacle Avoidance" (Local Planner). A people detection and a face tracking module are present for HRI. Actuators for communication are also present (voice out and eyes).

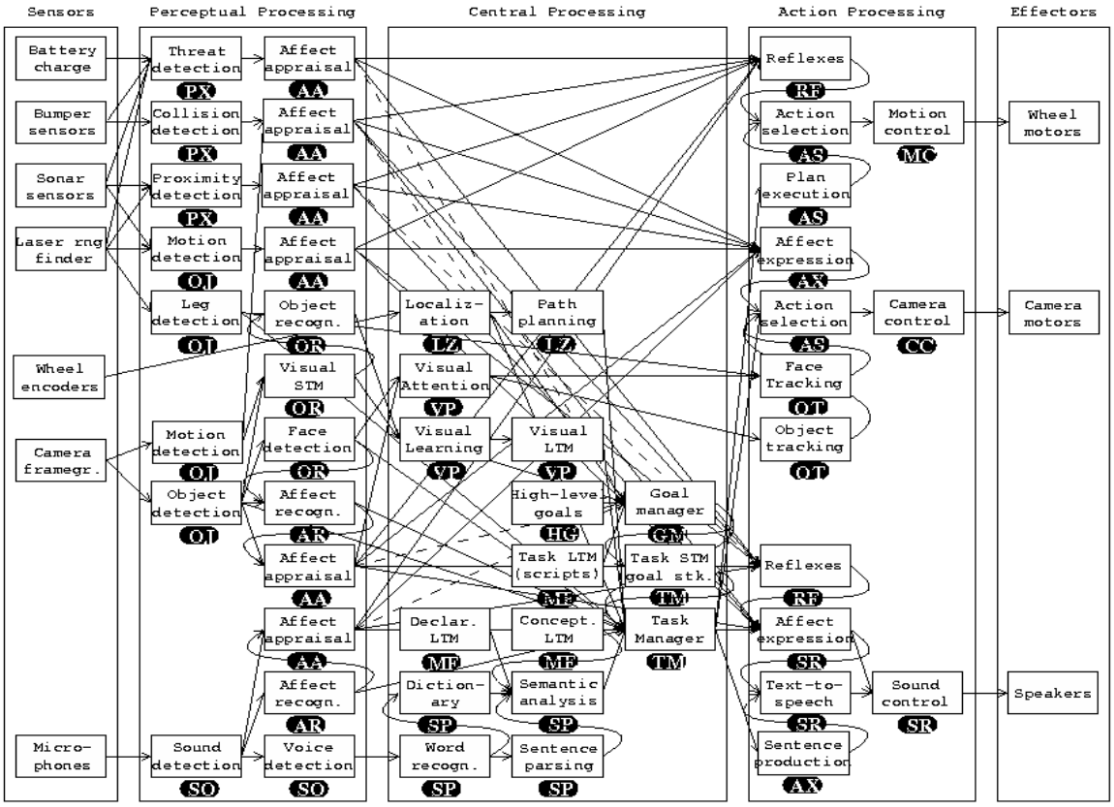


Figure 2.2: High level view of DIARC architecture, taken from [130]

The DIARC architecture [130] shown in figure 2.2 has a focus on cognition and communication. There are several components dealing with goals and tasks, and the figure reveals a split between path planning and "reflexes" for navigation.

Like ROBOX, RACKHAM [25] is a mobile platform without arms. Again a top-level central action selection module is present "OpenPRS Supervision", and a split between global and local planning ("VSTP" and "NDD" in the Figure). A face detection module and an on-screen face module serve for communication.

The JIDO robot [134] reuses several modules from RACKHAM, but additionally has modules for manipulation, as JIDO has a robotic arm. In addition to RACKHAM, it also uses hand tracking to detect gestures.

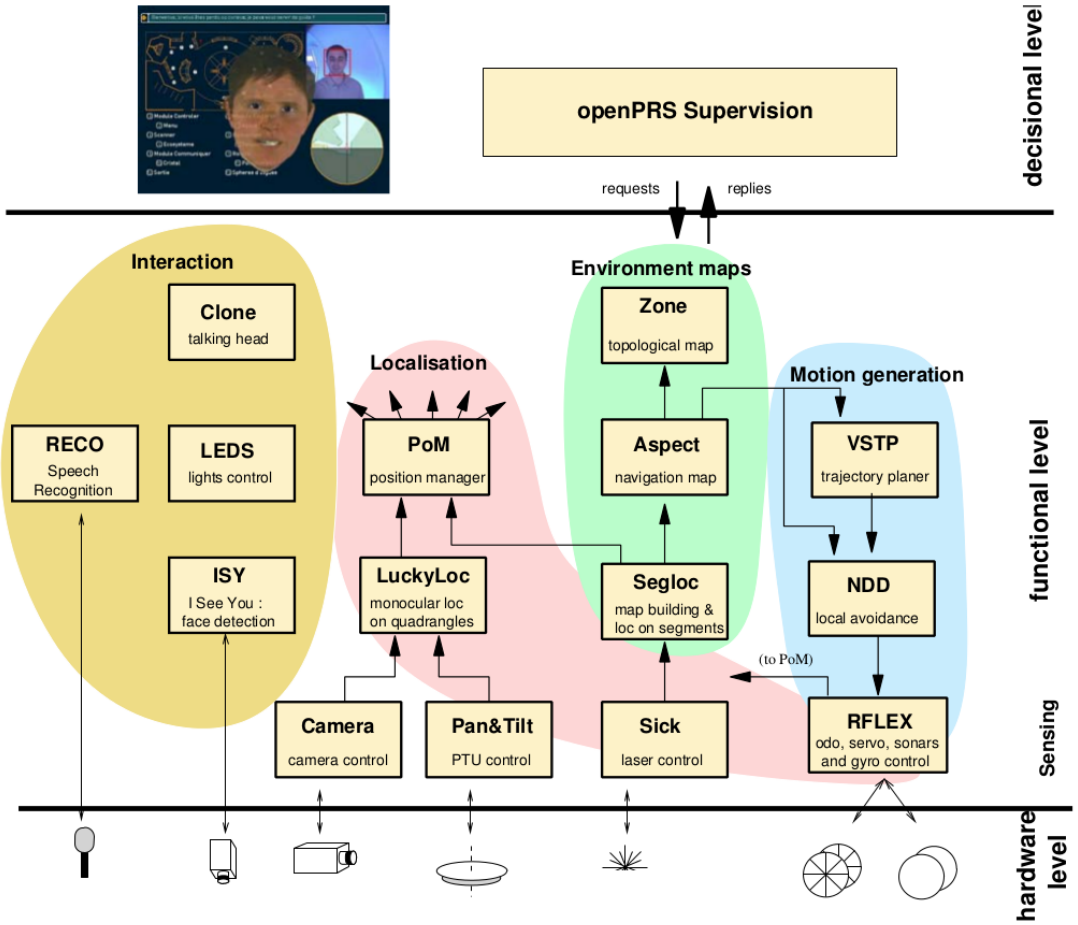


Figure 2.3: RACKHAM Overview, Figure taken from [25]

The TOOMAS robot [51] is a guide robot without arms. It uses a state machine for centralized action selection, and a blackboard architecture separating skill and application layer.

Figure 2.6 shows the Cognitive Robot Architecture Architecture developed by Prof. Beetz et al. and deployed on a PR2 Robot "James" [145] and the custom made ROSIE robot. The focus on CRAM is on high-level knowledge processing, so Figure 2.6 does not show lower levels in detail. However, it serves here to convey that the centralized action selection process on a robot can vary greatly between frameworks from rather self-contained modules as state-machines on TOOMAS to distributed knowledge processing and reasoning modules in CRAM.

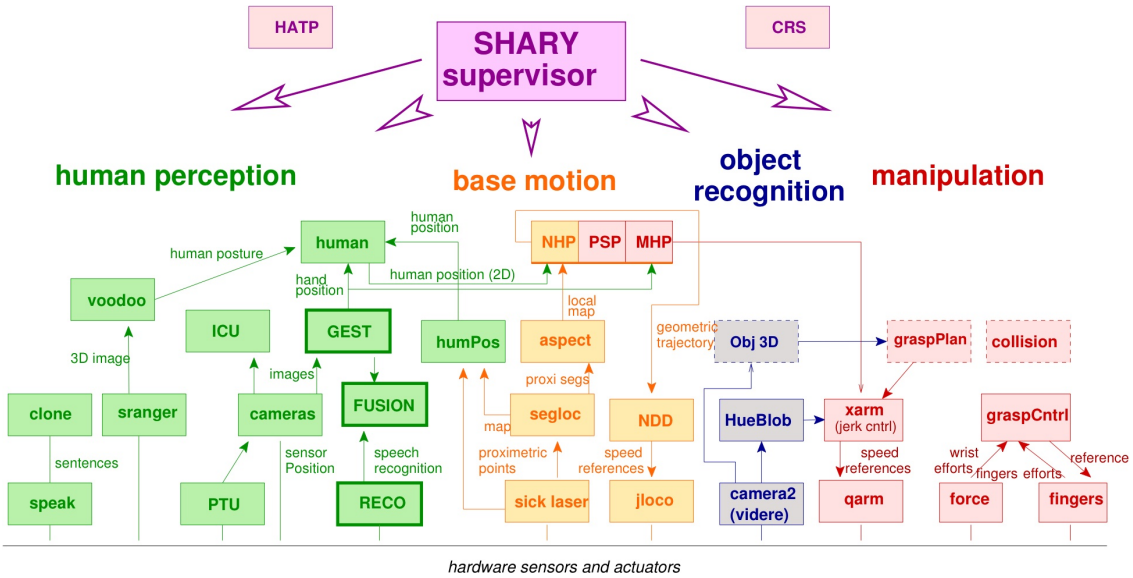


Figure 2.4: Jido Robot Software Architecture, Copyright ©2008 LAAS/CNRS

The figures show that there are many ways to partition the functionalities into software modules and processes. It is common for general-purpose robots to distinguish between low-level and high-level layers, and between typical responsibilities like perception, navigation, manipulation in a lower layer, and knowledge processing, action planning, and communication in a higher layer.

The control inputs that a human-aware navigation framework may consume is directly related to certain use cases. The most common robot navigation use case for autonomous robots is that an action selection module has decided that the robot base should next be in a different position than it is now, so the control input demands for the navigation framework to move the robot to a target location. However several other use-cases exist, and an exhaustive list cannot be given. The following list only serves to give an initial impression of the diversity of use cases that may be considered in the design of a navigation framework.

Possible commands:

- Create plan from start pose to goal state
- Move robot (and *n* guided persons) to a given goal state
- Replan under changed circumstances
- Freeze, stop all motion

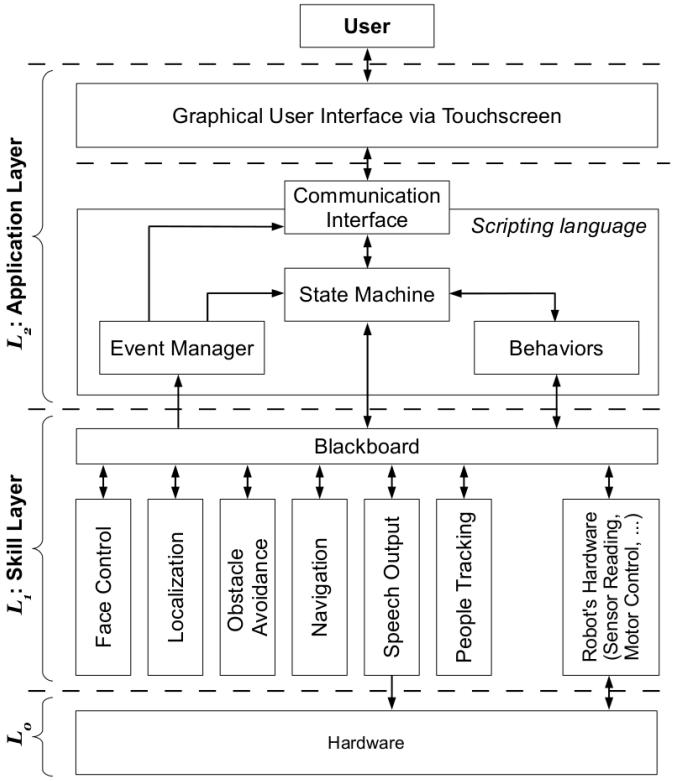


Figure 2.5: TOOMAS Overview, Figure taken from [51]

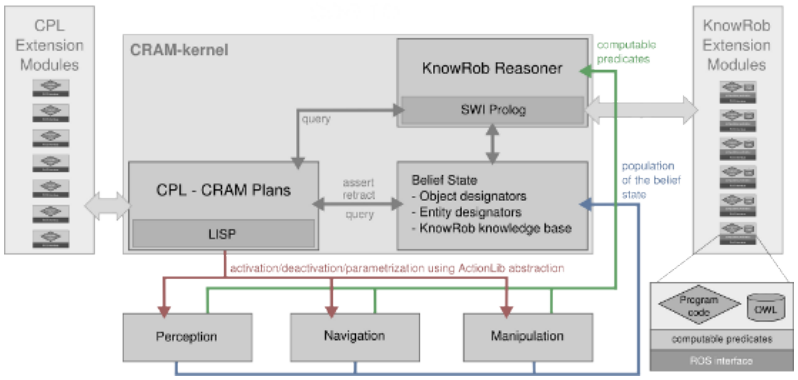


Figure 2.6: Cognitive Robot Abstract Machine (CRAM) Overview, taken from [10]

- Keep in formation with other agents
- Move according to given plan
- Intercept moving human
- Survey area
- Patrol area
- Stand by, giving way

The most common command for autonomous robot navigation is to move to a given location, usually as part of a larger ongoing task. There are also open-ended commands for the robot, such as following, exploring, patrolling, or merely standing by, waiting for further commands. "Standing by" is as interesting command as it still requires the robot to move in the HRI context, such as to give way to humans, or also to stand in a spot where the robot is easily accessible but not annoying.

General navigation frameworks have to deal with two core problems, finding a path to a desired location that has global qualities (e.g. the shortest path passing between known static obstacles) as well as reacting in time to unexpected obstacles and events while moving towards the goal. The former problem scales in response times with the size and resolution of the map, the latter problem requires fixed response times. This general problem leads to a common design of most general navigation frameworks into at least two separate planning processes, a global planning module and a collision avoidance module.

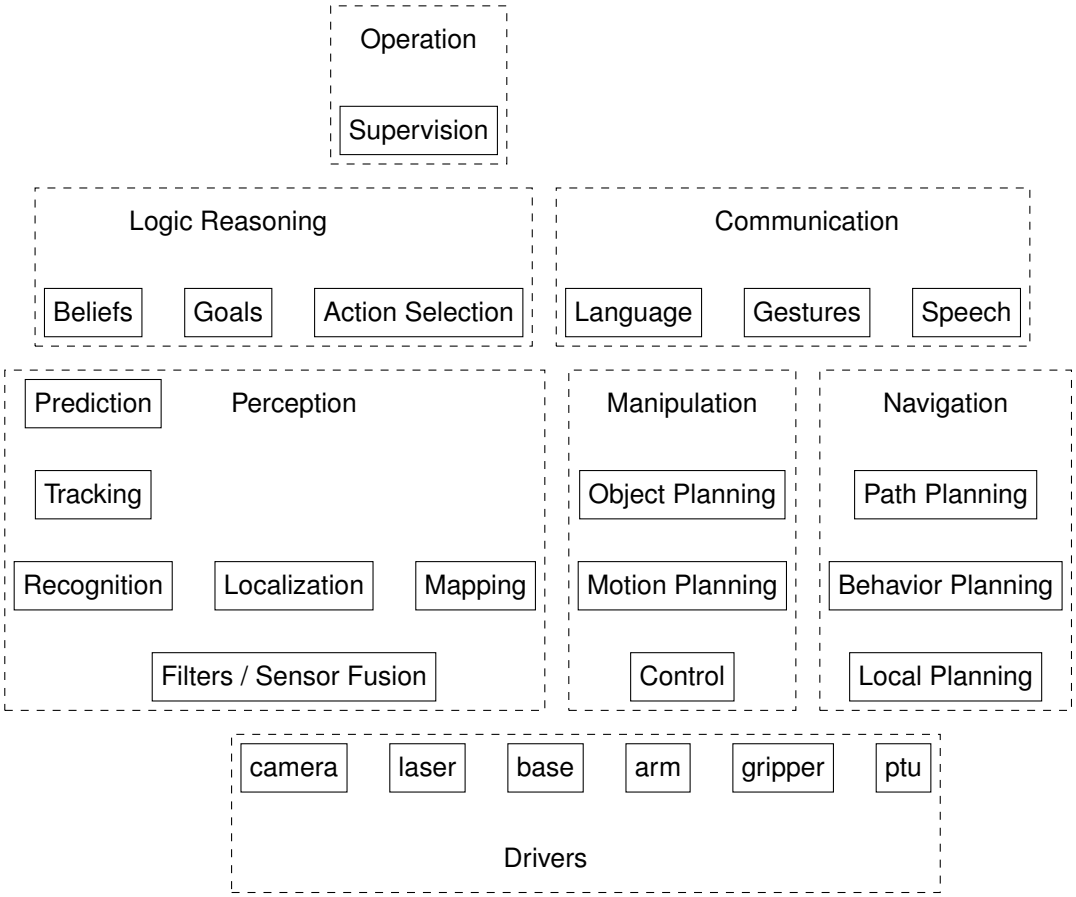


Figure 2.7: A sketch of the functionalities in a robot software architecture

A global planning module searches the space of all complete solutions for motion from start to goal, reasoning about quality. It is also called deliberate module. A collision avoidance module searches the space of possible motion commands for the immediate future only, to select a safe motion that adheres best to the global plan.

Based on observations of robot software frameworks, Figure 2.7 visualizes a possible set of functionalities for a single autonomous robots.

The mapping of functionalities to system processes, and their connections, allow too many variations for a general recommendation to be made. Still, at this abstract level it is possible to discuss the inputs a navigation framework has to process in general terms.

The functionalities in Figure 2.7 are grouped by more abstract categories of functionality. For work on human-aware navigation, the "Perception", "Logic Reasoning" and "Navigation" groups are the most important, though their relationship to the other groups must always be kept in mind. Perception, reasoning and acting are the three stages in the classical "sense-plan-act" (SPA) paradigm for robot motion control. In the SPA paradigm the three activities of sensing, reasoning, and acting followed each other in iterations as a single process. This paradigm is obsolete today, contemporary robots have independent processes of each kind running concurrently, and the planning category is optional, allowing for reactive aspects of behavior [15]. Still, the functionalities of robots are categorized into these three groups and the data flow is still usually a traversal of these three stages of computation. It is also notable that "planning" as a technique of thinking ahead is now used also for perception (e.g. to predict the actions of other agents), as well as during execution of high level plans, in the creation of low-level plans.

Sensing is important for navigation for the robot to know its own position in the world (localization), to observe obstacles to avoid, and in the case of HRI to identify and track individual humans in the environment. Reasoning synthesizes knowledge and produces plans. The activities go together as planning generates knowledge about the possibilities in the environment. As an example, whether a position in space can be reached by navigation within a time limit is knowledge that requires a navigation planner to be decided by reasoning.

For planning and acting, it is common to see a data flow that follows this sequence:

1. The robot selects an action (e.g. go to human)
2. The robot selects a pose for the action (e.g. in front of human)
3. The robot selects a path to the pose
4. The robot adapts an aspect of behavior while moving on the path

This sequence derives from robot tasks in static domains and non-HRI cases. It also has practical advantages for research platforms as the strict separation of planning phases allows for easier understanding of the robot behavior, and thus to tweak and bugfix software for experiments. However the separation

of these planning activities reduces the potential to globally optimize planned robot behavior, and to perform coordinated motions with other agents. As a consequence several authors merge some of the functionalities, or use a different order of steps than the small example above. But for many research purposes this is not a significant issue, as local optimization in individual planning aspects can better be studied when planning in isolation.

Accordingly the following sections review related work in human-friendly navigation as isolated problems of selecting a path, adapting behavior, and reacting quickly to minor changes.

2.4.2 Path planning

Path planning is the task of choosing a list of waypoints to follow on a map in order to optimize the performance of the robot according to some global target function, like the distance traveled. This is often called synonymously "navigation" in the narrow sense, as in "navigation systems" giving travel directions to car drivers. The minimum feature of a path planner is to find an end-to-end path from the current position to the goal. The path must be proven to be correct under the current assumptions about the world. Further optional features are to find the shortest path, or a path satisfying some other global property. Such a path is often called the "global plan" for navigation. Navigation as a reasoning activity is only necessary where a motion towards a goal cannot be achieved sufficiently well using reactive methods, such as turning towards a goal and moving straight. Reactive methods fail in environments that have sufficiently constraining obstacles like walls that also form dead ends. Navigation algorithms use maps of the environment representing blocked and available space. Humans can be taken into account as moving obstacles, as social agents not to disturb, or as partners in joint motion with particular interests.

The majority of work on human-aware path planning use graph-based search methods, with a graph representing states in a 2D map of the environment. In the graph each node is a point in which the robot can be without collision, and an edge between two nodes means the robot may travel between the nodes without

causing collision. The kinds of graph used in the surveyed research are square grids, arbitrary lattices, and expanding random trees with freely chosen edge angles. Another classical path planning structure is the Voronoj graph, but it no publication on HRI focused on that technique to solve a challenge in human-aware navigation.

Expanding random trees are usually used for search with higher dimensionality. They can have less memory consumption in average cases and may find non-optimal but feasible results in less time. While two publications use them [128, 137], their usage there for 2D navigation is not proven to be superior to grids or lattices. The concept has also been adapted for motion among dynamic obstacles by Fulgenzi [39, 122].

Grids automatically avoid duplicating states and have a fixed memory size, they are also simpler to program. The disadvantage of grids is that the paths do not take into account robot kinematic constraints. Lattices such as suggested in [88] are more suitable to represent kinematic constraints, and are thus probably superior to grids in most respects. However, they are rather new and more complex to construct than grids, so HRI publications have not yet adopted them. A lattice here means a graph that is built from motion primitives based on robot properties, such that an edge of the graph is a so-called motion-primitive. To prevent the graph to grow too quickly, the motion primitives are selected to build a regular structure, such that expanding a node often produces edges leading to existing nodes.

Regardless of grids, lattices or trees, global path planning uses a search method to find a sequence of states from the robot position to the goal that is considered optimal with regard to a cost function. A major improvement to plan results is also possible when planning with time. This allows finding solutions that static representations cannot find. On the other hand, adding the dimension of time exponentially increases the search space and can cause the robot to respond too slowly to changes.

An alternative to graph-based methods is navigation based on partially observable Markov Decision Processes (POMDP) [34]. This approach focuses on dealing with uncertainty, and is difficult to adapt to global quality criteria. Also

the approach does not scale well with the resolution of states. While POMDP-based navigation may generally have advantages, the technical effort to maintain such frameworks seems to have made it impracticable for research on human-aware navigation so far.

The next two subsections look at cost functions in detail, and then at temporal planning.

Global cost functions

Human-agnostic navigation will commonly choose the shortest or most energy efficient path in a graph. In HRI, the assumption is that the shortest or most energy efficient path is not necessarily the most desirable.

Instead in HRI the intention is to find a path that is also sufficiently comfortable, natural, legible, etc. to persons in the area. Some of these aspects can be achieved by avoiding certain behaviors known to cause stress, such as coming too close. Cost functions can identify such behaviors during planning and by increasing virtual costs, guide a planner to find better alternatives if such are available. A bulk of the work in path planning thus focuses on cost functions, even if not all quality criteria can be solved by applying cost functions.

This is a list of cost functions used in the publications, explained in more detail below:

- object padding [73, 137], Figure 2.8a
- object occlusion [23], Figure 2.8b
- hidden zones [135], Figure 2.8c
- zones of high/low noise [98, 99, 100]
- basic comfort distance [71, 73, 95, 128, 135], Figure 2.8d
- visibility [71, 128, 135], Figure 2.8e
- interaction regions [121, 128], Figure 2.8f
- pass on their left [107], Figure 2.8g
- space ahead for moving [84, 128, 149], Figure 2.8f
- discard compatible [81, 107]
- crowd density and velocity similarity [62]
- inertia [73]

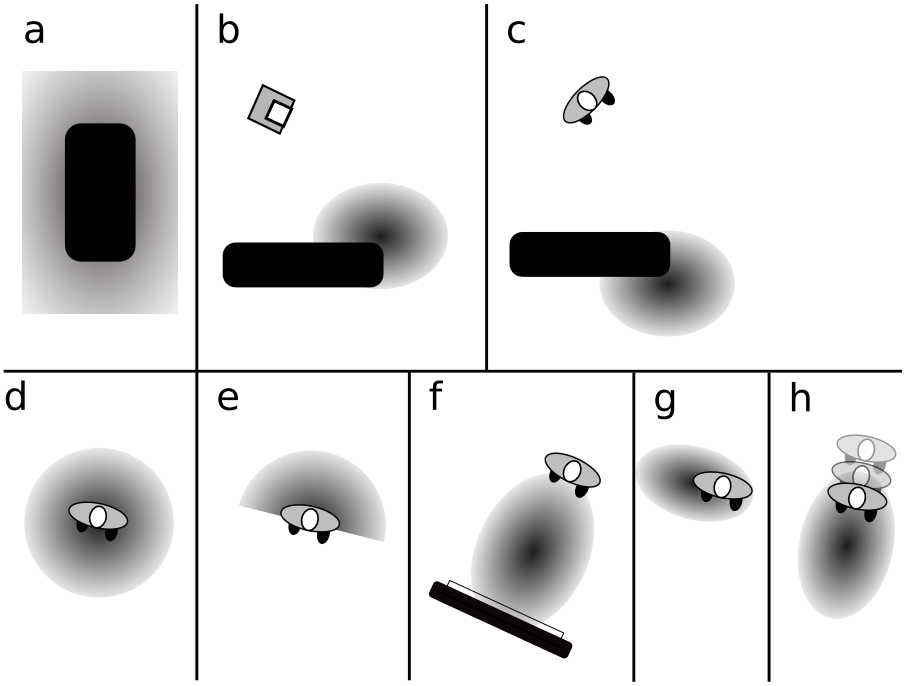


Figure 2.8: Schematic examples of separate cost functions, thick black areas are obstacles, the square in b is a robot, human in h is moving. Areas shaded in grey have costs, meaning the robot should prefer to avoid those, if possible. Shapes and sizes of cost function may vary and be context-dependent. All cost-functions can be combined.

Object padding [73, 137] as in Figure 2.8a describes cost regions around static obstacles so that a robot prefers to move at a greater distance from objects than what is strictly necessary for collision avoidance, unless the space is so limited it has to get close. The resulting behavior allows people in the environment to be less worried about the safety of the robot and obstacles.

Object occlusion costs [23] as in Figure 2.8b try to make the robot avoid motions along paths where the sensors of the robot cannot well perceive the area ahead. This relates in particular to convex corners of rooms, because the robot cannot sense humans behind the corner, and vice versa. Moving around the corner at a greater radius allows earlier coverage of the space behind the corner, and allows the robot to be seen earlier.

The social costs of hidden zones were described by Sisbot et al. [135] and this refers to the robot avoiding to appear surprisingly from behind objects close to

a human, as shown in Figure 2.8c. This is similar in its intention to object occlusion but requires the robot to know where a present human is. The costs can be discarded if the obstacle does not hide the robot body, even if the obstacle causes robot sensor occlusion.

Costs for object padding, object occlusion and hidden zones cannot replace safety features of the reactive layer of the robot. For purposes of safety, the robot velocity has to be adapted to be safe even when moving tightly around corners or close to obstacles, by adapting its velocity for example. This velocity adaptation remains a necessary feature of the reactive software components (not path planning). The cost functions can only reduce the likelihood of collisions caused by humans, where enough space is available.

The basic comfort distance costs [71, 73, 128, 135] cause a robot to avoid moving closely to humans where that is avoidable, see Figure 2.8d. The shape and size of such comfort costs can vary according to the context, and Luber et al. [95] even suggest to dynamically change those during a crossing situation.

Visibility costs [71, 128, 135] as in Figure 2.8e assume that humans prefer a robot to move where they can see the robot when the robot has to move close. This is similar to costs that make the robot approach a human from the front [71]. An opposing force are interaction region costs [121, 128], that assume a human prefers a robot not to cross the space between the human and an object the human interacts with, such as another human or a TV set (Figure 2.8f). This can be called territoriality, where persons claims free space for themselves. This pair of contradicting forces, visibility and territoriality, illustrates well how difficult it is to tweak and compare cost functions.

To pass humans on their left [107] is a social convention that may be tied to cultural traffic rules. Adding costs on the right of humans as in Figure 2.8g achieves this behavior in several circumstances like corridors. But it is not clear how these costs influence robot behavior in general.

The other cost functions presented here relate to robot motion among moving humans.

Several publications investigate the effects of defining costs in front of moving humans [84, 128] as in Figure 2.8h, in particular for non-temporal planning. At the same time, it is also apparent that for non-temporal planning, for humans moving in the same direction as the robot no costs should be applied in path planning [81, 107]. Instead, the robot may benefit from planning a path that follows a person if the robot wants to move in the same direction, in particular for moving in crowds. This relates most closely to natural motion rather than comfort or safety, as certain path deviations that emerge as a result of cost functions applied to an unsuitable context make the robot do unnatural motions. The approach by Ziebart et al. [149] uses an iterative planning approach that may eventually find paths being hindrance-free in time-space. Since the number of iterations is not bounded, this approach must sacrifice either correctness or reactivity in crowded situations. Crowd motion is also the focus of Henry et al. [62], who models a preference of path steps depending on crowd density and crowd motion.

The actual shape of individual cost functions is subject to tweaking. Most of them have growing costs as the distance to some area decreases. The growth can be modeled linearly, exponentially, sinusoid etc. Also proxemics research has shown that preferred distances between humans and robots depend on many context factors, and it is therefore a valid assumption that this equally holds for other behaviors captured by the social cost functions above.

Similarly, the combination of cost functions is subject to tweaking. Combinations used in publications are weighted sums [73, 128, 135], the maximum [128, 135], and context dependent enabling of costs [73, 81]. In publications cost function shapes and combinations were generally selected and tweaked manually, only few approaches used machine learning techniques [22, 62] for this purpose.

Temporal planning

Apart from work introducing cost functions for path finding algorithms, several authors investigated fundamental changes to path search in order to improve path planning of robots among humans tackling mainly the challenge of avoiding blocking each other's path, for navigation among moving humans. Temporal

planning has two main challenges: 1) predictions of future human motions decrease rapidly in accuracy, and 2) the added dimension of time exponentially increases planning times.

The early paper on human-aware navigation by Tadokoro et al. [138] describes an exploration of the search space using a genetic algorithm rather than classical search. However, they do not motivate that choice nor compare it.

The evacuation simulation planner used by Ohki et al. [110] uses the wavefront algorithms with addition of the dimension of time. The authors do not describe a means to avoid the scalability issues of adding a dimension to the algorithm.

An attempt at temporal global planning avoiding scalability issues is presented by Kushleyev et al. [88]. The idea here is to perform temporal planning for n steps ahead, where the authors suggest n to be determined by the time the prediction algorithm for moving obstacles can return meaningful values. A global planner can then perform static planning for the rest of the way up to the goal until the next replanning cycle.

Because the temporal planning challenge of scalability is so dominant, temporal planning is applied in local planning described in subsection 2.4.4 more often than in global planning. The local scope of local planning allows temporal planning to remain tractable.

An attempt to reduce the search space even with temporal planning is presented by Phillips and Likhachev in [116]. The temporal dimension in that work is segmented into safe and unsafe intervals instead of regularly discretized time, reducing the search space expansion. Gonzalez et al. [47] enhance that idea allowing for continuous cost functions as well, a prerequisite to apply social cost functions to search.

Before that, the next subsection describes methods to generate robot behavior considering people as interaction partners, not mere social obstacles.

2.4.3 Behavior Selection

Behavior is the result of planning and execution. General navigation behavior of a moving robot can be distinguished from the specific behavior of a robot when performing a task. The previous subsections described pose selection and path planning. Pose selection and path planning decide ways to move past humans avoiding unnecessary discomfort. Humans are thus treated as social obstacles, having properties to respect beyond those of non-human obstacles. So the interaction between robots and humans is regarded as a necessary evil if the robot has to perform a task, and the aim is to reduce the impact of interaction.

In behavior selection, the problems to be solved more strongly involve a person as interaction partners. Therefore interaction is not a necessary evil here, but interaction is the purpose of the robot. Research on navigation then focuses more on how to modulate robot base motion to best serve the interaction goal, rather than merely avoiding discomfort. Partly this reflects the aspect of sociability, partly this reflects the practical needs arising from interaction with a human.

On the more practical side, several authors use state machines to switch between behaviors such as approaching a human, following the human, patrolling, searching, standing in line, and so on [2, 49, 55, 113, 127].

Several authors address adequate approaching behavior for moving humans using a potential field [2, 53]. Althaus et al. [2] define three phases, approaching, keeping the distance, and moving away, for a lengthy situation of interaction. Using potential fields offers some advantages and some disadvantages over search based planning. Advantages are that reactions can be found in a timely fashion and there is no need for a fixed goal pose. Disadvantages are the problem of local minima and issues related to that.

Satake et al. [127] uses a custom estimator of whether the robot will be able to approach a walking human from the front.

The work by Chung et al. [23, 72] introduces a global planner for safety but also a computation of a safe velocity when moving towards an area that cannot yet be covered by the robot sensors. This maximum velocity is a constraint on

the local planner. A similar idea is presented by Krishna et al. [80] analyzing zones hidden from the robot field of view. The authors show how this analysis can help to constrain the local planner velocity, it can help a global planner in improving overall velocity, and it can be used to improve an existing global path reactively.

The concept of velocity constraint modulation can also be transferred to other areas where velocity limits can be changed, like for allowing increased velocities in corridors [148]. Similarly for a robot to guide a human being, the local planner can be constrained by a model of the joint motion [32].

Behavior selection can be regarded as a modification of global planning, we decided to treat it in an own subsection as the focus seems different enough. With path and behavior selection in place, a robot has a high-level plan of where to move next. What remains is the task to control the robot motors and monitor all sensors to follow this plan while preventing collisions, as described in the next subsection.

2.4.4 Local Planning

Local planning is the task of finding commands for the robot actuators for the immediate future. It is also called "reactive planning" and "collision avoidance". The requirement of local planning is to assure timely responses to changes, avoiding unwanted situations (such as collisions). A particular concern of local planning are the hard time constraints that have to be satisfied. Methods usually ensure that a controller response is given within a time limit by thinking only a limited time ahead, hence the name "local" planner. Apart from a time limit, this also limits the kind and amount of reasoning possible. There are several kinds of local planners as summarized by Fraichard [38]. These are shown as sketches in Figure 2.9, where Figure 2.9a shows the situation of a human moving forward in front of the robot and the task of the robot is to move ahead.

The Nearness Diagram (ND) method shown in Figure 2.9b uses range measurements at several discrete angles around the robot to find currently available open space, and moves towards gaps [147]. It can be extended to also detect

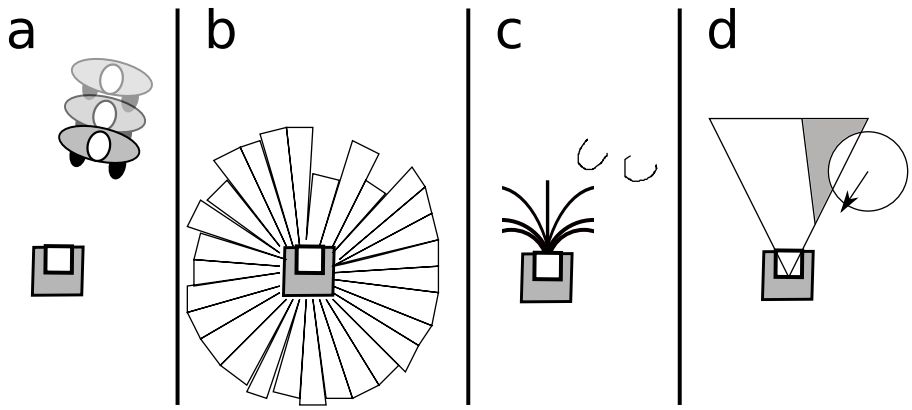


Figure 2.9: Schematic representation of local planning approaches. a: Real world situation with a person and a robot. b: Nearness Diagram approach taking into account low-resolution sensor data. c: DWA approach sampling many candidate trajectories taking into account sensor data and robot kinodynamics. d: velocity approach taking into account idealized moving obstacle interpreted from sensor data. All approaches exist in several variants not shown here.

humans by the shape of legs and increase safety by avoiding to move within a certain radius [89, 64]. However, this approach does not use prediction of human motions, nor take into account the dynamics of the robot. This reduces the maximally possible safe velocity of the robot compared to other local planners.

To take into account the dynamic properties of a robot, trajectory sampling algorithms represented in Figure 2.9c such as the dynamic window approach (DWA) [36] have been developed. In sampling algorithms the local planner calculates the future positions of the robot under different velocity commands, prunes those that are in collision, and selects the one of the remaining trajectories that best fits some objective function. This allows for more safety and thus for higher robot velocities than the ND approach. However, this still does not take into account moving obstacles, and can therefore not prevent collisions in dynamic situations.

The Velocity Obstacle (VO) approach assumes perfect knowledge about other agents' future motions and finds an analytic solution that prevents collisions that simple sampling algorithms could not prevent. This is shown in Figure 2.9d. Finding a solution analytically rather than by sampling may require less com-

putational effort, and may thus scale better with the time horizon. On the other hand it is more difficult to implement analytical solutions with nonlinear trajectories, uncertainty, and cost functions. Therefore, in human-aware navigation research, the VO approach is found more often for simulations than for real-world robot prototypes, see [21, 40] for examples. In such simulations the dynamic obstacles are usually on a linear path, but research on non-linearly moving obstacles also exists, though without focus on human-robot interaction [91].

It is useful to consider that persons in the environment cooperate, and will help to avoid collisions. This can be used for finding solutions with reflective or reciprocal collision avoidance [76, 143].

The DWA approach can be extended in a straightforward way to take into account moving obstacles as well, such as shown by Hoeller et al. [63] for a robot following a human guide.

The same problem was solved using the VO approach in Prassler et al. [117] for a robot trying to maintain a relative position to a person while being guided by that person.

Both approaches do not take into account the uncertainty of human predictions. Merely growing the size of obstacles over time to represent uncertainty leads to the freezing robot problem. If the prediction of the future human path is represented by growing occupancy regions, and the planner is forbidden to enter such regions, then the robot may be unable to find any allowed space left, and not move on.

This has been described by Trautman et al. [142], who suggest a solution using an Interaction Gaussian Process Model.

A similar attempt is made by Althoff et al. [3] to consider uncertainty in bilateral sampling and selection of most likely collision free trajectories. While the concept is promising, the approach requires a lot of processing was not attempted in realtime.

Kuderer et al. [87] presented an approach of learning joint trajectories predictions from observing multi-human and human-robot situations. They also con-

sider using this prediction as a control framework for the robot, to follow the predicted natural behavior. Given that it does not scale well over space, it can be counted as local planning method.

In sum, the collected list of publications on local planners shows that the main problem of human-aware robot navigation solved in local planners is the collision avoidance. So comfort and naturalness of motion have not been researched in the context of local planners. This may be due to the tendency of local planners to oscillate and be blocked in local minima when considering large obstacles, such as virtual obstacles created by social cost functions as in Subsection 2.4.2 on global planning.

2.4.5 Summary of related work

This section presented related work on human-aware navigation planning. The functionalities researched in literature were classified with the help of a schematic architecture. Planning modules which modify robot behavior in the presence of humans have been categorized into 2D path planning, behavior planning and reactive planning modules.

For path planning, the bulk of the literature considers planning robot paths such that the robot behavior becomes more acceptable by avoiding areas in space that make humans feel uncomfortable one way or the other. Some work also considers to optimize paths when going towards a person for interaction, as closely approaching a person is then inevitable, and special care needs to be taken. The problem of path planning among moving humans (not dense crowds) is approached by attempting to find paths that avoid an affordance space in front of a moving human.

Behavior selection and modification has mostly be done in literature for navigation situations with special-purpose robots, such as approaching people in the context of a robot working in a mall.

For local planning, the literature survey shows approaches beyond mere collision avoidance with moving objects for safety. Taking into account moving hu-

mans as intelligent agents rather than thoughtless moving objects allows a robot to find cooperative solutions in situations of conflict. Also modifications to local planning are required for a robot to move appropriately in formation, when guiding or following a person.

2.5 The HANP planner

This thesis relies on previous work on human-aware navigation. In particular it reuses HANP, which is a human-aware global path planner described and implemented by Sisbot et al. [135] at LAAS CNRS. The existing state of HANP before this thesis is described here in detail, to distinguish existing work from the contributions of this thesis.

(a) Path segments considered using 16 instead of 8 neighboring cells

(b) When selecting any given next path segment, it may help to only consider forward leading edges

Figure 2.10: Neighboring cells considered in grid-based path planning.

HANP uses a grid representation of the world. In grid-based methods a grid of square cells is build from sensor data, identifying "allowed" grid cells into which the robot footprint fits as opposed to "blocked" cells where the robot cannot be, near walls and furniture. Given such a grid, a global planner returns a sequence of waypoints to follow, one neighboring the next in the grid, such that if the robot moves through the waypoints in sequence, it will reach the goal position without collisions. For all experiments done in this thesis, HANP was extended to search through 16 neighboring cells as shown in Figure 2.10a, rather than 8 directions in the original HANP publications. This not only results in visually smoother paths, but also allows generally more natural paths in cases where the walls of the environment are not well aligned with the grid. Additionally, later in this thesis a path will be constructed of segments that avoid sharp turns, meaning considering a previous path segment, the next path segment cannot turn at a 90°

angle or more, as shown in Figure 2.10b. Both kind of changes to HANP, using 16 neighboring cells and avoiding sharp turns have theoretical impact on the search spaces, meaning with those extensions, a path planner can find solutions to some geometrical problems that were unsolvable before, but may not find solutions that existed before. In the practice of human-aware navigation those impacts are usually irrelevant, as the space of the robot is the same space that humans also use, and such space is generally designed to be wide enough to avoid geometrical corner-cases. And where the space is more constrained, usually a reactive local planner is a better solution than path planning.

Using the extensions, the generated paths become easier to follow at a stable forward velocity obeying kinodynamic constraints.

Using grids for navigation planning is common, the main used alternatives are Voronoi graphs, other lattices, and randomized graphs such as in probabilistic roadmaps, a brief comparison is given in section 2.4.2.

Navigating in the presence of humans compared to navigation in other environments has two additional challenges: 1) taking into account social preferences, a feeling of safety and in general the legibility of the robot's behavior and 2) the highly dynamic nature of such environments. The first challenge was the main motivation for the Human-Aware Navigation Planner (HANP) [135], which makes sure that the human is not just an obstacle, but that the robot respects additional constraints, for example to approach a person from the front rather than from the back.

HANP makes the simplifying assumption that the present humans are static, which is realistic mostly for a sitting person being served by a robot. When used in an environment with moving humans HANP does not define adaptations to those dynamics.

HANP [135] generates a path for a robot to move in the presence of humans. Unlike non-human-aware path planners, it considers additional constraints during motion planning. It defines cost functions representing human discomfort based on the concept of proxemics, which was introduced by the anthropologist Edward T. Hall [52]. It is the most common attempt in robotics to describe how closely agents should be approached in different contexts to avoid discomfort.

(a) Safety Costs (b) Visibility Costs (c) Hidden Zones Costs

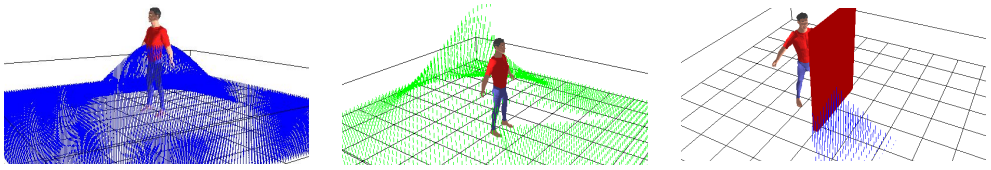


Figure 2.11: Cost functions for HANP A* search

Based on his work, conceptual regions or "bubbles" around humans can intersect and cause reactions such as discomfort. Based on user studies on human-robot space sharing [26], [77] a cost function defines areas of increased social costs, which is shown in Figure 2.11. With this function it is possible to model different social constraints on the robot movement like

safety: modeled by a Gaussian cost function around the human, Figure 2.11a;

visibility: modeled as the effort of the person to track the robot. A robot thus avoiding moving close while out of sight increases human comfort, Figure 2.11b;

hidden zones: evaluates the space behind large objects which is not visible for a person, taking into account the distance between the person and the object. The rationale here is that a robot should avoid regions where it would surprise a human moving around corners, Figure 2.11c.

Using the combination of all cost functions in a discretized representation of the world, HANP uses A* search to compute an optimal path in terms of human comfort.

For a single grid cell w_i and each human H, the comfort cost function ς_{Static} returns a value based on psychological considerations derived from a priori concepts such as proxemics or based on empirical data, more examples of such cost function are given in section 2.4.2 and shown in figure 2.8.

Equation 2.1 shows how such a cost function can be based on several specialized functions f_j, such as for visibility or safety as depicted in Figure 2.11.

$$\varsigma_{Static}(H, w_i) = \max(f_1(H, w_i), f_2(H, w_i)..., f_k(H, w_i)) \tag{2.1}$$

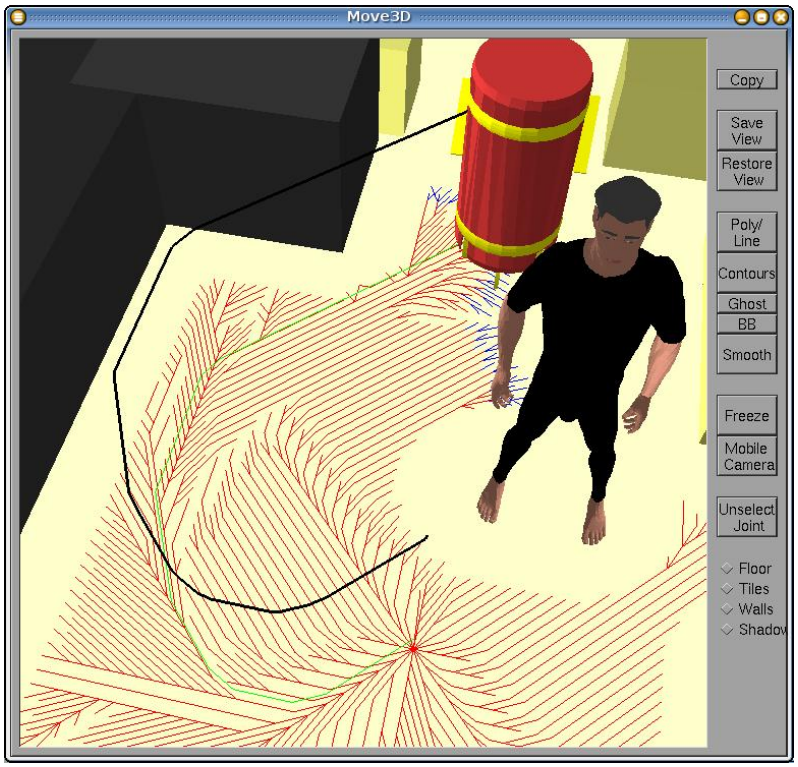


Figure 2.12: A* search visualization in Move3D, planning for safety and comfort. Initially the robot was in front of the human, the red lines on the floor illustrate the positions expanded during search. The planner did not consider moving "through" the human, and the resulting path (black curve) maximizes the distance to the human.

Following the cost function given in HANP [133, page 31] we define the comfort costs σ for a set of n present humans $\mathcal{H} = \{H_1, \dots H_n\}$ as follows:

$$\sigma(\mathcal{H}, w_i) = \max_{H_j \in \mathcal{H}} (\varsigma_{Static}(H_j, w_i)) \tag{2.2}$$

Definition 1. *The costs of a path P in presence of humans $\mathcal{H}$ of length l are a weighted sum of distance costs and social costs with weights ω_δ and ω_σ.*

$$\gamma(\mathcal{H}, P) = \sum_{i=2}^{l} (\omega_\delta \delta(w_{i-1}, w_i) + \omega_\sigma \sigma(\mathcal{H}, w_i)) \tag{2.3}$$

HANP uses the Euclidean distance as distance measure δ. The value of ω_δ and ω_σ depends on the ranges of the functions f_j and δ, and there is currently

no formal way to determine optimal values. We used a ratio of $\omega_\delta/\omega_\sigma = 10$. The path with minimal costs is assumed to be the most human-aware, acceptable path.

This concludes the summary of the original HANP planner and the modifications made for this thesis.

2.6 Action selection for HRI

This section gives an overview of concepts and related work on action selection in robotics. Previous sections have introduced the notions of plans and planning, and human-aware planning for the purpose of human-friendly navigation. Action selection refers to processes that decide for a robot which actions to perform to bring about certain changes in the environment, meaning which objects to manipulate, and how and when to do so.

The challenges specific to HRI have been mentioned in Section 1.4, such as acting in environments with unpredictable dynamics, having to produce socially acceptable behavior, using social interactions, and producing responses to events in a timely fashion. These challenges affect task planning as much as navigation planning.

Classical research areas of robotics are perception, navigation and manipulation. These research areas are important both for intelligent autonomous robots as well as for non-autonomous production machines. Within that context the research may be called "low-level" research, in contrast to research on how to make best use of perception, navigation and manipulation abilities, which depends on those activities, and in terms of dependency layers may thus be called "high-level" research.

To realize on day the vision of an independently acting robot assistant or companion, the low level abilities of a robot are combined via control structures for agency. So a robot would have to act as an intelligent agent to serve that purpose. The abilities or perception, navigation and manipulation can be combined to perform changes to an environment, and an important part of agency is the ability

to decide what changes to perform to an environment to achieve certain goals. Plan-driven behavior is one approach to make such decisions. This approach uses plans to drive robot action selection, so a plan will decide which objects to grasp and what to do with objects, as an example to set the table in a household. The other main alternatives to plan driven behavior are reactive behaviors and learned behaviors, but this thesis is focused on plan-driven behavior.

Planning for action selection in general requires reasoning about the effects of action ahead of time. Since this thesis focuses on robots build to help with daily life activities and chores, the action selection is based on notions of grasping and moving items, as opposed to other planning domains like chemical processing or scheduling of many independent processes. So the actions performed can loosely be described as "pick-and-place" actions, even when talking about some related actions such as opening doors or twisting valves.

Action selection for pick-and-place tasks has a long tradition in computer science, here is just a brief glance at it. The basic premise is that actions in this context have preconditions and effects. A precondition for picking up an object may be that there is no object on top of it, and an effect is that it is in the robot hand after the action. Examples of classical AI planning languages using simple symbolic representations are STRIPS [33] and PDDL [103, 37]. Those languages allow generating a plan as a partially ordered sequence of actions, based on reasoning. Using such plan languages, a plan is given in form of a tupel with a set of tasks $t \in T$ a partial ordering $\leq_T$ of steps and with causal links $(t, \leq_T, l_T)$. A causal link specifies the post-condition of a task to be a precondition of another task, preventing yet other parallel tasks to destroy the postcondition and thus render the plan invalid.

Additional orderings can be the result of complementary considerations, such as about efficiency, risk, or social appropriateness. Like carrying a trashbag in one hand while serving a sandwich in another hand may not violate a causal link but may not be socially acceptable.

Competing with those planning languages languages are languages that describe Hierarchical Task Networks (HTN) [30], which represent similar domain knowledge in a slightly different way. In HTNs the human experts designing

the primitive actions can specify compound actions, thus encoding how abstract action can best be refined into primitive actions. Both approaches have been extended over the years adapting the planners to various special cases, like dealing with quantifiers and conditional effects.

Such symbolic effects can be specified for simple theoretical domains, whereas the more realistic the application becomes, the harder it is to specify all preconditions and effects. In robotics as an example a precondition may be that there exists an arm configuration and a suitable grasp of the object, that can be realized for the robot given all other constraints [50, 18]. Other details of realism are uncertainty about object locations, and possibilities of action failures.

The approaches can incorporate the notion of costs for actions, which can serve to find cost-optimal plans. For HRI, social costs have been added to a HTN planner resulting in the HATP planner [1].

A common problem with actually applying task planners in robotics however is that planning algorithms do not scale well in terms of performance, and do not provide robust control of robot behavior. A plan in the classical approach is a sequence of actions, an action has a type and variable bindings (e.g. Picking up as type, and the object to pick up). Further infrastructure is then required to execute such a plan, meaning usually a controller for each action, and a task execution supervision module that determines robot behavior in cases of failures or unexpected events. This approach works well for static domains with high success rates, such as industrial robotics, but does not fit the circumstances of assistive HRI.

As an alternative to the separation of strictly declarative plan languages and action controllers for execution, the concept of Structured Reactive Controllers was presented by Beetz et al. [7]. The later contributions of this thesis rely on SRCs, so the concepts related to SRC are presented in more details in the next section.

2.6.1 Structured Reactive Controllers

Structured Reactive Controllers (SRC) can be described as a design paradigm for robot control architectures. It competes with other design paradigms such as 3-layer architectures or Behavior-driven architecture (BDA) [42].

In contrast to classical 3-layer architectures an SRC does not separate the plan definition from the action execution. Prof. Beetz describes the organization of Structured Reactive Controllers as follows: "The main components of a structured reactive controller are the process modules, the fluents, the structured reactive plan, and the RPL runtime system." [7]. In this framework process modules provide low-level operations of perception or manipulation. They should provide an interface to basic robot capabilities that is uniform across different robot types. Fluents are dynamic data structures for inter-process communication. They provide operations to monitor value changes caused by other processes in the system. These two concepts are not exclusive to SRCs. What sets SRCs apart from other architecture paradigms are the structured reactive plans, written in the RPL language.

A Structured Reactive Plan (SRP) is a software program, unlike classical plans which are directed graphs with symbolic actions as nodes. The RPL languages facilitates expressing concurrent software behavior and inter-process communication. The SRP spans both action composition of robot behavior, as well as many details of action control. This enables to encode in the plan many immediate reactions to unexpected events and failures, achieving highly reactive robot behavior across actions. There is no planning algorithm available that will create SRPs based only on he definition of process modules, or actions. While classical Plans can be transformed into SRPs, the core value of SRPs is in the ability to express reactive behaviors in addition to the minimally required behavior for the purely static and deterministic cases. SRPs can be manually created defining robot behavior in a "hard-coded" way, using the experience and domain-knowledge of human programmers to create robust plans. Such plans can be modularized and organized in plan libraries thanks to additional support infrastructure and language constructs.

Such a plan library [106] can contain SRPs for low level goals such as grasping an object, or for high-level goals such as setting the table. high-level goals can be composed of low-level goals for reuse, in the same way that functions invoke each other in programming languages.

For the task of setting a table that means with a classical planning a approach, a planner would create a sequence of actions to perform until the table is set, and these actions would then be executed one by one. An additional supervision process needs to define for the context of each action execution what environment state needs to be monitored for changes. And on an action failure, the most likely reaction would be to replan.

With an SRC however, an SRP for setting a table would be selected and executed, and the SRP itself would define what environment state needs to be monitored in the context of table-setting, and can specify immediate reactions to several kinds of failures, reacting differently for the same kind of failure depending on the table setting context.

This approach of driving robot behavior via a library of plans rather than plans created at runtime can pragmatically be started by hand-crafting several plans for a robot. In a way the paradigm however reflects agent behavior that solves problems by applying solutions that worked before, as a learning approach. Consequentially the idea of a plan library has been augmented with several ideas of incorporating machine learning in the evolution of the set of plans, such as TRANER [105] or RoLL [74].

Another important concept developed for SRPs are designators. All plans need to relate to real world actions in some way. In classical planning, all real world objects usually have to be known before plan construction, their properties (such as their location) will be used to build a correct and optimal plan. Similarly it is typical to define locations to place objects, other agents, standard robot poses and so on. An action in a plan can then refer to those real world concepts (such as put Cup with id 7 to location with id 13). For the assistive HRI scenarios however, the set of world objects is not defined as well, the objects in the environment are not generally knowable for the robot, and their position can be unknown or shift. Environments for humans are designed with human modalities and common

knowledge in mind, so cupboards often only reveal their contents after opening their doors. For a robot having to act in such an environment, this means that the real-world entities that have to be manipulated eventually are often not known at plan execution time, so no plan can refer to them.

The paradigm of SRCs allows to define plans under incomplete knowledge. The concept to bridge this gap of knowledge are designators. A designator is a description of a real-world concept, rather than a symbolic identity. So a plan to fetch a can of beer for a person may just use a designator like "A can of beer", instead of a fixed identity like "The can of beer with ID 142235". Since humans constantly use designators in natural language, the concept of designators has received considerable attention in philosophy of language, e.g. by Kriple [79]. He distinguishes rigid from non-rigid(flaccid) designators. Rigid ones always refer to the same concept in the world (such as social security numbers for persons or GPS coordinates for locations), whereas non-rigid ones may refer to different real world concepts. Both kind of designators are relevant to robotics, though the non-rigid ones are the more peculiar ones. The current incarnation of SRCs is part of a framework called "Cognitive Robot Abstract Machine" (CRAM), and publications on this framework explain the concept of designators as well [10, 8].

As some examples, designators may look like this in LISP dialect:

```
1 (object (type cup) (color blue) (on table))
2 (location (to see) (obj CUP-BLUE))
```

The first designator describes a blue cup that is on a table, the second the location where something referenced by the variable CUP-BLUE can be seen by the robot sensors. Both designators may be resolved to several real world objects or locations, or to none, depending on the circumstances.

As an example for a plan snippet written in RPL, Listing 2.1 shows a plan body executing a turn of a knob to adjust the temperature of a hot plate in a kitchen. In line 11 a lower-level plan is invoked to turn an entity in the world to a certain angle. This invocation is embedded in a "with-failure-handling" construct (line 4) that not just defines error handling, but a "monitor" block with code to execute concurrently to detect events that count as failures. In this case this block raises

an error when a fluent changes its value. The entity to turn is defined by non-rigid designators in lines 2 and 3.

Listing 2.1: Fictional plan snippet to show RPL features.

```
1  (with-designators (
2        (hot-plate '(an entity (type hot-plate)))
3        (regulator '(the entity (regulating-temperature hot-plate))))
4    (with-failure-handling failure ()
5      (recover ( T
6                 (handle-plan-failure :entity regulator))
7      (monitor
8        (whenever (slippage-fluent)
9           (fail :class gripper-slipped-failure)))
10     (perform
11       (achieve (make-instance 'entity-turned
12         :entity regulator
13         :angle (temperature->degree temperature)))))))
```

The interesting aspect of non-rigid designators is that they represent a decision to make in case several objects in the world correspond to the designator. Like all decisions during runtime, these decisions can impact the acceptability of a robot when acting near persons, and the quality of that decision is subject to changes when the environment changes, as it does frequently when acting near humans.

The common strategy for binding designators to real world objects (or percepts of those), is to defer the decision during runtime when an action needs to be done, and thus be robust to changes in the environment between the beginning of a plan and the respective action. This is a compromise between early and late commitment. Early commitment has the benefit that global optimizations of a plan are possible once decisions have been taken. Late commitment has the benefit that more information can be gathered to make a better decision than with early commitment. In either commitment mode, it is most common to only reconsider a decision in the event of a failure. As shown later in this thesis, for HRI it is even better to combine both early commitment as well as free reconsideration. That means continuously monitoring taken decisions to find opportunities to improve behavior by changing a previous decision.

The vision of CRAM as a project is to eventually produce a library of plans for assistant robots which are robust against many variations and failures. Whereas

in classical robot architectures, a plan is usually a custom-made solution to a given problem and discarded after execution or failures, a CRAM plan ideally works for many common situations and includes appropriate behavior for many common failures. The philosophy behind that idea is that human environments are usually structured already in such a way to allow achieving most common chores by applying a routine approach, rather than having to plan. So in order to clean an apartment, the routine approach might be to stow away all things lying on the floor, clearing table surfaces, wiping the surfaces, vacuuming the floor and wet-sweep the floor. It is generally unnecessary to use a domain specific problem solver defining the dustiness of surfaces, and considering applying all kinds of items on the surfaces (knifes, forks, crayons, etc), considering all possible configurations of objects in the household etc. In a joint action scenario with humans and robots it would be particularly annoying if the robot was creative for the chores, rather than following a well-established routine. So an CRAM plan rather tries to represent such a routine approach, that works well under different conditions,and even in different households. The RoboEarth project [9] demonstrates this vision of multiple robots sharing knowledge about both environments and procedures.

This overview of Structured Reactive Controllers and reactive plans should be sufficient to understand the extensions for opportunism in HRI in chapter 4.

2.7 Summary of concepts

This concludes the chapter on the concepts of planning in the context of HRI. The activity of planning was informally introduced as thinking about the best sequence of actions ahead of time. Plans can be created by autonomous planning algorithms, also called planners. Plans are the input driving plan-driven agents who act by following the plan. In task planning for HRI, plans are usually partially ordered sequences of abstract actions, that have to be executed e.g. by using lower-level planning techniques. In navigation for HRI, plans are usually sequences of waypoints in a 2d graph representing the robot environment.

In the context of HRI, planning is often modified such that plans are selected giving some weight to human-related qualities such as comfort or naturalness,

rather than only maximum efficiency. In navigation planning most of the existing work is based on proper distancing behaviors or avoiding other areas inappropriate for the robot. A second topic of large interest in literature has been to define the best path to approach a non-moving person.

Tasks with explicit and dynamic interaction such as guiding or following are commonly solved with reactive (local) planning methods that only look a limited time ahead, but achieve suitable response times.

The rest of this thesis shows improvements on the state of the art by considering "legibility" or robot motion, in particular investigating dynamic situations where both rigidly sticking with a current plan or switching plans often can create confusing robot behavior.

Chapter 3

Contributions to navigation planning

This chapter provides my own contributions to human-aware navigation planning, the following chapter those to execution of action plans in the presence of humans.

All work for this thesis used a navigation framework that is split into a global path planner and a local planner following a global path. As global path planner, HANP has been used as described in section 2.5, with several modifications.

However, given the literature research from section 2.4, it is obvious that considering path planning and obstacle avoidance is insufficient to enable a robot to perform a broad range of navigation tasks, in particular among moving humans, such as moving in dense crowds or in formations. Therefore the first section of this chapter accumulates ideas and experiences into a generalized human-aware navigation framework, such that the results gather from the actual framework we used can be seen in a global context.

3.1 A Human-Aware Navigation Planning Framework

This section presents a recommendation for a human-aware robot performing tasks requiring navigation among humans.

The framework defines inputs and outputs to each module in particular with respect to perception of humans. The intended domain of such a stack is restricted to realistic domains for mobile manipulators. This framework requires the robot operations to be limited to a finitely sized known area. For that area, the

robot must have a map that represents the surfaces the robot may travel through, and the robot must be capable to localize itself in that area. The potential set of moving objects in the environment, their shape and respective maximum speed must be known to be taken into account by the robot planning activities (e.g. whether there are cars moving in the environment, or only pedestrians). Example deployments for such robots are buildings or building clusters, like Museums, shopping malls, Corporate office sites, factories, airports, etc.

This means the framework is not designed to serve purposes like controlling car-like robots in cities, rough terrain outdoor robots or underwater and air robots. Similarly, purposes like simulation of crowd movement or computer games may have different requirements, and are not considered here.

There are several kinds of planning approaches for both global and local planning. The list of possible variants seems infinite, and in literature the variants are also quite numerous, so this section merely attempts to describe several examples to convey the extend of variability. A main variation point is the search space considered. E.g. planners can search in continuous or discrete spaces, consider temporal planning. Discrete search space representations are most frequently used, and temporal planning is usually avoided due to the combinatorial explosion of the search space. In human-aware navigation most frequently the robot is assumed to have a rigid torso and arms during motions, as opposed to planning complex motion through obstacle courses requiring torso and arm motions. Again, taking into account more actuators causes a combinatorial explosion of the search space.

The geometric space of search is discretized as a graph for all global planners. The shape of the graph can be grid-like, random trees, or sparse non-symmetric graphs like Voronoj graphs. The edges in the graph are commonly linear, but may be curved for more realistic or smoothed planning. The algorithms to find solutions in such graphs are not specific to HRI, only the cost function modeling path preferences are specific to HRI.

Another key difference between planners in literature are the semantics of its output. The output can be intended to be followed strictly, or have a supporting purpose to lower-level planners. In any case the output of a planner will be a

Responsibility	Planning layer
Choose collision-free trajectory	global (trajectory) planner
Choose motion command to follow trajectory	local planner (controller)

Figure 3.1: Typical split of responsibilities for navigation in known static environments

plan (in the case of planning success), that can be represented as a list of graph nodes and edges in the graph used to explore the geometric space, effectively an acyclic connected chain-like graph. However the interpretation of that chain-like graph can vary a lot, depending on the general contract between the global planner and lower level planners established by the navigation framework. The list of graph edges may have to be interpreted as a plan complete in every detail of actions for the robot, meaning that the edges between the graph nodes completely describe the motion to be executed. The graph node list is then interpreted as a trajectory to execute, as a function $traj(t) -> P$ mapping time to robot positions. In that case lower level planners have little freedom in their decision other than making sure the robot follows the plan strictly, and safely aborting the motion on unexpected events. This is depicted in figure 3.1. This split works well for static environments where it can be safely assumed that no unexpected events will happen. In such environments, the global planner can optimize even small details like jerk of motions, and plan for complex rotations to pass through very confined areas.

However as an alternative, the chain of graph edges can also be regarded as a mere path instead of a trajectory, meaning lower level planners have the freedom to decide themselves at what time and with what velocity the robot advances on the edge of the graph. Finally the chain-like graph may also be interpreted as a mere list of waypoints, meaning lower level planners may also decide by themselves on the path to choose between the waypoints, based on some contract of how far the lower level planners may deviate from the graph edges. Waypoints might even be areas rather than points, such as room segments of a building, where lower level planners merely have to traverse the areas in order.

This split between global and local planning is show in Figure 3.2, and common in dynamic situations where unexpected events are assumed to happen often enough. As another extreme, environments with too many unexpected

Responsibility	Planning layer
Choose waypoints for efficient traversal of buildings	global waypoint planner
Choose motion command towards next waypoint while avoiding (moving) obstacles reactively	local planner (obstacle avoidance)

Figure 3.2: Typical split of responsibilities for navigation in dynamic environments

changes may benefit from not using any global planner at all, but moving only reactively. A robot that needs to act in both extremely dynamic and very static environments may benefit from being able to switch between different splits of global and reactive planning.

The local-planning modules used with any given global planner vary accordingly in what decisions they make autonomously.

For HRI environments, it is useful to allow lower-level planners larger degree of freedom to make decisions, as humans frequently move in unpredicted ways, which can make any plan made so far invalid. The disadvantage of using flexible lower-level planners is that the resulting global robot behavior becomes less predictable than in the cases where a low-level controller strictly follows a trajectory.

One consequence is that the quality aspects the global planner considered may not be exhibited in the robot behavior, given that a lower-level planner may deviate too much from the expectations of the global planner. So the contract between higher and lower-level planners may also restricts the quality criteria a global planner can usefully apply, but allows more potential for quality criteria in lower level planners, to react to unexpected changes in more desirable ways.

In the state-of-the-art split between global and local planner for domains with unpredictable moving obstacles, the global plan serves as advice to a local planner giving a waypoint to reach, such as in Figure 3.2. The local planner in that arrangement is expected to be able to finds its way to the next waypoint only using only limited (local) knowledge about the environment. The purpose of the global planner then is to find a sequence of waypoints using global information

such that the given local planner is able to always find its way from one waypoint to the next using only local information. The actual distance between waypoints can still vary, for example, between a few centimeters and several meters, even for typical robots in research literature. In the context of HRI, an arrangement of planners is desired that can make the robot exhibit human-friendly behavior, as an example taking social distances. To plan for an appropriate distance from a perceived human can be made a responsibility of either the global or the local planner. If the design decision is to make it a property of the global planner, then the output of the global planner need to specify a plan that, if followed, will produce the desired behavior. For that, a contract is required that makes the local planner follow the plan closely. If on the other hand a design decision is made for the local planner to plan for appropriate social distances, then the global planner may provide waypoints in large distances, and the local planner is free to choose alternative local paths around humans. However the local planner typically used for obstacle avoidance should possibly not be burdened with such planning decisions in order to keep the response times low, but a separate local planner mediating between the global planner and local planner may be created as a solution.

In architectures where the global plan is to be interpreted by lower level planners in a flexible way, the output of the global planner serves as a mere orientation help for lower level planners. As an example, in a room with several doors, the global planner may have a waypoint indicating which door to traverse next, for the robot to reach the final goal. The actual path the robot will eventually take to that door can then be decided reactively by lower level planners reacting to circumstances like the presence of moving humans.

As a conclusion from these observations on splitting navigation planning into global and reactive processes, figure 3.3 shows the sketch of an improved navigation framework robots for moving among humans. They key here is that most additionally required functionalities for social behavior are neither located in the global planner nor in the local planner, but in intermediate layers. This allows to have adapted replanning strategies, that allow lengthy replanning cycles for high level of abstractions while still allowing the robot to exhibit reactive social behavior with respect to its local surroundings.

Responsibility	Planning layer
Choose waypoints for efficient traversal of buildings	global waypoint planner
Select current behavior	intermediate planners
Choose virtual target to move in formation with human or static position for interaction	
Choose path to next waypoint according to social constraints	
Choose micro-movements for non-verbal communication	
Choose motion command on path while avoiding (moving) obstacles reactively	local planner (obstacle avoidance)

Figure 3.3: Sketch of human-aware framework split of responsibilities for navigation

In the recommended framework, the task of the global planner is to provide waypoints such that at least one feasible path between two waypoints is expected to exist given the static obstacles (walls) of the environment and known blocking dynamic obstacles, and also that a robot path through these waypoints is expected to be close to the optimal path with respect to costs. This may be called strategical planning. The decisions of this planner should in most cases be robust against many unexpected events, such as motions by humans, so a change of the plan on this level of abstraction should rarely happen. Information about present humans would be used in a very coarse way, such as whether the density of a human crowd is expected to block the traversal of the robot (and any person it guides), or whether some social event takes place that the robot should avoid to prevent disturbance.

This allows the robot to exhibit an overall stable strategy to human observers. The global waypoint planner is only needed for robot missions where the robot has to move to a defined goals. For other tasks such as following a human or exploration, the planner may not be used.

Figure 3.3 next shows a lower level of abstraction for selecting a behavior. This is required for different phases of a single navigation action, and to adapt to special circumstances. As an example when navigating in order to meet a person, a robot can move in the same way as when moving to an arbitrary spot, but

once the target person is close, the robot behavior can be adapted to seek the attention of the person early and indicate its intent. When following a person, a robot may switch between behaviors for staying in formation and for searching a person lost by the sensors due to occlusions. When guiding a person a robot may switch between leading the way, waiting for the person, or going back to the person in case the person stopped. While the above examples come from papers on human-aware navigation [17, 49, 55, 113], there is overall no consensus on what state or what algorithm to use. It should also be noted that some notion of behavior variation can be applied at any layer of abstraction of navigation planning, depending on how much the factors driving a behavior change are predictable. If a robot opens a door and perceives a person on the other side wanting to cross as well, the behavior to move aside to let the person pass first instead of the robot crossing the door first can only be made at that time.

Next as a lower level of abstraction there is an intermediate path and pose planner. The path planner serves to find a more detailed path to the next waypoint, or to through the next n waypoints. Its main purpose is to reason about the spatial relationship between the robot and humans in the immediate vicinity, and produce decisions like on what side to pass a given human, not to unnecessarily cross through a group of humans, whether to overtake a given human, and so on. The pose planner serves for guiding and following missions and decides on a virtual goal the robot should move to in order to build a suitable formation with the person or persons it guides of follows, also taking into account obstacles and other humans in the environment. Both path and pose planning are subject to replanning whenever people in the vicinity behave unexpectedly. This level of abstraction may also be called tactical planning. Splitting tactical planning from strategical planning has several benefits: the long-term goal of the robot does not easily oscillate, and the strategic planner can scale with any size of the environment without affecting the response times of the tactical planners.

At an ever lower level of abstraction, robot motions can be planned in the terms of non-verbal communication, such as prompting gesture, gestures of respect, indicating attention or acknowledgment of a perceived human intention. The robot would perform such motions in a rather small area around itself not leaving the path planned by the path planner.

Finally the lowest layer of abstraction is that of collision avoidance, taking into account moving humans. While higher levels of abstraction also take into account moving humans (such as when reasoning about whether to overtake or not), the lowest layer of abstraction has to perform temporal reasoning, and in extreme cases may have to make extreme motions for the sake of safety. As such it may overrule all plans from higher levels of abstraction, but in the general case it is expected to merely execute plans.

3.1.1 Global waypoint planner

The global waypoint planner in the recommended architecture plans routes at large scales, on the overall map of the environment. This scale is intended for robots with an area of operation that is large but still bounded, such as a large building or a campus. This level of planning takes into account known static obstacles as well as knowledge about the density and flow direction of humans or other robots in areas, to have an estimate of travel duration. This may include special means of transport like elevators. No location of presence of individual humans is required as an input. Strategic planning may of course also involve other factors such as risk, and balancing traffic in a fleet of robots.

In very small environments, like households, a global waypoint planner has no benefit over a human-aware path planner. The global waypoint planner is used for commands with a defined goal end state, like going to a position. For open tasks like following, the planner is likely without much use.

Most research work on global planners for HRI focuses on path planning at which we will look next, and publications do not suggest using a planner at a higher level of abstraction than the human-aware path planner. However, path planning does commonly not scale well with the size of the map or the number of humans, meaning the reactivity of replanning suffers.

Algorithms for constructing waypoints can be found outside HRI, such as Voronoi diagrams. An alternative method is given by Thomson et al. [140], using random paths to detect interesting intersections, but not using social data in the construction.

Example research work on that kind of global planner with social information is given by Tipaldi et al. [141], using static knowledge about daily routines of humans to prefer passage through rooms of an apartment which are likely not occupied. Similarly the work of Chung et al. [22] on spatial effects may be used to collect knowledge about areas of the map where passing in certain ways may be inappropriate. This kind of static information can well be used by a global waypoint planner.

Inputs: Map of obstacles and socially relevant areas, task context (e.g. going alone or guiding)
Outputs: A list of waypoints, possibly with estimated time of arrival (ETA)

3.1.2 Behavior Selection

Behavior selection is usually not planning into the future, but merely adapting to the current situation, unless included in waypoint or path planning to e.g. calculate the quickest route taking into account behavior restrictions on the allowed maximum velocity in certain areas. Another example to consider here is moving within different densities of crowds. In non-crowded areas, a robot can mostly plan paths "around" humans taking advantage of free spaces. In lightly crowded environments of moving humans, a robot may rather plan paths that follow paths of other humans, in emerging lanes such as observed by Helbing et al. [60].

In denser or more dynamic crowds, local path planning may still work, but longer term path planning will probably not be beneficial, apart from the difficulties of observing humans in spite of occlusions. As an example Henry et al.[62] use path planning for such scenarios, but observe that only a short part of the full plan is acted out, before a different full plan is produced. In even denser crowds, where persons wanting to move have to stand nevertheless waiting for space to move, path planning is probably useless for a robot, and mostly reactive behavior is required, based on coarse waypoints.

Common solutions to behavior selection are finite state machines and subsumption architectures.

Inputs: Map of obstacles and socially relevant areas, task context (e.g. going alone or in formation), all perceived persons and their predicted motion
Outputs: The id of one behavior out of a set of candidates as part of the task context

3.1.3 Pose Planning

Planning for a target position has two major applications in HRI. One is selecting a spot for an explicit interaction, like handing over an object. The other one is to plan a virtual moving target while moving in formation with a person. In both cases, the planning needs to take into account all present persons, and in both cases, movements by the person involved can quickly make any previously selected spot unsuitable for the task. For moving in formation, the default case is that the virtual target is always shifting. Also for that scenario, the virtual pose is usually very close to the current position so that no further path planning is required.

As an example in the work by Pandey et al. [113], the potential poses for a guide robot are fixed with respect to the human position, and path planning is only used when the robot is too far away from the person, in which case a path is computed to such a pose. More references for pose selection can be found in the works of several authors mentioned in Section 2.4, i.e. [63, 67, 146, 45, 62, 118].

Inputs: Map of obstacles and socially relevant areas, task context (e.g. going alone or in formation), all perceived persons and their predicted motion
Outputs: A pose that the robot should move to next

3.1.4 Path Planning

The role of path planning is to find a detailed path that satisfies many quality criteria such as not discomforting present humans. The bulk of publications on navigation planning in HRI concerned with proxemics cover this area of path planning. The general task is to balance several quality criteria when moving through a given space. Existing research has been summarized in section 2.4.2.

Path planning layer is subject to frequent replannings. Usually a new planning phase is required whenever any human in the environment moves, because human motions cannot be well-predicted and thus usually affect the quality of the current plan.

Inputs: Map of obstacles and socially relevant areas, task context (e.g. going alone or in formation), all perceived persons and their predicted motion
Outputs: A detailed path to follow closely

3.1.5 Body Gesture Planning

Gesture planning refers to non-verbal language cues and signals. Whether intentional or not, while moving a robot constantly exhibits behavior that an observer may interpret to estimate the internal state of the robot, such as the robot's beliefs and intentions. Autonomous navigation primarily deals with moving the robot to a target location. But in the context of HRI, the relationship of the robot to present humans is relevant, and thus reasoning about the communicative aspect of navigational actions can improve the quality of the robot behavior. Additionally, a robot may deliberately use base motions to make or support explicit communication acts.

A general insight into signals and cues is given by Hegel et al. [59]. According to them, a signal is an act made for the purpose of altering the behavior of another agent, whereas a cue is any observable aspect of a robot being used to estimate by an observer to infer some information about the robot. So the robot has full control of the signal it sends, but what aspects are being used as cues depends on the given observers. Any signal becomes also a cue once an observer perceives the signal and interprets it, but not all cues are signals, and a signal toward one person can be a cue to a different person.

In HRI, corresponding research would attempt to find out what cues humans commonly use to determine information about a robot, and as a possible second step how to design a robot to improve any interaction. A very simple cue is stillness: When a robot does not move, an observer may take that as a cue that

the robot is off, or without active perception and manipulation processes, which may already be misleading.

While most work on signals and cues in HRI deals with robot arm motion ti support spoken dialogue, several publications deal with signals and cues in navigation. As an example Peters et al. [114] considered prompting signals at small passages where a robot would move aside to invite a human to pass first. Pandey et al. [113] considered taking such a path that the guided person is influenced to move towards the goal, which satisfies the definition of a signal. An example for research on cues rather than signals is given in the work of Hayashi et al. [55]. They considered two different modes of patrolling a shopping mall: friendly-patrolling and busy-patrolling. Friendly patrolling means a guard checks the environment for safety, but is also available to provide guidance to passing people. Busy patrolling means a guard is focused on safety checks, not to be distracted by passing people. They compared the behavior of human guards in 4 aspects when given as task one of the two modes of patrolling: speed, gaze and trajectories. The difference in these aspects would then not qualify as signal, as guards would not alter gaze and speed to alter the behavior of passing people, but to passing people, those differences would serve as a cue to the availability of the guard for questions.

While all robot motion planning in HRI is somehow involved in the problem of cues, gesture planning is a planning activity directed at signals and at cues that can be created by very small body movements. Gesture planning refers to both creating special small motions as signals, as well as suppressing small motions that could wrongly be interpreted as cues for something. Modulation of the robot gaze and head motions can be used to create signals and motions serving as cues. As an example, the robot head in many navigation frameworks is not taken into account and rests at a fixed position during base motion. This is in contrast to human motion where the head and gaze motions make anticipatory motions before the body turns [65].

The fact of the robot not pointing its head and gaze towards an oncoming person may be taken as a cue by that person that the robot is not aware of the person, even if by means of laser sensors, the robot may be well aware of the person. The cue given by the robot head position would thus be misleading. Similarly, a head

and gaze accidentally pointed towards a person for a longer time may be interpreted as an "aggressive stare", or attention seeking. An example for studies of gaze direction is the work by Takayama et al. [139], finding among other things that a robot should avert gaze when approaching persons closely.

With respect to body motion, it is possible to imagine a path crossing situation where the robot internally intends to give way to a person and thus plans to stop at a certain spot. Failing to visibly reduce the speed early on, the person may take the velocity of the robot as a cue that the robot will not break, leading to confusion. Gesture planning could attempt to visibly reduce velocity as a signal, which could be interpreted by the person as a cue that the robot is indeed going to stop, allowing the person to move on with more confidence.

Inputs: Map of obstacles and socially relevant areas, task context (e.g. going alone or in formation), all perceived persons and their predicted motion, currently planned path or pose if any
Outputs: (optional) gaze directions and small trajectories to execute

3.1.6 Obstacle Avoidance

Reactive planning deals with safety and obstacle avoidance, as well as reacting to human input to the handle sensors. The scope of this planning activity is driven by safety. Reactive planning has to ensure the next speed commands minimize the risk or damage of collisions with obstacles, humans or other robots. In the recommended navigation stack, the obstacle avoidance module gets as possible input a nearby pose to reach, a path to follow, or small motions to execute. The planner has to take into account the current kinodynamic constraints of the robot and select a motion command that adheres best to the plan input without being at a risk of a future collision.

Inputs: Immediate motion target (pose, path or trajectory), obstacle sensor readings, perceived moving obstacles with position and motion data
Outputs: motion command

3.1.7 Process Integration

The different layers of planning can be integrated in multiple ways. The most common way seen in literature is a top-down control flow where the highest level planner is invoked by a process exterior to the navigation stack, and then each planner in the hierarchy is invoked in turn. The main advantage of this strategy is simplicity of design. Since in navigation, all planners deal with base robot motion in 2D however, it is possible to use different strategies, such as invoking the lowest-layer planner first, and invoking any higher-level planner only if a given planner cannot find a valid plan. As an example, to move to a visible spot a meter away, a global waypoint planner or path planner may not be required, a typical local planner can handle this challenge without a global plan.

Therefore, this thesis does not present a recommendation on the integration of different planning layers, as there are any possible different designs, and the literature offers no qualitative comparison of ideas.

However, as a word of caution, it should be noted that the purposes of research and of production are quite different. For the purpose of research in HRI, it is common to focus on just a few aspects of human-awareness, and thus to have a simplified navigation stack. This may still serve the specific research focus, while being simple enough to be reproduced on other robots. So for research platform even in HRI, as shown in the examples earlier, it is most common to have merely two separate processes, a globally planning deliberate one and a locally planning reactive one. This simplification may sacrifice scalability or safety but not to a degree relevant to research. Similarly, the navigation frameworks used in the experimental parts of this thesis derive from 2-layered navigation layouts.

The purpose of actually deploying a robot in the real world is very different than the purpose of researching individual aspects of social navigation. This section has given an overview of the concepts to consider for producing human-aware navigation frameworks. This was given to explain how that architecture used in the following is useful for specific research aspects, but not intended for deployment in the real world.

3.2 Human-aware Static Cost Models

The previous section presented a generalized sketch of a human-aware navigation framework. For the rest of this thesis, a simplified navigation framework has been used based on the global path planner HANP described in section 2.5 and a local planner following the path. More details on the local planner will be given later.

HANP produces paths based on a static model of the environment, meaning temporal planning is intentionally avoided to prevent the combinatorial explosion of the search space and the consequences to robot reactivity. In order to deal with the unpredictable dynamics of human motion, HANP merely offers replanning as a strategy.

In the following it is shown that mere replanning is insufficient as a strategy for planning with static environment models, and offers adaptations to the navigation framework to improve the robot behavior.

The adaptation to the framework is twofold, the cost functions that determine the planned path are modified to take into account humans as moving or movable obstacles, and the local planner is changed to follow the global path while adapting the velocity taking into account humans in the vicinity of the path.

In general, when representing unpredictably dynamic situations in a static model for planning, information about motion has to be encoded. An environment model for planning represents the floor, walls and objects as perceived by the robot, it is essentially a 3-dimensional model of space.

In dynamic situations, the past and future dynamics which are 4 dimensional (given the added dimension of time) are projected into static (3 dimensional) space. All methods of projecting from higher dimensional space into lower dimensional space cause loss of information, and this loss of information necessarily means that a planner will not be able to find solutions to certain problems in a static environment models, that a similar planner would find in a temporal model of the environment. The easiest example is backtracking: A planner using a static environment model will never be able to solve a spatial conflict by mov-

ing the robot back first allowing some moving obstacle to pass, and then move forward again, because in the static environment, an obstacle never moves.

There are different ways of projecting past and future dynamics into static space. Any given projection type will allow a planner to respond benignly to certain dynamic situations and less favorably to other situations. So for HRI, a question is which kind of projection from temporal space representation to static space representation allows the most desirable planned robot behavior in dynamic situations.

The projection used by HANP is to discard both past and future motions, and to only represent the current situation as if it were static. This section investigates changes to that projection in several ways, and provides experiment to convey what situations are improved by changing the cost functions to represent moving obstacles differently.

Autonomous robot navigation among humans has many dimensions of quality that can be increased to improve overall perceived quality. In the recent survey [86] three main categories for research on human friendly navigation were given: comfort, naturalness and conformance to high-level social rules. The strategy of projecting dynamics into a static world model affects the robot behavior in dynamic situations, and perhaps the best concept to describe the issues that can be caused is the term “legibility”. This term is described first in the following.

3.2.1 Legibility

Roughly speaking, motion legibility means motion that is intuitively understood by human observers. To establish a better definition, the concept of signals and cues are required. As introduced in section 3.1.5, this thesis refers to signals as acts an agent makes for the purpose of influencing the behavior of another agent, whereas cues are behavior patterns that an observing agent uses to estimate the state of an agent it observes. So signals are bound to a communicating sender, whereas cues are selected by an observer of any system. Legibility is closely related to predictability, but in this context it is a distinct concept, that can sometimes even be opposed to predictability. As an easy reminder of this

notion of legibility, in literature legibility relates to how easy it is to understand a given piece of text, whereas predictability could refer to how easy it is to predict the next word or sentence. As an example in poetry, the use of rhyme and verse can make the ending of a line of a poem very predictable. Legibility refers only to understanding of the robot behavior up to the present, including the robot's intentions. A distinction between legibility and predictability is also made by Dragan et al. [28], who express that legible behavior is designed with the aim of being understood, while predictable behavior merely attempts to complete an action using known patterns. However their formalism reduces legibility to inferring just one internal state of the robot, the intended motion goal. This is useful for the definition of legibility in the narrow context of object grasping. In general robotics however, a broader definition of legibility is required that relates to observers inferring many other internal robot states than just the intended goal.

Knowing the robot intention only partially helps to predict the robot future behavior, as there are always degrees of freedom to achieve any intention, and even small differences in the robot intention can make big behavioral differences. As an additional difference, for predictability plenty of context outside the robot behavior may be used as cue to predict the robot behavior, not the least any explicit contract about what the robot was programmed to achieve, so a robot can sometimes be highly predictive to an observer even if the robot behavior is not very legible. It is difficult to quantify how often and to what degree legibility and predictability are positively or negatively correlated in general. For theoretical purposes, it is merely important that one property cannot be reduced to the other in general.

Legibility of robot behavior attempts to describe the degree to which the behavior of a robot allows arbitrary observers to estimate aspects of internal state of the robot. The success of any observer in estimating the state depends obviously a lot on the observer, his a priori knowledge about the given robot or similar robots, his experience with that robot, etc.

In navigation, general aspects of the internal state of the robot that may be relevant to present observers are as examples:

- the robot's state of being switched on and active
- the absence of software or hardware error states (robot "health")
- the navigational intention / mission of the robot (e.g. guiding, following, approaching)
- the robot's awareness of its own location
- the robot's awareness of obstacles
- the robot's awareness of the presence of humans
- the robot's awareness of the state and intentions of perceived humans
- the robot's awareness of recent relevant changes in its environment
- the robot's currently targeted waypoint (if any, e.g. the goal or the next doorway to move through)
- the preferred and intended path of the robot (if any)
- the preferred and intended velocity of the robot (if any)

In specific situations, there are more detailed internal states of the robot that may be of interest, e.g. when there is a queue of humans, an aspect may be for a robot standing somewhere near the queue whether the robot intends to be in the queue and considers itself part of the queue, or not.

The key to remember here is that regardless of whether the robot has any matching internal state, and regardless of whether the robot reasons about legibility, an observer will use robot behavior aspects as cues to estimate robot internal state, and react accordingly. As an example, a robot while moving forward may be well aware of a danger ahead, but if its behavior does not offer a cue that the robot is aware of that danger, an observer may feel obliged to intervene for the sake of safety. If the robot has a visible head with eye-like sensors, and moves forward while the head points to the side, an observer may take this as a cue that the robot does not see where it is going, regardless of what other sensors the robot may have.

While any of those aspects might be of interest to observers, a given observer may also be busy with something else and not interested in any of those aspects of state, so it is not viable for the robot to use explicit signals all the time that fetch the attention of observers to declare state. Instead, a beneficial feature of robot behavior is for these internal states to be generally easy to estimate from non-signaling behavior. So as an example in path planning, a path may not only

balance shortness and social distancing, but also optimize how the path may convey any of the aspects listed above.

Most of the aspects of internal state to reveal in navigation can best be revealed by lower level planners in a navigation planner hierarchy, meaning reactive local planners controlling immediate velocity changes of the robot. This thesis is mostly concerned about aspects of legibility that have to be considered in path planning.

Some factors that positively affect general robot behavior legibility are:

Distinctiveness of behavior aspects over different internal states
Consistency of behavior in similar internal states but different external situations
Naturalness of behavior modifications according to internal state

Distinctiveness means that when a robot had two different internal states in two similar situations, the behavior of the robot is noticeably different. If there was no noticeable difference in behavior, the robot would not offer cue to which of the two internal states it is in. As an example, imagine a robot with a coffee mug moving towards a person in two situations, in one situation to hand over the mug, in another to put the mug down on a desk behind the person. If the robot behavior is very similar in both situations, an observer would not be able to estimate the intention of the robot.

Consistency of behavior relates to longer-term human-robot relationships. Once a person has learned to rely on certain cues to correctly estimate the intention of the robot, then it helps this observer if the robot continues to exhibit these cues for other situations with similar intentions. As an example a robot handing over a coffee mug to a person sitting at a desk, and the robot handing over a baseball bat to a person standing in a doorway, some aspect of behavior in both situations should be similar enough to be usable as a cue to the robot intention.

For both consistency and distinctiveness, it may be sufficient for the robot to be consistent or distinct in a subset of all behavior aspects. So for distinctiveness, the robot may vary the gaze direction sufficiently in the examples above, but use the same approach velocities. For consistency, the robot may use the same

gaze behavior during an approaching phase, but may have different velocities adapted to the situation.

Naturalness for legibility describes how well-suited a given signal behavior or cue is to convey a given internal state. The most convenient way of achieving natural signals and cues is to mimic the behavior of agents in nature (or otherwise well-known artificial agents). Naturalness often derives from function, meaning when an agent acts in a goal-directed way, then his actions help to approach the intended goal, and observers can take any changing state as a cue to the goal of the acting agent. In simple navigation cases, it is natural to choose the most efficient trajectory towards the goal, which is usually the shortest path. Where possible, the shortest path is the direct line towards the goal. So where a person is observed to be moving on a straight line, the observer can take this as a reliable cue that the person's intended goal (or significant waypoint) is somewhere in the extension of that line. So in simple enough cases, a robot acting goal-directed already provides good cues to its internal state. However in unpredictably dynamic states, this is not as evident.

Measuring legibility can be challenging because different observers will use different cues, and have different models of the possible robot internal states. An ad hoc approach to measure robot legibility would be to expose observers to robot behavior and then attempt to determine how well these observers can estimate the internal robot state. Given the variability of observers this approach has a large error margin. An alternative approach is to inform observers to some degree about the robot internal state prior to the observation of the robot. The observers can then after their observation express to what degree the robot behavior fit their expectation given the assumption about the robot internal state. The design challenge is then to prevent the robot from producing behavior that does not match the expectations of observers. In navigation, this could be to tell observers where the robot is supposed to move to, but not telling them how the robot is going to move there, and then letting the observers rate how well the robot behavior matches its assumed goal.

While this may be a weaker test of legibility in that it does not allow testing how easy the robot internal state (e.g. intention) can be guesses without a priori knowledge, the advantage is that this test can be quantified in experiments.

In current autonomous robot navigation, robot behavior can usually be described in terms of the motions of the robot base, in some cases also of the robot "gaze", meaning the motions of camera units or humanoid heads. For mere base motion the behavior aspects than can vary and thus serve as cues are the direction the robot body is facing (if the robot has a discernible "front" direction), the robot rotation, the direction of current motion, the path the robot has chosen over the last moments in the past, and any other recognizable motion patters such as oscillations, strong velocity changes, and the relation between the robot base and environment objects (such as mimicking another object), or standing still facing a door closed door (possibly as a cue that the robot wants to cross the door but cannot open it).

Using only the robot base motion, it is difficult to provide cues for all aspects of internal state of the robot. A robot having an extremity (e.g. a head) that can serve to provide a "gaze" direction independently of the base motion direction may use this to convey more aspects of internal state than if just the base motion is used. As an example consider the internal states of

- the intended short-term path of the robot
- the next waypoint along the path the robot wants to reach
- the robot's awareness of a human standing nearby along the path

This example gives three directions from the robot that concern internal states of the robot, which an observer could be trying to guess from behavior cues of the robot.

By only moving the robot base (and not controlling the gaze), the robot can point in one direction as his motion vector, and possibly a second direction as the extension of his front side if the base allows sideways motions. But there is no degree of freedom to "point" at the third direction of the example. Using the head, a robot can direct its gaze to offer more cues to its internal state.

However in this thesis we only consider robot cues as offered by the robot base motion (not taking into account head motion), and also consider the restricted case of a robot design where the torso is linked in a fixed way with a differential-drive wheel setup, such that sideways motion is not possible. In such a setup, the robot path planner decides the base motion direction and also the direction

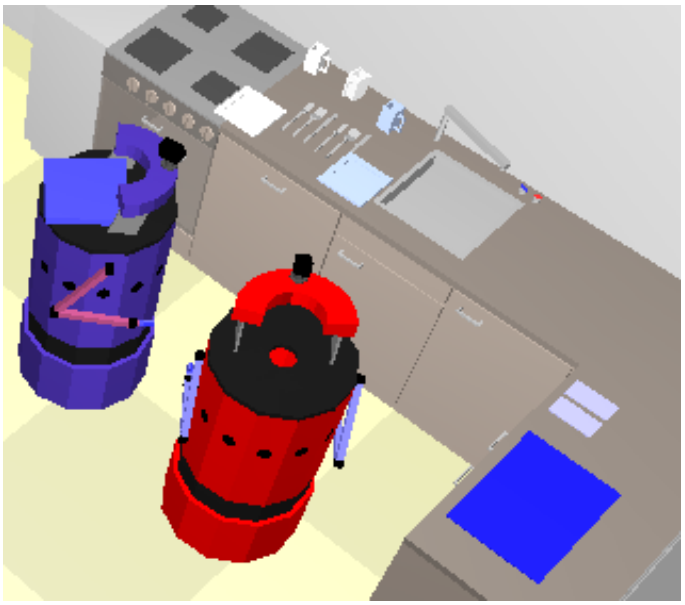


Figure 3.4: Example spatial conflict for two agents in the Gazebo simulator trying to reach a position to grasp items

of the front side of the robot, which evidently strongly influences the legibility of the robot.

3.2.2 Moving in confined spaces

The work in this section has partly been published in [84, 85].

A typical human household such as considered in the introduction section 1.1 is not generally spacious. Households balance the need for space with the need for furniture and a desire to keep the required size small. Such households have many confined spaces not considered in most works on human-aware navigation, where wide areas like shopping malls with plenty of space are the focus or research.

In confined spaces, a cost function that represents humans as fixed obstacles produces unnatural navigation paths. Unnatural means that humans in the same situation would choose different paths.

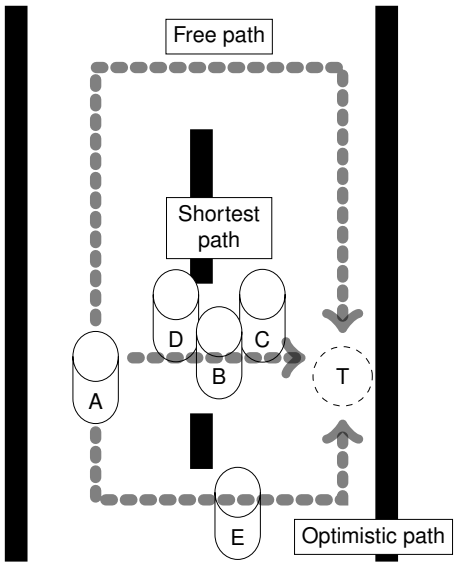


Figure 3.5: Strategy alternatives: Collision free path, shortest path ignoring humans, optimistic path balancing length vs. human comfort. This shows optimistic planning being different from path planning where humans are simply ignored.

In the situation shown in Figure 3.5, the shortest path seems blocked by persons. The alternative path using the free space is extremely long. In a model of the environment modeling people as fixed obstacles, a planner cannot find a natural solution such as approaching the seemingly blocked path giving the people in the way the chance to make room for the robot, to avoid the robot service to take unnecessarily long.

Similarly, considering humans as fixed obstacles may cause a planner not to find any solution. The two robots in Figure 3.4 need to move to the same position for performing a task. If any of the two occupies the end position the navigation planner does not find a solution for the other. However a human solution would be to move to a spot visible to the other and ready to move in when the other moves. A path solution to the goal would then intersect with the human in the static model of the environment.

A motion controller of a robot can ensure safe motion while allowing a global planner to return such paths to positions occupied by a human. The robot in the experiments use a local-planning algorithm that linearly predicts the motion of human based on the perceived current velocity and speed. The local planner

thus not only avoids entering occupied space, but also space that is likely to be occupied in the immediate future.

In the experiments of this section no explicit communication between the agents has been used. For navigation in indoor contexts, humans rarely need to communicate verbally, so this restriction should not limit the value of the research.

The contributions presented in this section are

- an extension of the human-aware navigation planner HANP to make plans more efficient in the presence of moving humans in small areas;
- means for plan execution, which ensure safety and take into account possible future movement of other agents;
- an experimental evaluation of the extended planning and execution mechanisms including a discussion on their usefulness in different situations.

The next subsection provides and overview of related work. The following one details on the human-aware navigation planner. Then my own contributions to the planner are detailed. The following evaluation compares different combinations of these strategies in different environments and discusses their applicability. The paper ends with a conclusion.

3.2.3 HANP limitations

The possible output cases of HANP can be classified as follows:

1. A path which is optimal even if humans moved out-of-the-way
2. No path, though a path is possible should humans move out-of-the-way
3. A path which is much worse than a path possible should humans move
4. No path, and no path is possible even if humans moved

The first case and fourth case cannot be improved. We focus on the second and third case, where the dynamics allow solutions which a static planner does not consider.

(a) Endpoint blocked

(b) Corridor middle conflict

Figure 3.6: Cases where static planning fails due to agent B blocking collision free solutions for Agent A to goal.

Figure 3.6 illustrates examples for the second class of results: the goal position is currently blocked by another agent in a. And in b an agent is in the way in a room too narrow to plan a safe path around it. These are cases where humans wouldn't capitulate, but coordinate their actions with the other person, in most cases implicitly without the need to communicate verbally. Also often this situation does not even require a different plan than if there was no human obstacle, as by the time the robot reaches the blocked location, a human there could have moved. The capability and likelihood of the given human moving by the time the robot reaches the occupied space could in theory be estimated by higher-level knowledge about the given state and activity of the human. Considering this is outside of the scope of my research, in this context a cooperative human is assumed.

The third case — of finding a very inefficient plan — is illustrated in Figure 3.5. Although HANP would find a valid plan (the long way around the large obstacle) it is doubtful if such a plan can still be considered most legible or efficient. A human in the same situation would rather indicate its intended goal and hope for other humans to make way rather than moving the long way around. The Figure also shows that when there are several paths blocked by humans, some paths are preferable based on the number and activities of the humans.

The second and third case can be expected to happen frequently in realistic domestic environments. A robot which fails in those cases would disappoint. The following describes methods for generating plans in HANP, so that the robot shows adequate behavior in such situations.

3.2.4 Dynamic Planning and Plan Execution

As explained earlier, HANP projects potential dynamics of the environment into a static model by merely discounting any dynamics or potential dynamics. This projection can be changed to represent dynamics differently. By allowing path planning to consider positions in space that are occupied or probably will be occupied by humans, more natural and legible path solutions become available as solutions, with the drawback that sometimes the solution may not be realizable. Such a change to the cost models also requires an adapted navigation framework where the local planner takes into account that the path planned by a path planner may require predictive obstacle avoidance with social distancing, rather than mere path following strategies.

In contrast to other local planning strategies, the local planner used here does not consider moving off the planned path to avoid moving obstacles. Instead, it reduces velocity on the path in case of conflicts, and relies on the global planner to produce updated plans. Such a local planner is not suitable for robots outside the research context, where the ability to leave the global path for emergency maneuvers is crucial in extreme cases. However in the context of research on path planning, such a local planner is sufficient.

The local planning approach can maybe best be understood using the visualization of Figure 3.8. Agents (humans or robots) are indicated as cylinders. An agent shown with dashed lines indicates a previous position of this agent.

The starting positions s are indicated as a circle, target positions g as a circle with a dashed outline. A *waypoint path* P (indicated as a dotted line in Figure 3.5) is defined as a sequence of points: $P(s, g) = (p_0 = s, \cdots, p_n = g)$ for the robot to traverse in sequence. Any kind of output of motion planners could be transformed into this format, in our case the planner already returns the path in this format. Other motion planner output, such as the robot joint configuration or preferred speed between points, is not considered here.

The localplanner has to consider the starting list of points on the waypoint path for a length sufficiently large to stop the robot when social distancing requires it. This is a requirement that is added to the safety requirement, so the time and

space horizon for a local planner may be larger than for mere safety, but on the other hand as we restrict the robot motion to stay on a global path, the computational effort is less than when trying to optimize robot velocity exploiting all available space.

For local planning, it is required to perform dynamic obstacle avoidance and social distancing tasks while moving forward on the planned path. To achieve this, the local planner investigates the near-future development along the path. Dynamic obstacle avoidance can be performed by state-of-the-art local planners, the challenge in HRI is to produce social robot behavior on top of dynamic obstacle avoidance. To achieve this, the local planner was restricted to follow the global path where safety allows it, and require the local planner to reason about social distances in cases of conflict.

Based on a motion plan P and the robot's predicted positions x_i over time on the path, an immediate occupancy path T_l of maximum length l is a set of points by adding the line segments until the sum of their length reaches l (Equation 3.1).

$$\begin{aligned} T_l(P, x_i) = \{t \mid & \exists i : p_i, p_{i+1} \in P \wedge t \in \overline{p_i, p_{i+1}} \\ & \wedge \|\overline{p_i t}\| + \sum_{k=x_i}^{i-1} \|\overline{p_k p_{k+1}}\| \leq l\} \end{aligned} \tag{3.1}$$

Given a required minimum social distance d_{Social} and a maximum traveled distance for the robot at its current velocity to reach a full stop d_{Stop}, the length of the subpath to consider should be at least $d_{Social} + d_{Stop}$. In confined environments, there is not much space for large social distances, so d_{Social} is considered to be a small fixed value.

The *area of conflict* A_c for a robot with radius r is then the area around its immediate occupancy path T_l with the with of the robot radius r, where other agents overlapping with the area would be too close (Equation 3.2).

$$A_c(T_l, r) = \{a \mid \exists t \in T : \|\overline{at}\| \leq r\} \tag{3.2}$$

The *geometry of an agent i* is defined by its 2D bounding circle D given its current position p and its radius r as: $D_i(p_i, r_i) = \{q \mid \|q - p\| < r\}$.

A *geometric conflict* for robot A_c and agent $D_2(p_2, r_2)$ (as in Figure 3.8) is then defined as a pair of robot position and other agent position that are too close to each other (Equation 3.3).

$$C(A_c, D2) = D2 \cap A_c \neq \emptyset \tag{3.3}$$

Since the modified path planner may return a path with potential collisions, a suitable local planner must check this area for potential conflicts before moving on. Agents further along the path need only be considered when the robot gets close, as the situations might have changed until then, with other agents having moved. The area of conflict may also serve to detect, avoid and resolve potential conflicts heuristically.

3.2.4.1 Optimistic Planning

As described in Section 2.5, HANP uses several cost functions for modeling social aspects of the navigation problem. In the standard costs of HANP, an area around the human is removed from search where the robot would collide with the human. This prevents the robot to consider paths that require the human to move. In order to have a useful cost function for dynamic situations and situations of potential dynamics, the cost function is changed to actually allow planning future motion through currently occupied regions. For humans or other agents in the environment the cost functions already present in HANP were used, but planning paths through the positions of agents was allowed(though at high social costs).

Using these cost functions ensures that the robot will use a path around the human if the detour is not too big. However, when an alternative path produces higher costs (e.g. by a longer path or the violation of other constraints like visibility), the modified HANP produces a navigation path intersecting with the

human position. When the robot executes this path, it will move towards the human and the localplanner will stop the robot before colliding with the human. Only when the human moves out-of-the-way, the robot can follow this path and achieve its goal.

The modified path planner can yield one of the following results:

1. A path is found which requires other agents to move.
2. A path is found though other agents lack free space to clear that path
3. No path is found and no path is possible even if other agents moved.

In the cases shown in Figure 3.5 and Figure 3.6, optimistic planning will return solutions, which require the other agent to move. It might be surprising that the second class of results may find a path which cannot work in any case. The cost function strategy that assume space that is currently occupied by a human may become available during execution of the plan has to fail where it is impossible for the given person to move away.

This should not be an issue if there is generally at least enough space to allow two agents to squeeze past each other in the working environment. This should generally be the case in household domains where robots will be introduced, because in cases of robot failure or shutdown, an emergency path past the robot needs to be available. Another alternative heuristic strategy that could be tested is to evaluate the ability of the human to move to a location out-of-the-way of the robot once the robot is near, using temporal planning for n agents. However such extreme cases are unlikely in the assistive context considered by this thesis.

3.2.4.2 Predictive Planning

When a human is moving, the cost function for humans can be adapted further to account for the direction of movement. In the static case, HANP assigns a cost function, which is higher directly behind the human than in front, which leads to the robot approaching humans from the front.

In a situation where the human moves, it is possible to reverse the costs as illustrated in Figure 3.7 so that the movement for the robot in front of the human

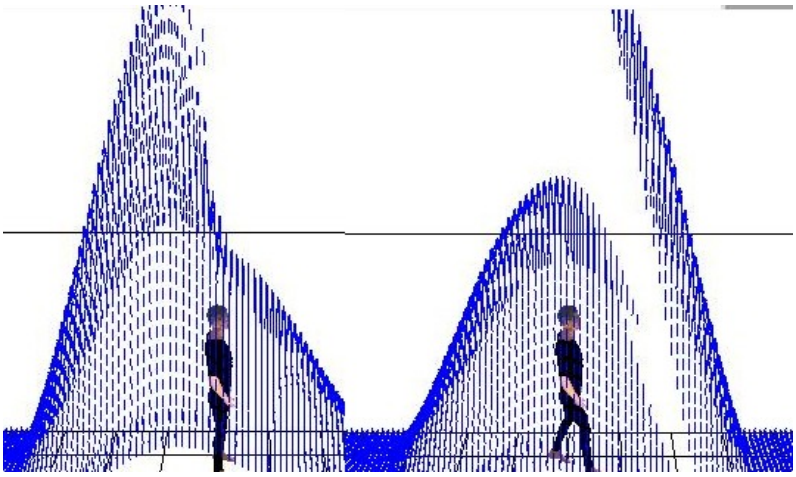


Figure 3.7: Costs in front of moving human increased in comparison to standing human. For a standing human, unnecessary motions in the back of the human are prevented by assigning high costs to that area. For moving humans, we rather avoid moving in their direction of motion.

is more costly than at its back or current position. Assigning costs in front of a moving human has the effect that a path planner will select paths that circumvent this area. In a first experiment, the benefits of this for situations when two agents approach each other frontally, and in confined situations is shown.

Later in this thesis, it is shown that for other spatial conflict situations with more available space, assigning path costs in front of other agents unconditionally can lead to confusing robot behavior. In many spatial conflicts, it is preferable to not assign costs and not deviate in the path, but instead to adapt the robot velocity while remaining on the shortest path. This keeps the directional intention clear, and does not make the situation more complicated for other people moving nearby.

3.2.4.3 Conflicts in confined spaces

The modifications in the use of HANP can lead to paths which can intersect with human positions. A robot moving on these paths has to adapt its velocity in time to avoid approaching persons too closely. For executing the plans, the execution has to take into account different conflict situations. Figure 3.8 shows

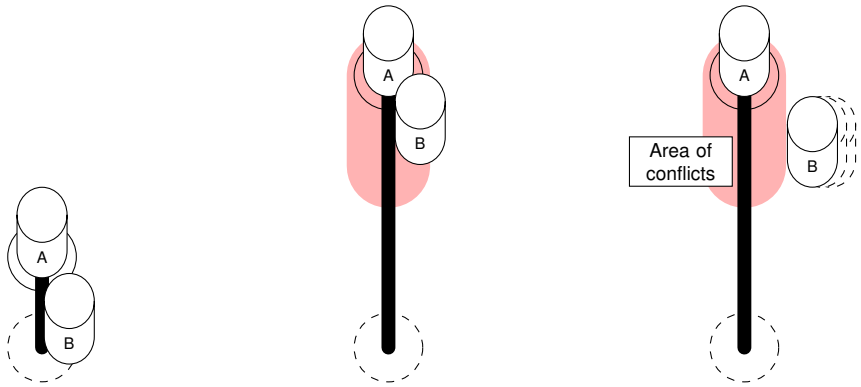


Figure 3.8: Possible conflict situations during motion. a Goal Conflict bPath conflict c Potential future conflict. The red region shows the area of conflict, the black line shows the planned path for robot A. Such conflicting paths may be the result of path planning if there are sufficient other obstacles or other agents in the environment (not shown here).

the cases used to classify such situations, using the definition of conflict given earlier in equation 3.3.

Figures 3.8a and 3.8b show the different cases of conflict which would cause the robot to be blocked in its approach to a goal, unless the other agent moves.

In the case of another agent blocking any part of the path, possible resolution strategies are to wait for the other agent to move, to initiate communication to demand the agent to move, or to fail the execution of this path and replan under different assumptions, such as choosing a different goal, or planning under the assumption that the given agent will not move aside.

The robot can also expect future conflicts by observing agents that are not in a state of conflict at present, but can be assumed to enter this state very soon (considering their current position and direction of movement) as shown in Figure 3.8c. This is minimally part of collision avoidance with dynamic obstacles, but in addition to collisions, the robot has to avoid confusing or threatening behaviors with respect to moving humans. So the controller following the path has to adapt the velocity to avoid approaching humans more than a social distance in the future. It can do so by predicting the future robot position along a small part of the global path, in cycles.

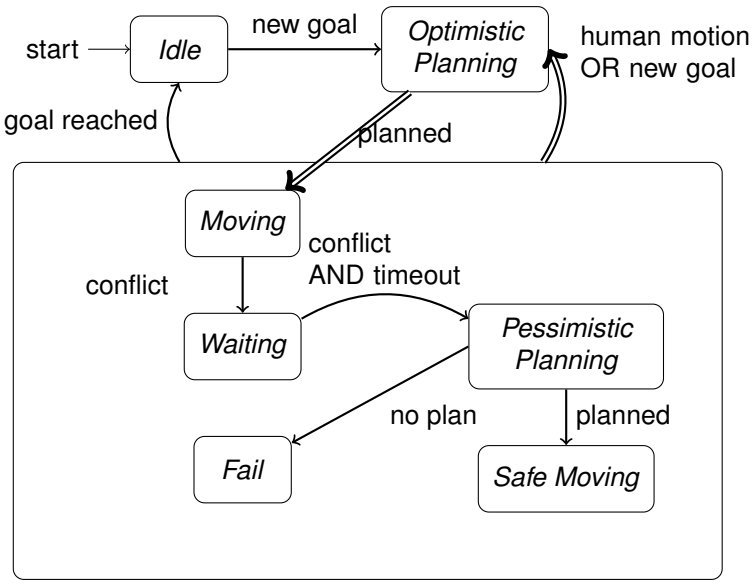


Figure 3.9: Flow chart for dynamic execution of human-aware motion plans in the presence of a single conflicting human. Plan-move cycle shown as double lines.

The length of the area of conflict needs to be minimally long enough for the robot to break in time for on that sub-path. In the following experiment an immediate occupancy path of minimum length 1.2 m is used to constrain the area of conflicts. The robot can come to a stop from its maximum velocity much earlier than reaching the end of that area.

3.2.4.4 Dynamic Plan Execution

When executing plans in dynamic situations with uncertainty about the behavior of other agents, it is necessary to replan and change behavior if assumptions or expectations are not fulfilled. Section 3.1 presented a sketch for a human-aware navigation stack, with several planning layers. In addition to geometric path and velocity planning, optimistic planning sometimes requires to coordinate robot behavior on the path to solve conflicts. This is one of the cases considered in Subsection 3.1.2, where a selection of robot behaviors needs to be made depending on the context.

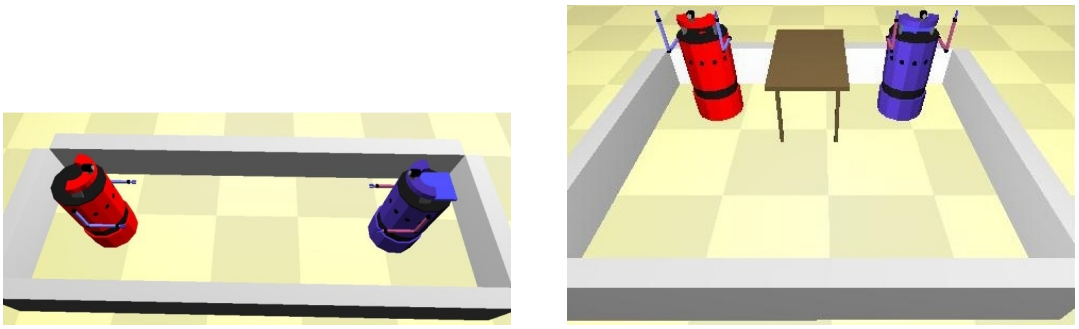
When executing navigation plans with possible conflicts, the robot needs to ensure it does not violate social distances. Instead, the robot should approach humans on the path with adequate speed, and convey its intent to move through the space occupied by the human. It is however possible that the given person does not wish to move aside for the robot. In that case, the robot needs to resolve the current navigation problem by searching for a solution that does not require that person to move. Without explicit communication, it is difficult to decide whether the given person is willing to move or not, so for simplicities sake, the ad hoc strategy used here is to wait a given time before giving up on the planned path, and the replanning under the assumption the location of the human is blocked long-term.

So when the robot detects a conflict, it stops and waits for some time. The presence and passiveness of the robot might be an implicit signal for the human to clear the path.

Figure 3.9 shows the conflict resolution approach as a flow chart of states simplified for navigation with a single human partner. The flow chart has one basic plan/execute loop that causes a new optimistic plan to be generated each time a human moves. This causes the component to constantly replan in dynamic environments. When a human is on the path of the robot, the robot will initially wait, relying on the planner to grant no alternative path is preferable to waiting for a certain time, indicating intent to pass. When a robot has waited for a person to make way and the person has indeed moved, a new plan will be calculated which hopefully allows the robot to pass.

The *Optimistic Planning* state in Figure 3.9 represents the modified version of HANP using optimistic and predictive planning, whereas the state *Safe Planning* is the planner planning path which never intersect with humans. Such a safe plan can be executed without the monitoring for conflicts by just moving along the generated path, but if a human moves, a new plan is still generated.

The navigation task can fail if the robot's goal position keeps being blocked by another agent or there is no alternative path if the human doesn't step aside. So far the controller only fails execution in this case, however with future work a



(a) Corridor, area size 5 m × 1.6 m (b) Crossing, area size 3.6 m × 3 m

Figure 3.10: Two experiments chosen for evaluation. In a the task was to swap positions, in b the task was to go to the opposite corner of the room as shown in Figure 3.14.

higher level controller could decide to start communication with the person or change its course of action completely.

The flow chart describes the behavior in a conflict case, not an actual algorithm. In particular when adding more humans, an algorithm would look simpler on the abstract level, as no distinction between optimistic and pessimistic planning could be made. Instead, for each perceived person, the robot would have to distinguish and maintain a flag of whether the robot will plan optimistically or pessimistically about the likelihood of that person moving, given that persons abilities, current activities, and prior experience with the person. Should the robot be able to infer that the person is bound to a location by an activity, additional reasoning may calculate the costs of waiting until that activity has ended. But those are aspects of human activity recognition that lie outside the scope of navigation planning.

3.2.5 Evaluation of strategies in confined spaces

This section analyzes the properties of robot behavior for resolving spatial conflicts in confined spaces with different strategies.

The algorithms are applied in a simulation with two robots, using a custom Player driver for the motion control, HANP for the path planning, and Gazebo

for the physical simulation of the 3D environment which models robot motion realistically. The robots are B21 platforms with differential wheel drives.

Two different situations were used in which both robots moved to individuals goals, with the shortest paths crossing each other. The examined environments are illustrated in Figure 3.10.

The compared strategies are

A: Optimistic planning
B: Optimistic planning with asymmetric waiting during potential conflicts
C: Optimistic planning with predictive planning

All strategies use optimistic planning, as non-optimistic planning always leads to deadlock situations in confined spaces. As an example, in the experiment in Figure 3.10a, both robots fail to find any solution using non-optimistic planning, and would never move. For strategy B, only one of the robots had the strategy of stopping and waiting while the other robot was in a *potential* conflict. This because preliminary test showed frequent occurrences of swaying and livelocks for strategy A.

In C predictive planning was assessed, where each robot will try to find a path avoiding the space in front of the other robot if the other is moving.

The corridor situation in Figure 3.10a represents all potential situations in the real world where a robot and a human need to pass each other in a space where one would block the other when standing in the middle between obstacles. In the situation in Figure 3.10b, the direct paths of the robots cross each other. This is like a crossroads scenario, only that in this form it is more likely to occur in a household. The measured variables were the time needed for both robots to reach their goal positions, the time the robots stood waiting, and the total distance traveled.

3.2.5.1 Results

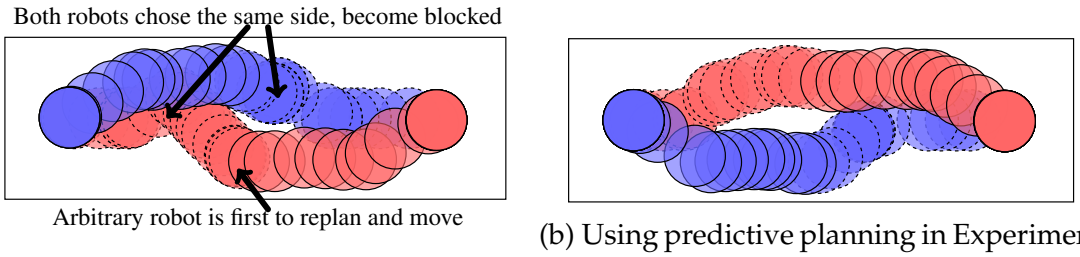
Figure 3.12 shows the quantitative results for the experiment in the straight corridor. The measures are visualized in the figures as boxplots, the box indicating the range where 50% of all values occurred, the line in the box being median, and the "whiskers" representing the minimum and maximum values.

The measured variables were the time until both robots reached their goal position ("sum time to goal"), the sum of the times the robot waited during their navigation task ("sum stopped time") and the distance both robots covered during one trial ("sum distance").

The results first of all show that optimistic planning is a suitable solution for confined spaces, returning path solutions where the path is blocked, yet becomes available after a change in the world by other agents moving.

The experiment often led to deadlocks as shown in Figure 3.12a. This happened when the robots met in the middle of the corridor due to swaying. The deadlock cases were not used for the analysis of times, but it is interesting that predictive planning reduced deadlocks to almost zero. This is a first indication of predictive planning reducing robot swaying. Deadlocks are not a problem when interacting with humans, but the robot swaying in corridors is a problem. Therefore a reduction in deadlocks for a robot-robot scenario indicates an expectable improvement for human-robot scenarios.

The differences in performance between experiments A to C can be explained due to swaying behavior which naturally occurs with non-predictive planners. Figure 3.11 shows samples of trajectories traversed during the experiments. In the depicted non-predictive case a, both robots at least once chose the same side for passing each other at some time. Additional swaying was most probably prevented by variances in planning duration of the individual robots. The sample for predictive planning shows robots having seemingly chosen different sides of the corridor, preventing a block. This is merely the effect of one robot choosing a random side of the corridor in optimistic search, moving in that direction, and by that motion making the predictive search of the other robot prefer the other side of the corridor. The swaying cause for performance differences is underlined by



(a) Using static planning in Experiment A

(b) Using predictive planning in Experiment C

Figure 3.11: Behavior in experiment A. The circles represent the bounding circles of two robots at different times. Earlier spots have dashed borders.

the amount of stop times being non-predictive for static planning. Strategy B did not reduce the swaying in the straight setting, as can already be seen from the performance.

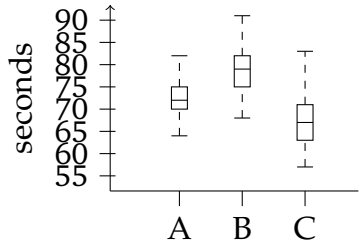
Figure 3.13 shows similar statistics for the curved room. While the performance of the strategies measured in time is similar as in the straight corridor example, it is noticeable that the distance traveled improved with Strategy B. This is due to one robot stopping and waiting as soon as the other approached its area of conflict, thus making it possible for the crossing situation to be resolved sequentially always in the same order of robots, without a lot of swaying. This distance benefit did not occur in the straight corridor environment. However, despite the reduced swaying, the time taken was still worse than when swaying was reduced using predictive planning.

3.2.6 Validation in Kitchen

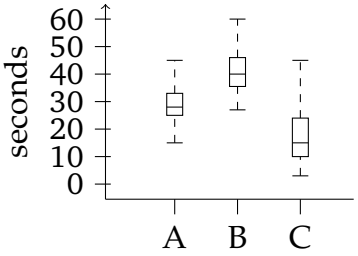
As a separate experiment, the optimistic and predictive planning algorithm were tested in a situation of manipulation, to observe validity and the likelihood of real-world benefits. This was done in a kitchen scenario in simulation, as can be seen in Figure 3.15. In the experiment, two independent autonomous robots, representing a robot R and a human H, execute two different independent manipulation plans at the same time. The robots had to bring items from a sink to a table, the plans were deliberately chosen so that robot R would in three occasions have to move to a spot robot H occupied temporarily. This was done in two sets of 30 trials. The robot H representing the human always planned

	Success	Deadlock
A	272	151
B	272	151
C	434	4

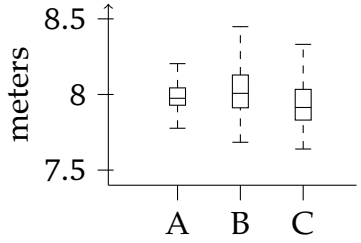
(a) Experiment successes



(b) Sum time to goal

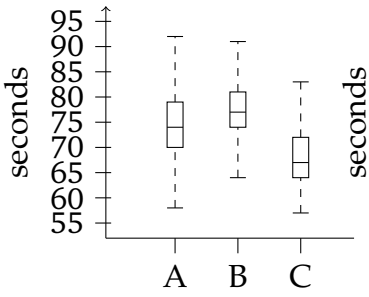


(c) Sum stopped time

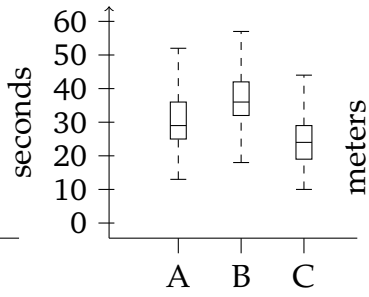


(d) Sum distance

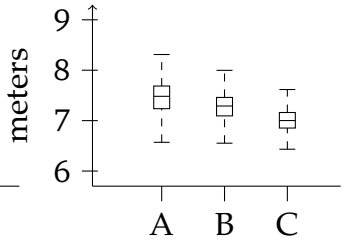
Figure 3.12: Results for linear corridor displayed as boxplots. A-C all used optimistic planning.
A: Non-predictive planning, B: Non-predictive with asymmetric wait, C: predictive planning.



(a) Time to goal

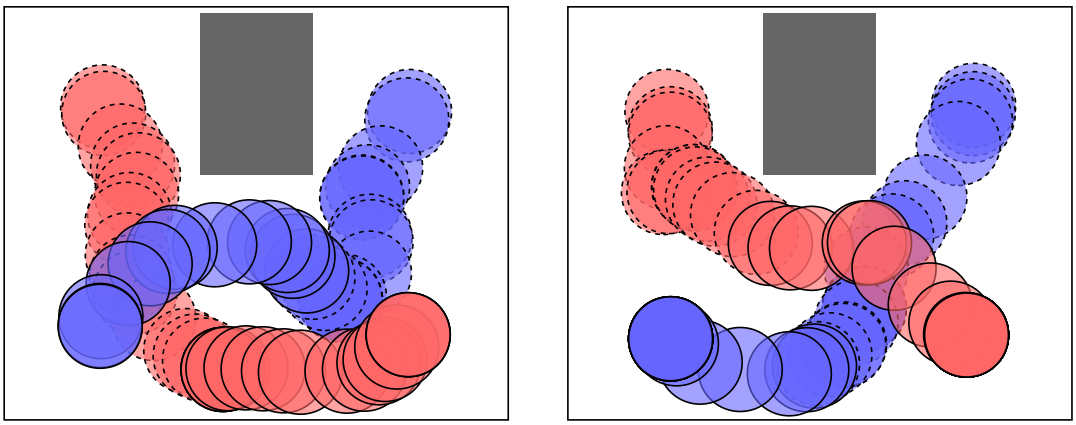


(b) Stopped time



(c) Distance

Figure 3.13: Results for room with table for 300 successful samples. Values are the sums for both robots in each trial.



(a) Non-predictive planner (b) Predictive planner

Figure 3.14: Behavior in curved room. With static planning, both robots first plan to take the outer "lane", making the blue robot sway.

the path optimistically and predictive, as in strategy C. Robot R had a motion planner that was neither optimistic nor predictive in one set of experiments, and an optimistic predictive planner in the other set.

The kitchen experiment on the one hand validated that for realistic scenarios, allowing optimistic planning does not deteriorate the robot behavior, and on the other hand showed increased legibility of robot behavior. As Figure 3.15 shows, optimistic planning made a visible difference to robot R. In the case of non-optimistic planning, the robot R merely waited at the position he was in when the goal region became blocked. From looking at the pictures the intention of R waiting in the first row of pictures cannot be guessed. Whereas using optimistic planning, the robot R in the three cases displayed waited closer to his destination, conveying legibly its intend to move there.

The evaluation shows that the proposed measures for human-aware navigation in dynamic, confined spaces lead to more efficient and more importantly to more legible plans than using the assumption of a static environment.

3.3 Reactive replanning in dynamic world with static models

The work in this section has partly been published in [81].

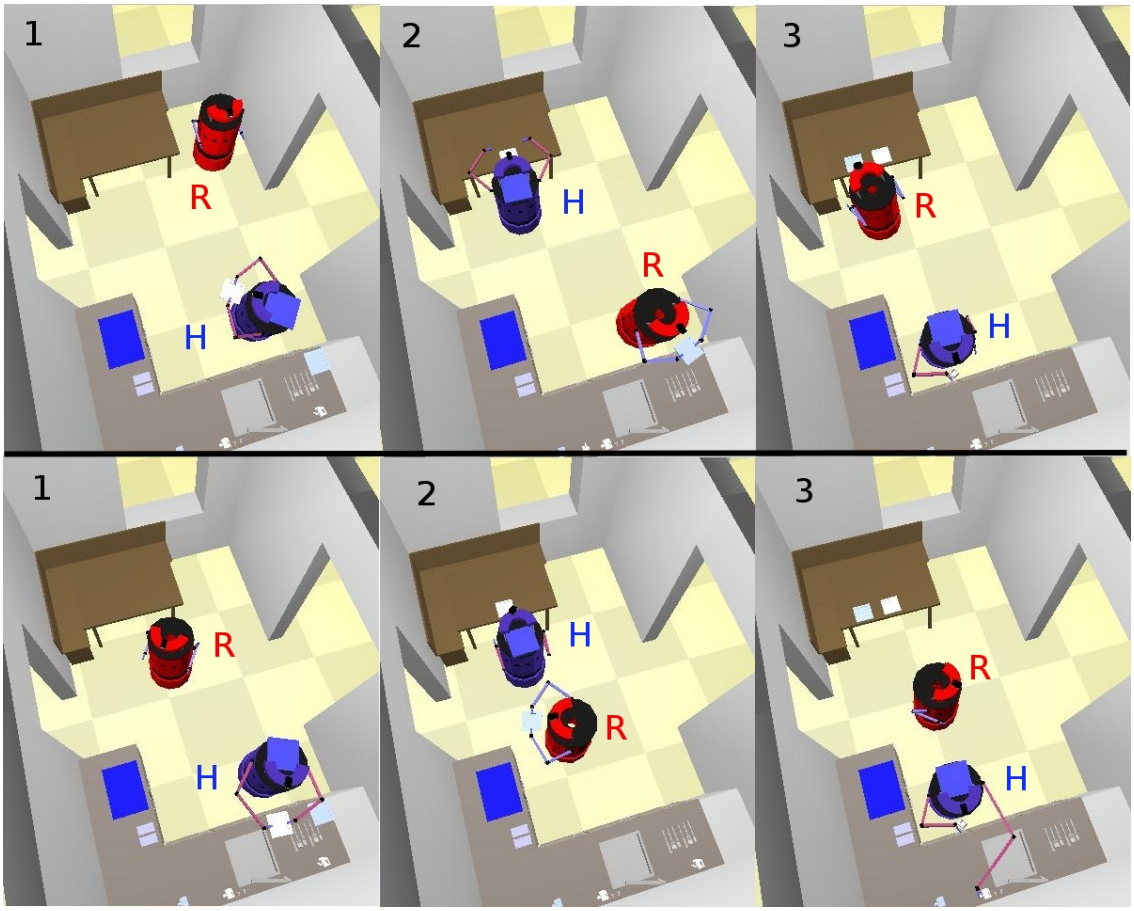


Figure 3.15: Snapshots from the kitchen experiment. Two individual trials, from left to right. The top one shows robot R with non-optimistic strategy, the bottom one with both robots using optimistic and predictive strategy. The images show instants where robot R needed to go to the spot of "human" H next.

Section 3.2 focuses on robot motion around humans in a home environment. It shows how using optimistic planning and predictive costs can improve the robot efficiency, and also an informal notion of legibility. While efficiency can e.g. be measured by comparing the time or energy required for a task, it is more difficult to formalize and measure soft qualities of behavior like legibility. One common attempt to estimate robot behavior quality from data is to compare robot behavior to human behavior. This section contributes with an analysis of behavior in specific situations in comparison to human behavior.

The focus remains on robot navigation in the proximity of moving humans ensuring maximum comfort and legibility in dynamic situations. Legibility means that a person is likely to estimate the intentions of a robot from observing the robot behavior. Legibility is understood to be improved if an observer achieves correct estimates more often, earlier, or in more details. Lack of legibility can exhibit itself both by wrong estimates of robot intention and general uncertainty about the robot intentions, or the robot having any stable intention to estimate. Legibility may also relate to other internal state of the robot than intentions, such as beliefs, but this section focuses on a simple navigation situation without giving a larger task context, so the main internal state a robot can be legible about is its intentions at various levels of abstraction.

Using a Human-Aware Navigation Planner (HANP)[135][1] allows a robot to plan paths that, for instance, avoid areas outside the human field of view and keep the robot from moving closely behind the back of the person. Cost functions applied to a grid can model acceptability of locations by predicting how much a human will feel distracted by the robot moving too close, in the back of the human or suddenly appearing behind obstacles.

In contrast, the criterion of legibility is hard to model as a cost function on space. Understanding the behavior of a robot depends largely on the situational context and the preferences of humans. There is no general model of agent behavior nor an accepted model of how to rate the legibility of observed behavior. However, studies that link humans' capacity to anthropomorphize with social robots suggest that human-like qualities in a robotic agent facilitate its role in

[1]Original name HAMP changed to HANP in [133]

human society [29]. This could mean legibility is improved by acting in a natural, human-like way.

The original cost model of HANP, called *Static* here, was designed for static environments. A common straightforward method to account for the dynamics in the environment is to recalculate the path each time the environment changes. This section shows that this cost model *Static* combined with replanning produces behavior that is not human-like, and that has features prone to confuse observers about the robot intention, thus decreasing legibility.

A different cost model *ContextCost* is proposed later, which takes into account the dynamic nature of the navigation task and models human-aware costs in the context of the dynamic situation. This cost model produces more legible and efficient behavior in the crossing scenario. Note the goal in this work is not to find a psychological model for human behavior, but to implement robot navigation in a way that results in a similar behavior to humans. This means that humans might have very different strategies for navigation than the robot, but the demonstrated behavior should be alike.

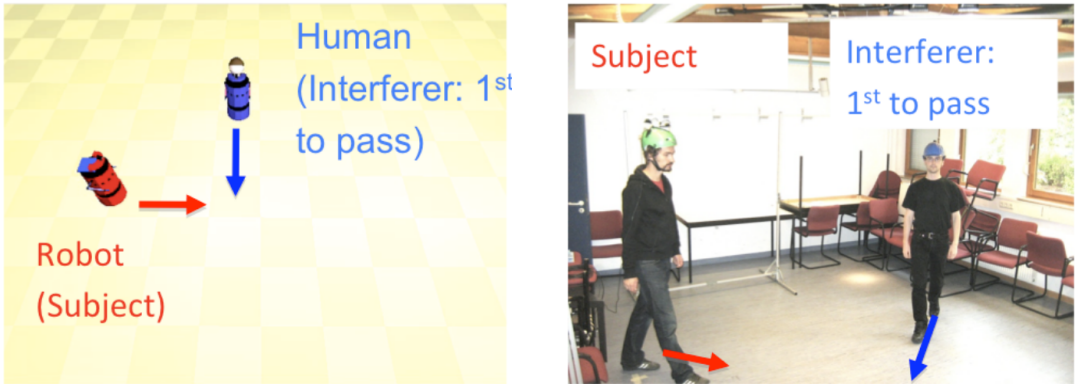
3.3.1 Static Cost model

Per default HANP uses a static cost model as described in Subsection 2.5. A robot moving in a dynamic environment may adapt to changes by replanning. Each time a new plan is generated, this plan is a best-effort attempt at finding a path that optimally balances path length and social rules.

The experiment described here is aimed to show that while each such planned path optimally balances path length and social rules, the resulting robot behavior of partly executing multiple changing paths is inadequate with respect to legibility.

The experiment was conducted on the Gazebo simulator, as described in section 3.2.5.

The robot's interaction capabilities were limited to mere motions, so no communication or signaling was possible between the robots.



(a) In Gazebo simulator using HANP planner

(b) In Laboratory

Figure 3.16: Experimental setup

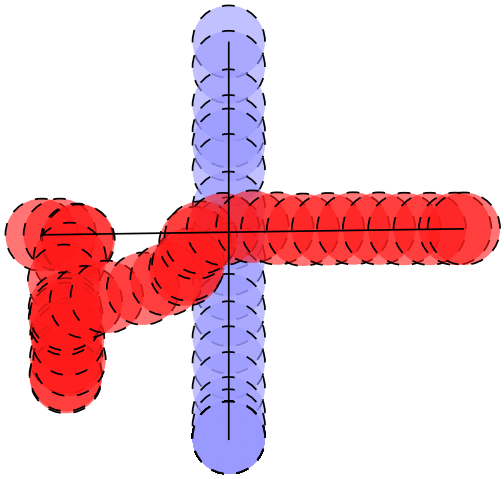
The blue robot on top in Figure 3.16a, labeled Interferer, always moved from the real world coordinates (1.5 m, 4 m) to the coordinates (1.5 m, 0.3 m). It was programmed to move in a straight line at its maximum speed of 0.2 m/s without avoiding collisions. The red robot on the left, labeled Subject, moved from the world coordinates (0 m, 2.4 m) to (3.5 m, 2.4 m). These coordinates were chosen to match the human-human experiment described in subsection 3.3.2.

HANP was used as the global planner with the costs defined in [135] with a grid width $\varepsilon = 15$ cm.

As a local planner, a PID controller was used for robot velocity commands in a local planner framework that would follow the waypoints calculated by HANP with a tolerance of 5 cm for smoother motions. The local planner projects its own motion and the human motion 5 s into the future, and selects a speed that avoids predictable collisions. The human motion was predicted linearly assuming constant speed and direction, while the robot motion was predicted using the global path returned by HANP.

The global planner duration was about 300 ms per global plan, and the robot replanned at a rate of 1 Hz while the interferer was moving (these values also hold when using the other cost models). The interferer moved on a fixed path without replanning.

(a) One sample



(b) Plot for 100 samples

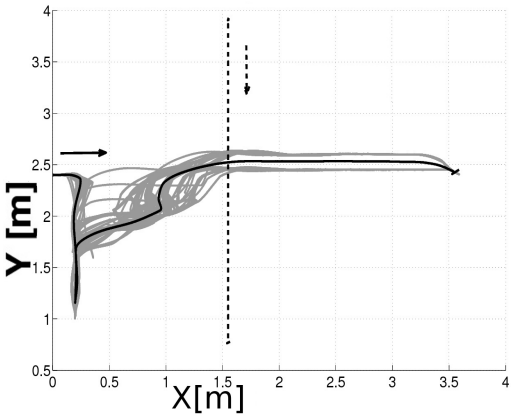
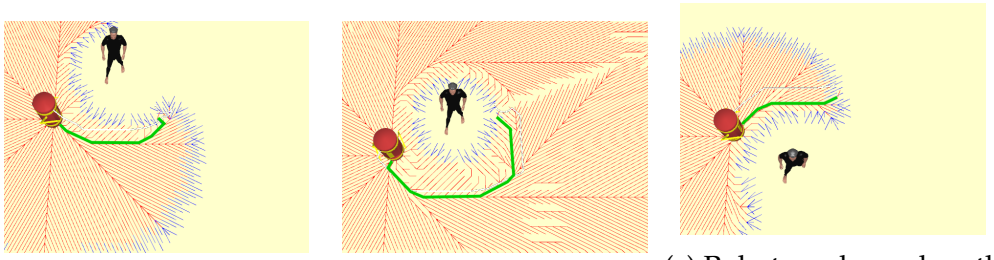


Figure 3.17: Robot behavior with cost model *Static*. Interferer moved from top to bottom, subject from left to right.



(a) Robot starts to deviate from straight line

(b) Robot deviates more, becomes stalled

(c) Robot replanned path deviates by going behind the interferer

Figure 3.18: Evolution of global plans during a sample trial with cost model *Static*. The lines on the floor are edges found during search. The thick line is the cheapest path to the goal. Note that the search tree was created with breadth search here rather than A* for mere purposes of illustration. In the trials A* search was used.

Evaluation of static cost model

Figure 3.17a shows a representative path taken by a robot in the simulation. It shows the (x, y) positions of the interferer robot (or the "human") and the adapting robot (the "subject") in regular time intervals. Figure 3.17b shows the average behavior over 100 trials. The variance in Figure 3.17b stems from natural dithering in the process starting times and the local planner.

The robot behavior seems not very goal directed in the first part of the experiment. This is explained in Figure 3.18, showing some of the plans the global planner generated by replanning. In an early stage of the conflict (Figure 3.18a), the global planner finds a cheapest path moving in front of the interferer in a certain distance as driven by proxemics. As the interferer moves forward, the new navigation plans take larger detours in front of the interferer (Figure 3.18b). At some point the path cost is minimized by a path around the back of the interferer (Figure 3.18c). But until this path was considered as more appropriate, the robot had already started to execute the previous plans passing in front of the interferer. How long the robot moves in parallel to the interferer before the global planner finds a solution that passes behind the interferer depends on the individual velocities.

With respect to legibility, an observer is able to observe that the robot perceived a change of the human position as the robot reacts to it. However the nature of the robot movement obfuscates the robot intention of moving to a specific goal, reducing the legibility. Also the behavior does not seem very efficient.

There are several possible solutions to generate more efficient behavior. The path could be straightened and discomfort avoided by modulating speed. Or a path curving around the back of the human could be chosen from the beginning. But without a formal model of navigation legibility, it is difficult to design an algorithm generating most legible ones. The next section describes an analysis of the same crossing situation when executed by humans, as also published in [6].

3.3.2 Human behavior in crossing situations

This section presents an experiment on human gait in crossing situations that was conducted as a joint research effort both for insights on human gait and for a comparison with robot behaviors. The experiment and its evaluation have been prepared and executed by Patrizia Basili. The results on human gait have been published as [6]. The comparison was also published as [81]. The contribution to this thesis is the connection to robot path planning algorithms and the creation of legible paths.

The experiment measured among other things the path taken by human subjects in a crossing situation, and the interpersonal distance they chose when avoiding a crossing interferer. Both measurements can serve to measure the similarity of robot behavior to human behavior.

In the experiment, participants had to walk across a room from one point to another, while an instructed interferer walked in the same room between a different set of points. The points were chosen such that the linear paths between the participants pair of points and the interferer's points crossed in the middle at 90°.

The interferer was always the same person, unknown to the participants before the experiment. The interferer was instructed to ignore the subject by not looking at her/him and by trying to be the first to pass. The subject's task was to reach a predefined goal position. That way, the subjects were indirectly induced to adapt to the interferer's trajectory in order to avoid a collision. The trajectories of the subject and interferer were recorded using a motion tracking system (IS-600 Mark 2, InterSense Inc., USA) with infrared and ultrasound signals at 150 Hz for the subject and 20–50 Hz for the interferer. Both persons were wearing a helmet with a tracking sensor on it. The size of the tracked area was 4 m × 4 m in the middle of a room of 38 m^2 (see Figure 3.16b).

In the experiments participated 10 subjects (6 females and 4 males) between 25 and 43 years old. The subjects were instructed to walk to a marked point at the opposite of the room, on a direct line from their starting position. Four trials were performed.

The situation was chosen in such a way that data for interpretation of human trajectory generation and decision making could be collected. This is in contrast to other studies such as by Pacchierotti [111] who performed a user study asking subjects for feedback after having passed a robot in a real world hallway, with the intention of evaluating robot performance only.

3.3.2.1 Results

The results showed a modification of the subjects' trajectory in terms of velocity adaptation rather than of path alteration. This velocity adaptation consisted in an initial deceleration and a subsequent acceleration within a time gap of one second, before the interferer had reached the intersection of the paths, i.e. the possible collision position (see Figure 3.19).

It is noticeable that the velocity profiles show two local maxima, meaning the humans did not prefer, or were not able, to find a trajectory that has just one acceleration and deceleration phase. For robots this means that it may not be necessary to always avoid multiple local maxima in velocity for similar situations, even if avoidance would reduce jerk.

Figure 3.20 shows an average minimum interpersonal distance of almost 0.5 m. At this time point the subjects were walking behind the interferer, i.e. outside his field of view.

The result shows that participants in this situation solved the spatial conflict by adapting their velocity rather than by adapting their path. Participants could have avoided the moving obstacle in the same way as they would avoid a static obstacle in the middle of the way, walking in a slight curve without reducing the velocity. Instead, they chose to remain on the straight line, and adapt their velocity.

The interpersonal distance chosen by the subjects was as low as 40 cm, measured from torso center to torso center. This is much closer than the humans would approach each other e.g. for interaction in such a context of relatively free space [5]. The corresponding zone in the scale given by Lambert [90] is the "Social Zone", given as 1.2 m to 3.6 m for e.g. conversations to non-friends. Even

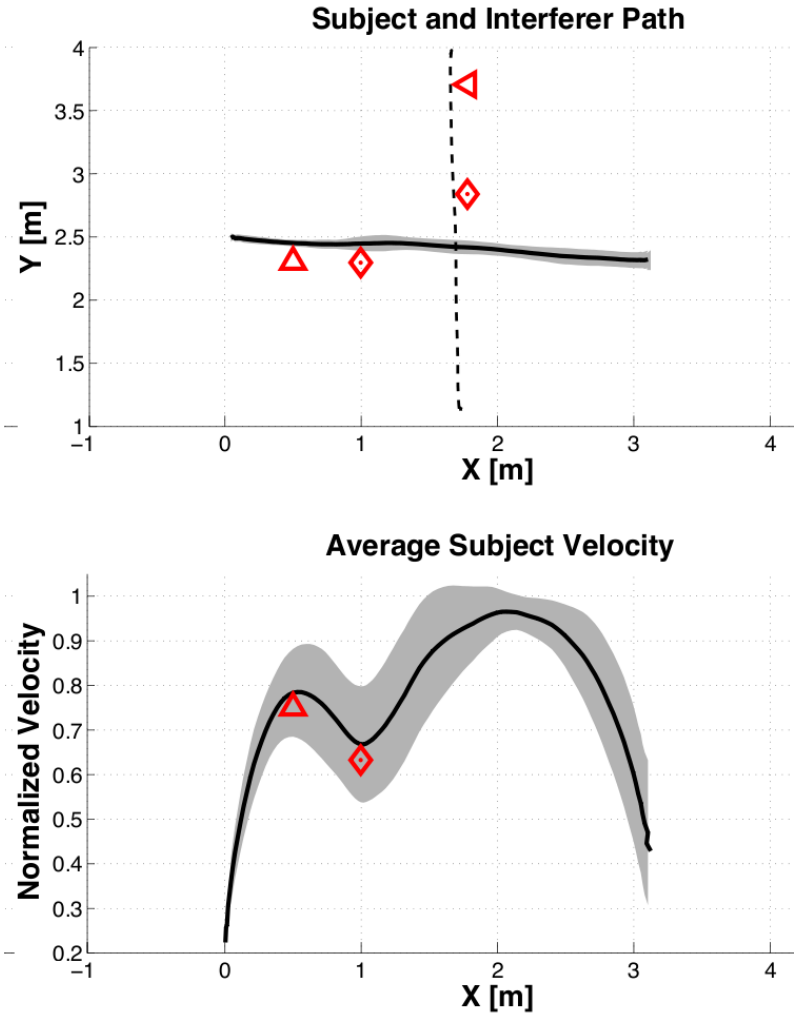


Figure 3.19: The average trajectory of the subject and interferer are shown on the top. The gray area represents the standard deviation over all the subjects' paths. The red triangle indicates, on average, where the subjects started to decrease their velocity, while the diamond where it started to increase again , i.e. the start and the end of the adaptation phase. On the bottom the average and standard deviation velocity are shown. Due to the fact that a person never stands completely still, the start of the motion was set to 0.17 m/s. Image taken from [81]

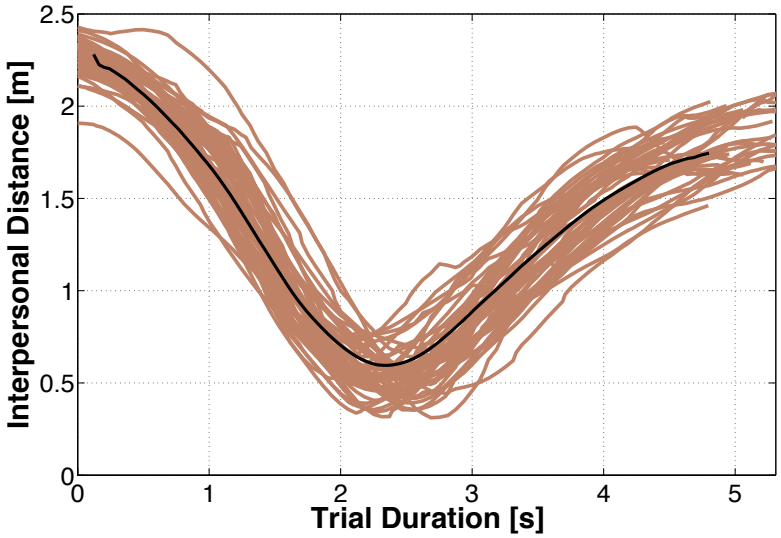


Figure 3.20: The brown lines represent the interpersonal distance over the duration of each trial while the black curve is the average interpersonal distance. Image taken from [81]

though this result cannot be transferred to all real world scenarios, it is an indication that humans will tolerate agents in less distance while moving than for interacting while standing: This may also be due to such crossing situations lasting only a predictably short time. For proxemics in general this could mean that for coming close to a person for predictably short durations is generally acceptable behavior for humans, which could also be true for robots. However, there may be additional restrictions on how to socially behave when being so close to other humans. Restrictions could be imposed on velocity, acceleration or motion direction, such that humans tolerate other humans at such a close distance only if the motion of those other humans are predictable and appear safe. Additionally human tolerance for other persons coming close may also depend on personal factors, similar to the original proxemics model.

In any case the experiment showed velocity adaptation instead of path adaptation for the avoidance of another human. The next subsection looks at recreating this strategy for robots with human-aware path planners.

3.3.3 Context-dependent Cost Model

As shown in Subsection 3.3.1, cost model *Static* produces behavior in certain conflict situations that does not seem legible. Based on the results of the human experiment in Subsection 3.3.2 the robot path selection behavior is also not natural in the sense of being not human-like. One attempt at improving both legibility and naturalness based on the human experiment is to change the robot behavior such that in certain situations of spatial conflict, the robot prefers to adapt the velocity, rather than the path, to solve a conflict. as an example, by simply not considering humans in the vicinity as obstacles, a robot always chooses the direct path to a goal, and a local planner can adapt the velocity on the path to reduce the velocity. However trivial counter examples show that this cannot be a general strategy: When a person and a robot advance towards each other, the robot should mot probably not stay on that path, but rather stand aside, to act in a human friendly way.

A new approach to calculating social costs during path planning incorporates these ideas. This cost model is called *Static*.

The cost model *Static* is adjusted in the following to generate human-like paths in global planning. Two important features were required: The participant's path did not deviate from the direct line to the goal, for the search this means that the costs to the goal would have to be minimal on the straight path to the goal in this case. Also even without experiment it is evident that a human subject would deviate from the direct line if an interferer were walking towards the subject from the front. So the costs to the goal have to be minimal on a deviation from the straight line to the goal in this case.

This observation led us to a new cost model that does not only define a penalty on grid cells as HANP, but reduces this penalty based on the direction this grid cell is traveled to during search. This allows for social costs to exist around humans, but to reduce these social costs to zero in specific cases.

As an example, in HANP a grid cell 50 cm close to a moving human might have a numerical penalty of 42 during search regardless of search context. With contextual cost, the same cell could have costs 42 for a path approaching the

human from the front via this cell, meaning discomfort for the human, but costs 0 for moving in the same direction as the human via this cell. Similarly a whole path does not have the same costs regardless of what direction it is traveled in. Following a human on a path going in the same direction as the human has less costs than attempting to pass through the human in the opposite direction on the same path. This intuitive relation between path and direction of motions can not be captured in HANP nor in any other cost model with fixed costs around humans.

Based on these context-dependent costs of cells, a heuristic function is constructed in the following to reduce costs in situations such as for crossing, but not in all situations. The construction first requires the concept of compatibility of paths.

The paths of two agents can be considered *compatible* if they can be followed by both agents concurrently with velocities $v \geq 0$ at all times such that no deadlock occurs, the distance between the agent remains above some threshold, and both agents eventually reach the end of their paths. Any two agents on compatible paths can reduce their planning efforts to modulation of forward speeds along the path, reducing their "cognitive effort". Some explicit communication or other protocol may help to optimize the joint motions, such as deciding who goes first.

In other words, the more any agent has to deviate from the intended path due to the intended path of another agent, the less compatible two paths are. In general agent intentions remain unknown between agents, but may be estimated from observation. So compatibility of paths can also only be estimated because prediction of human motion is not a solved problem. Research approaches to predict human motion for robotics are presented in [34, 149, 46, 87], but the approaches still have too many constraints to be generally applicable. For this thesis it was sufficient to use a linear projection of the human path based on his current velocity vector.

This notion of compatibility allows us to modify a path planner to only deviate from a straight line towards the goal, if the straight line segment is not compatible with an estimated intended path of another agent. Currently using HANP,

a path adaptation during planning is caused by social costs applied to path segments. To avoid path adaptation in specific situations, the social costs have to be discarded in such situations. In other situations, they may still apply and provide path adaptation where it is appropriate.

The concept of compatibility is formalized by defining a function for social costs on grid cells depending on the direction each cell would be reached given its predecessor cell in the path. As a consequence, the costs in each cell cannot be calculated independently of the search context. The original cost functions of HANP define for each cell the social costs for the robot to move through this cell. For each search context, this number is defined based only on the positions of all agents. However, with compatibility, the cost value of a cell varies during search as the planning algorithm tries to reach the cell from several predecessors.

This is achieved by replacing the cost function $\varsigma(H, w_i)$ in a grid cell for one human with a function $\varsigma'(H, w_i, w_{i-1})$ taking also into account the last waypoint w_{i-1} on the currently planned path $\tilde{P} = (w_1, \dots w_{i-1}, w_i)$. The final cost function $\sigma(\mathscr{H}, w_i)$ considering the set of n humans $\mathscr{H}$ is replaced accordingly with a cost function $\sigma'(\mathscr{H}, w_i, w_{i-1})$.

As social costs can only be reduced in comparison to HANP, the maximum possible costs for any cell w_i in cost model *ContextCost* are the original costs of HANP: $\forall \tilde{P} : \sigma'(\mathscr{H}, R, w_i, w_{i-1}) \leq \sigma(\mathscr{H}, w_i)$.

As shown in Figure 3.21, two factors are used to determine estimate path compatibility in a waypoint: distance and heading angle. If the minimum distance d_P between the waypoint and the estimated path of the human is large enough, compatibility is granted. The human can move along his intended path without the robot being in the way. If the distance is small, the paths may still be compatible, if robot and human don't intend to move towards each other. So the second factor is the difference in heading angles α. The angle α is determined by the heading of the person and the heading of the robot at the potential waypoint w_i given by the cell of the waypoint w_i and its previous waypoint w_{i-1}, as shown in Figure 3.21 and equation (3.4). The vectors $\hat{hd}^{-1}$ and $\hat{rd}$ of those directions serve to get the angle, normalized to the domain $[-\pi, \pi]$:

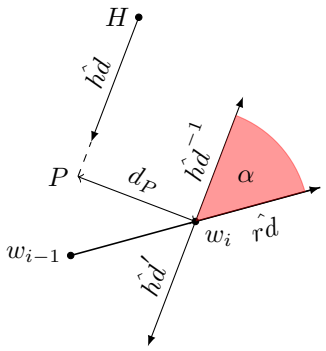


Figure 3.21: Context dependent costs: H: current position of human, $\hat{hd}$: heading direction of human, $\hat{hd}'$: translated human heading, $\hat{hd}^{-1}$: inverse translated heading. w_i: potential way point on path, P, closest point on projected human path to w_i, d_P distance between P and w_i, $\hat{rd}$: potential heading of robot arriving at waypoint w_i
Whether social costs apply depends on distance d_P and angle α between $\hat{rd}$ and $\hat{hd}^{-1}$. If both d_P and α are small, paths are incompatible, if either is large enough, paths become compatible.

$$\alpha = norm(acos(\hat{hd}^{-1} \cdot \hat{rd})) \tag{3.4}$$

Incompatibility is then calculated by function ϕ, that calculates a number $\in [0, 1]$ based on the angle α and distance d_P. An incompatibility of zero means paths are compatible, so social costs leading to path adaptations do not need to be considered for a given grid cell. An incompatibility of one means paths are maximally incompatible, and path adaptation should be attempted by applying social costs functions. This function ϕ can easily be integrated into HANP by multiplying the costs of cost model *Static* with ϕ. The function ϕ has three parameters to tweak what motions are considered compatible.

$$\phi(H, w_i, w_{i-1}) = \begin{cases} 1 & \text{, if } \hat{hd} \text{ undefined} \\ 1 & \text{, if } d_P \leq d_{low} \\ 0 & \text{, if } d_P \geq d_{high} \\ 0 & \text{, if } \alpha \geq \alpha_{max} \\ \frac{d_P - d_{low}}{d_{high}} \cdot \frac{\alpha}{\alpha_{max}} & \text{, otherwise} \end{cases} \tag{3.5}$$

As shown in equation (3.5), for distance d_P, anything below d_{low} is considered spatially incompatible, and anything above d_{high} is considered spatially compatible. In this thesis the values $d_{low} = 1\,\text{m}$ and $d_{high} = 2\,\text{m}$ are used. For the crossing experiment, using different values did not change the behavior a lot, but in other situations it might. Similarly, angles α greater than α_{max} are considered directionally compatible.

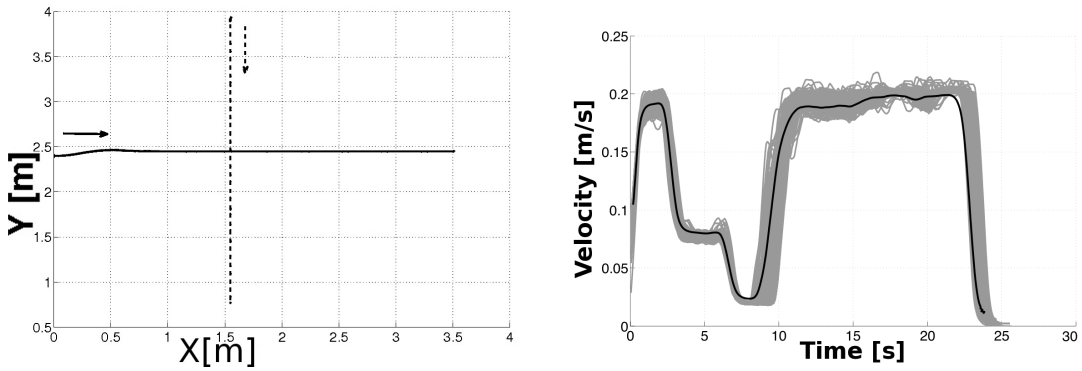
The case of $\hat{hd}$ undefined relates to humans for which no direction could be calculated, e.g. standing humans.

For all incompatible value sets, a linear function is used to calculate degrees of incompatibility. As seen in the human-human experiment, at angles around 90°, the subject did not deviate from the shortest path. To approximate that behavior, and angle $\alpha_{max} = 80°$ was chosen in this thesis. This means incompatibility zero is returned for angles greater than this.

$$\varsigma'_{ContextCost}(H, w_i, w_{i-1}) = \varsigma_{Static}(H, w_i) \cdot \phi(H, w_i, w_{i-1})$$

The cost model *Static* is modified by replacing ς_{Static} with $\varsigma'_{ContextCost}$ as seen in equation (3.3.3).

For cost model *ContextCost* the original neighbor predicate of Figure 2.10a was also modified: For all search states except those neighboring the start cell, the neighbors expanded during search would only be those where the angle in a grid cell to the previous state and to the next state was no bigger than 90°. So the planner could not plan to turn in points at angles greater than that. This means of the 16 directions given in Figure 2.10a, only 10 were used, as shown in Figure 2.10b. The main reason for doing so was to eliminate a phenomenon that can occur as a result of the directional costs modifications with *ContextCost*.



(a) 100 samples with cost model *ContextCost* (b) Subject robot Velocity

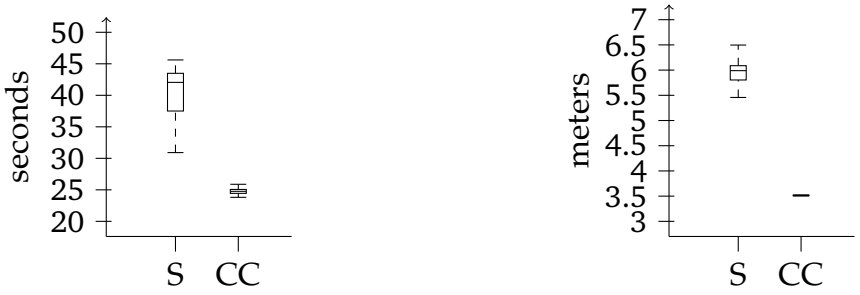
Figure 3.22: Robot behavior with cost model *ContextCost* for the crossing scenario. Interferer moved from top to bottom, subject robot from left to right.

When costs in a grid occur only in one direction, then in some cases the least cost path found is a zigzagging path. This looks similar to "tacking" motion of sailing ships when trying to move against the direction of the wind. Such a path can move against the direction of a moving person without accumulating social costs, but still accumulating costs for the length of the path. By reducing the allowed turning angles, social costs are still accumulated in the turns, and those zigzagging paths did not occur in our experiments anymore.

It should be noted that dynamic obstacle avoidance using temporal planning may also produce robot behavior that stays on a rather linear path, but there are several issues with this. Temporal planning requires very good quality perception data of the human to work reliably, it does not scale well with the distance and the amount of agents. Also when the predictions of the human partner in the conflict oscillate (both due to sensor noise and actual human hesitations), the result of temporal planning would also start to oscillate.

Evaluation of Cost Models Static and ContextCost

The same evaluation experiment was performed for cost model *ContextCost* as for cost model *Static*. Usually in navigation experiments the approaches are compared with respect to efficiency. Efficiency is one factor that can benefit legibility. Inefficient behavior implies unnecessary actions or, in the case of navigation, detours. So an approach that is very inefficient is likely to also have low legibility,



(a) Time taken

(b) Distance traveled

Figure 3.23: Statistics for 100 runs for each cost model for the crossing scenario. S is for *Static* and CC for cost model *ContextCost*. The measures are visualized in the figures as boxplots, the box indicating the range where 50% of all values occurred, the line in the box being median, and the "whiskers" representing the minimum and maximum values, except for outliers.

when talking about productive tasks (In certain contexts, it is possible that low efficiency may increase legibility, such as a robot expressing uncertainty about state by acting slowly). In addition to efficiency, the similarity of the robot behavior to the human behavior is compared.

In the crossing experiment the cost model *ContextCost* shows the same general behavior as the human in the human-human experiment: The angle α between the heading of the person hd and the movement direction of the robot rd is always close to 90°, shown in Figure 3.22a. A collision between both agents was prevented by speed modulation of the subject as seen in Figure 3.22b.

Figure 3.23 shows the durations and distances needed to reach the goal position in the crossing scenario for the two cost models. In 100 trials, cost model *ContextCost* performed in a stable way despite natural dithering in the simulator. The statistics also show that efficiency was improved over the straightforward cost model *Static*, as the time taken with both alternative strategies dropped from around 40 seconds to 25 seconds. Without interferer, at the maximum speed of 0.2 m/s allowed for the experiments, the robot would cover the same distance in 17.5 seconds.

So the robot behavior with *ContextCost* is both more efficient and more human-like than *Static* for certain dynamic cases. The robot behavior in static cases remains unchanged from cost model *Static*.

Legibility in navigation scenarios was defined as a measure of how well a robot's motion intentions (and beliefs) can be estimated by an observer of the robot behavior. A simple robot has only a limited set of modalities available to convey beliefs and intentions. As an example the pointing direction, the motion direction, velocity and acceleration, and variation patterns of all of those. For a robot using cost model *ContextCost*, the direction that the torso is pointing to at all times shows the intended paths and the direction of the eventual goal. As the intended path is a straight line, it can be conveyed to others just using the torso pointing angle. So staying on a straight line whenever possible makes it possible to indicate the intended path just by the torso direction, whereas curved paths would require additional actuators to indicate an intended path, e.g. by moving in one direction while gazing into another direction.

The robot velocity pattern (slowing down) can indicate to an observer that the robot is aware that a person requires the space at the crossing center, and that the robot intends to cede that space.

The cost model *ContextCost* showed seemingly more legible behavior in the simulator than mere replanning. However this does not prove an improvement of robot behavior. Therefore the next section presents a user study to validate that the resulting robot behavior is rated significantly better by uninstructed persons.

3.4 User study on crossing behavior

The work in this section has partly been published in [82].

The adapted cost function *ContextCost* has proved to produce robot behavior that is visually similar to the behavior of that of humans in crossing situations. A further evaluation is done in this section to establish whether uninstructed

Cost model:	*Static*	*ContextCost*
Path Behavior:	Changing paths	Straight line
Velocity adaptation:	Maximum possible speed without collision	Reduce speed to keep distance

Table 3.1: Comparison of experiment conditions

humans can perceive the difference and whether the behavior using *ContextCost* is perceived as an improvement.

For that purpose a user study is presented here that puts participants into a crossing situation with a PR2 robot [145]. Within a holistic perspective on technology acceptance such as established by Heerink et al. [58], the study focuses on functional acceptance. Similar user-studies investigating optimal parameters for robots approaching standing persons have been conducted [17, 19, 144]. Sardar et al. [126] performed a user study where a robot approached a standing person concentrating on another task, and found significant differences in socially normative behavior for robots and humans. A study by Pacchierotti et al. [111] also looks at a mutually dynamic situation of passing each other.

The user study presented here examines if the seemingly confusing robot behavior with the ad hoc cost model *Static* observed in simulation also happens in the real world, and whether the cost model *ContextCost* improves the perceived quality of the motion of the PR2.

The PR2 was run using its default controllers based on the ROS [119] robotics libraries. For the experiment, the "move_base" navigation framework of the PR2 robot was modified. The "move_base" framework defines a scheduling process invoking a global path planner for global path planning and a local planner for velocity selection and obstacle avoidance. As a library, "move_base" includes a default global and local planner. Both the path and local planners were replaced for the experiment.

As a global planner HANP was used to enable human-aware path planning. It was invoked at a regular frequency of 10 Hz. HANP was configured with one of two cost models $\varsigma'_{ContextCost}$ and ς'_{Static} as a parameter of the experiment trials.

As a local planner, for this experiment the same waypoint-following algorithm was used as for the experiment in simulation used in Section 3.3. The waypoint follower causes the robot to visit each waypoint of the global plan, by turning towards each waypoint in turn and then moving forward to reach it (as opposed to allowing sideways or backward motions). Velocity is reduced up to zero in case the projected distance to a human is below a threshold, but the robot does not evade to prevent collisions.

The strategy of velocity adaptation was varied according to the global planning strategy. When using the simple proxemics based cost model *Static*, the local planner tried to move at maximum velocity on the path, except when a human was in front of it at 1 m distance (center to center), in which case no forward motion was allowed. The latter is a measure to grant a minimum feeling of safety for human participants. Using the cost model *ContextCost*, the velocity was adapted by projecting the motions of the robot and the human into the future, and selecting a velocity that would in those projections not predict their mutual distance to decrease below 1.3 m. This value was established in preliminary trial as yielding a distance that felt safe while keeping the robot close enough to establish a crossing conflict situation.

So in the experiment the robot used one of two strategies in each trial, and each strategy changes the cost function for the path planner, and the velocity adaptation mode of the local planner. The differences are summarized in Table 3.1. In more simple terms, the result of applying the two compared robot strategies in a 90° crossing situation can be described as follows. In the cost model *Static*, the robot attempts to move always at the maximum possible and allowed speed staying on the current path, adapting the path to not cross the current human position nor an area in front of the human. The path frequently changes as the human moves, causing the robot to swerve. Using the cost model *ContextCost*, the robot attempts to follow a straight path to the goal, but reduces its velocity before reaching the crossing point, and accelerates again once the human has advanced enough.

The robot head was moving during the experiments to avoid the impression that the robot is ignoring the human. The control scheme turned the robot head to point at the participant head whenever the robot head could point towards

the participant without deviating more than 90° from the frontal position, and vice versa for the participant. In other words, the robot tried to establish eye contact when this was possible for both the robot and the participant without looking over their own shoulder. Otherwise the robot head pointed straight forward. When moving, the head moved slowly at 0.3 rad/s. Based on participant remarks this appeared appropriate except in rare cases when the robot was very close and moved in front of the participant, in which case the gaze seemed a bit provocative, which is similar to other results in robot gaze studies [139]. The participants were informed before the experiment that the head will move, to prevent the discovery of this from distracting. The robot arms remained folded during all the experiment in the default PR2 "tucked" position. Only two participants mentioned this afterward as a bit unnatural, as opposed to making prompting gestures.

The user study was set up to determine whether the robot behavior with simple static cost models in crossing situations can be reproduced in the real world, and whether the context-dependent cost model visibly improves the robot motion quality, such as its legibility.

Legibility is difficult to quantify, as a robot may behave very legible with respect to one aspect of internal state, but at the same time obfuscate another aspect of internal state. Therefore the experiment avoids to ask participants directly to estimate aspects of internal state. Instead it establishes a good knowledge of the robot internal state for the participants, then showing how the participants can become confused by the robot behavior under assumptions about the robot internal state. In other words, the robot observable behavior is judged considering a given assumed internal state. The internal state established here was simply the knowledge of the participants that the robot intends in each trial to cross the room at the same time as the participant, while avoiding collisions with the participant.

17 participants were given the task to act as interfering walkers with a robot moving from one spot to another in an area without static obstacles. The human participants were tracked using an infrared motion capture system, based on four passive infrared markers placed on a lightweight helmet the participants wore and 10 wall-mounted cameras, as shown in Figure 3.25. The room has an

experimental area with offices and desks surrounding it, during the experiment people were quietly working around the experiment area.

The participants where all given the same set of written instructions to read, based on their choice in either English or French. The instructions stated that the experiment is about robot navigation, the hardware involved, and the procedure to follow. The instructions stated that multiple different robot strategies would be used, but not how many or what variation to expect. The participants could ask for clarification of the written instructions, but were not given further information about the experiment.

The procedure was the following: In the preparation stage, the robot and the participant moved independently to their starting positions. The starting position of the participant was marked on the floor. This situation can be seen in Figure 3.25. When both were ready, the experiment instructor pressed a button to start the execution phase. This triggered the robot motion planning and also made the robot say "go". The participant and the robot would start moving towards their respective goal positions at roughly the same time. The map in Figure 3.24 shows the coordinates. The participant goal was a small shelf on which lay a questionnaire. Going towards a shelf for a purpose rather than going onto a marked spot seemed to benefit a natural walking behavior. The goal area of the participant was slightly elevated (10 cm), and there was a low barricade with a gate spanning 1.5 m, seen in white in Figure 3.25. This was an existing feature of the room, which was included to gain a sufficiently large area for the motion capture system. The participants started their movement at coordinates (0.5, -3.5) on the map, but would not be visible to the motion capture system before reaching roughly (2.5, 3.5). The maximized the walking distance appeared more valuable for this experiment because the robot motion quality was to be established, not properties of human gait.

When the participants reached the shelf, they could then immediately rate the robot behavior on the questionnaire placed on the shelf. Once the participant and the robot had both reached their goal positions, the execution phase ended. Then a new cycle would begin with the preparation phase. An instructor was present and visible during the experiment, sitting at the desk shown in Figure 3.24, to react in cases of emergencies or disturbances, but also to reassure the participant

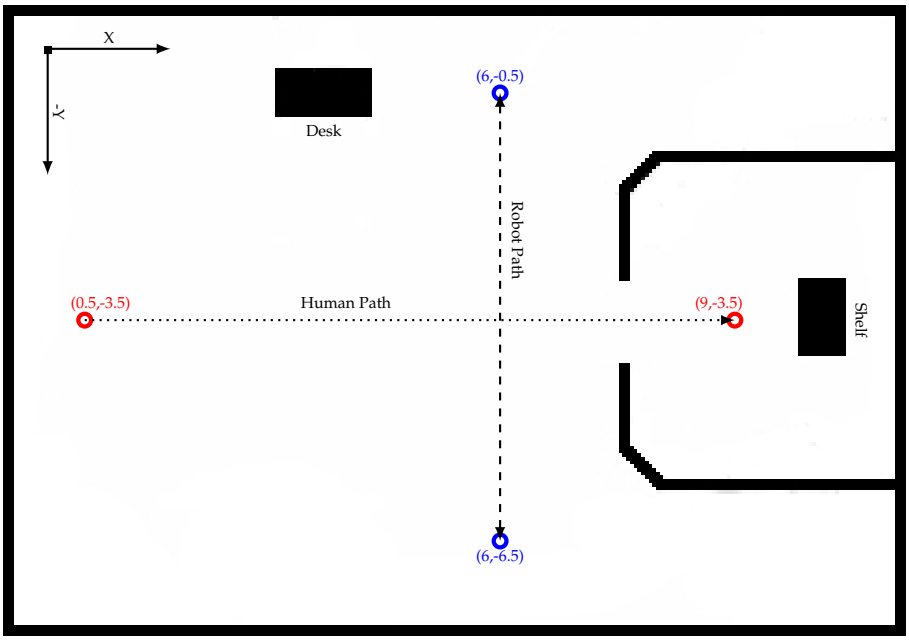


Figure 3.24: Top-Down view on the room layout, showing the experiment path, the instructor's desk, and the shelf where the participants marked their answer after each trial

and verify they were following the procedure. Also the instructor modified the robot maximum speed between 0.45 m/s and 0.55 m/s according to the participant's walking speed during the first 3 trials to create a spatial conflict without imposing a walking speed on the participants. The robot would approach alternately from the left or the right hand side.

Each participant was asked to perform the same task 13 times, and was told that the first 3 cycles would not be taken into account, but serve to get used to the situation. The robot strategy was randomized but never the same three times in a row. All participants were presented with both strategies to allow them a comparative opinion.

Ample time was given for remarks and questions of participants immediately after their last trial. Participants were not offered any financial reward for participating, but sweets were given as a minor token of gratitude.

As opposed to studies on human gait, the experiment was interactive, meaning the robot behavior is a function of its algorithms, the person's behavior (in

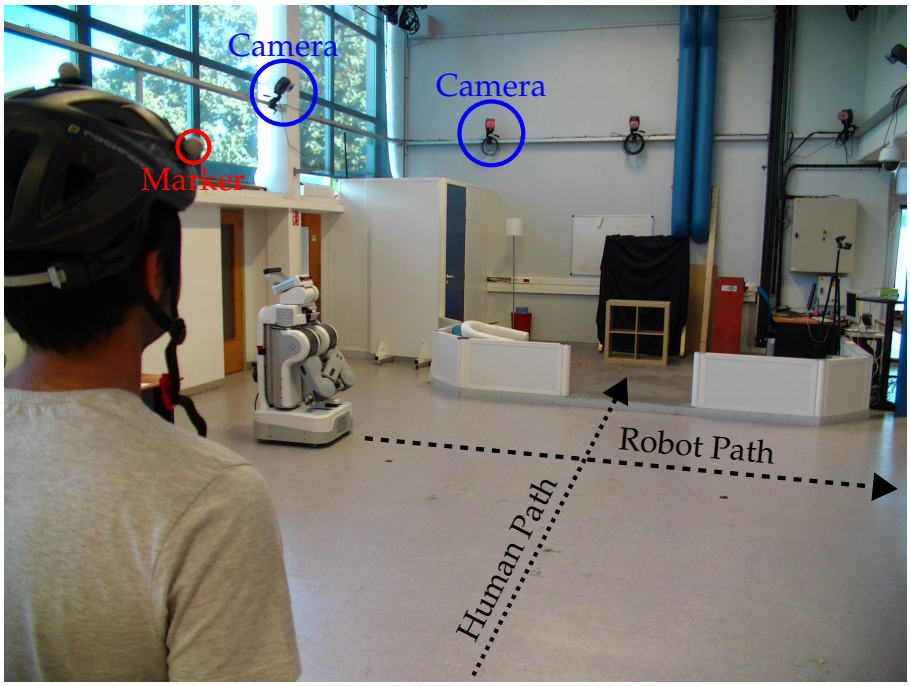


Figure 3.25: Experiment setup showing wall-mounted cameras (2 out of 10), participant at start position with helmet and passive marker (1 out of 4), robot at start position, the intended paths crossing at 90°

particular the walking speed), the adapted robot maximum velocity and of noisy technical side effects like varying WiFi network latencies. Because of that, the captured data of the human trajectories is only of limited usefulness to interpret exact motion patterns.

3.4.1 Failed Evaluation

A first attempt at performing the user study failed to produce a useful comparison of the different cost models, since the acceleration behavior created by the local planner dominated the reactions of the participants. This subsection describes the lessons learned from the attempt and the modifications made, before the evaluation was conducted again with a different set of participants.

In particular two main problems surfaced in that evaluation, when controlling the robot velocity: “stopping late” and “restarting early”.

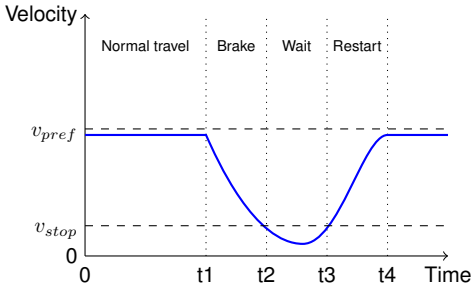


Figure 3.26: Idealized velocity profile scheme for 90° crossing situations with cost model *ContextCost*. v_{pref} indicates the preferred travel velocity, v_{stop} a velocity so low a robot is considered not traveling.

Figure 3.26 shows a velocity profile that appears optimal for crossing the path of a human when giving way. It describes the velocity of moving forward, not rotations. There are three distinct phases for braking, waiting, and restarting. Ideally, there is no acceleration during the "Brake" phase, and no significant acceleration during the "Wait" phase. So the ideal robot behavior cannot merely be described in terms of distance and velocity, but acceleration (and possibly jerk) can have significant effect on humans nearby.

The phases should be clearly observable by the human whose path is being crossed, and the timing of the phases can be crucial to human comfort. In the failed first attempt at evaluation, the robot would sometimes initiate the braking phase too late, sometimes re-accelerate slightly during braking and waiting phases, or initiate the "Restart" phase too early. Each case caused some participants to feel uncomfortable, and the discomfort was strong enough to significantly influence the participants' answers. This robot behavior was not planned nor explicitly parametrized, but a consequence of noise and latencies in the whole perception action loop, and the human motion prediction accuracy.

Figure 3.27a shows the transition to the "Brake" phase. When the robot decelerated late or had the slightest acceleration motion, participants felt the situation was uncomfortable. It seem likely that participants feared the robot may accelerate to full speed again. For the second attempt at evaluation this symptom was suppressed mostly by adding uncertainty to the future human velocity, which causes the robot to be more conservative about its choices of velocities. Figure 3.27b shows the "Restart" phase. Sometimes the participants felt uncom-

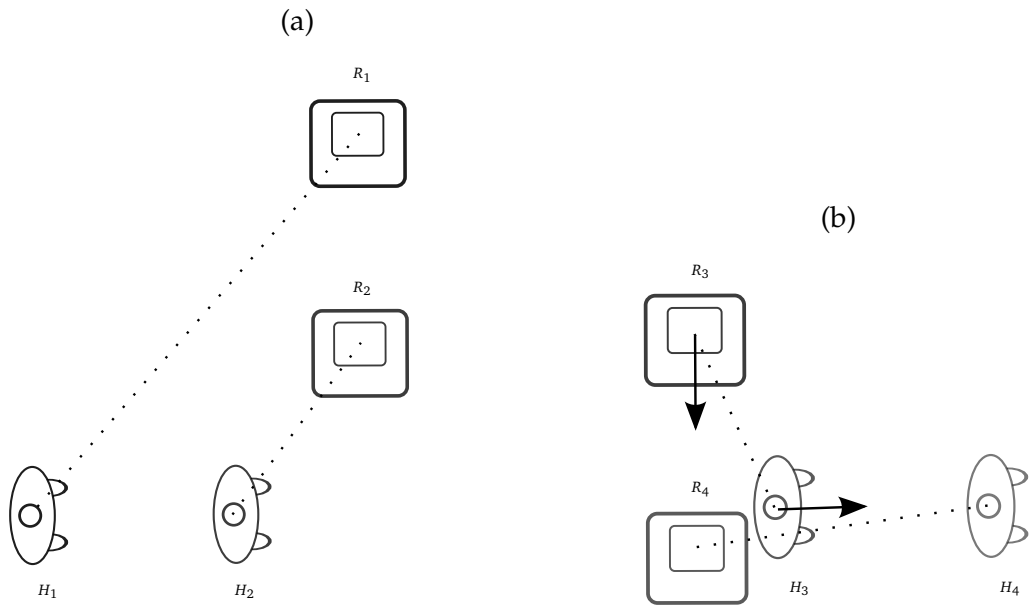


Figure 3.27: Situations relevant for comfort while crossing with adapted cost model *ContextCost*. R_t and H_t are position of the robot and human at time-step t respectively. Stopping late in (a) or starting early in (b) cause discomfort.

fortable expressing the robot moved "too early". It seems likely that the participants feared the robot quickly accelerating to its full speed, this behavior was suppressed by projecting the robot position not at the desired speed, but at full speed, which caused the robot to accelerate later, and did not discomfort the participants anymore.

Using both increased uncertainty about the future human positions and projecting the robot at full speed, the robot behavior in the crossing situation became rather similar to the scheme in Figure 3.26, except for a very small re-acceleration during the "brake" phase, which was noted by participants, but did not have strong impacts on their responses.

Additionally, for the second attempt at evaluation, the instructions and questionnaire were adapted. In the second evaluation the instructions stressed more that the intention of the experiment was not to validate whether the robot acts generally nicely or not, but rather to compare different strategies, thus encouraging participants to use the full scale, instead of shying away from using very

negative ratings. The instructions in the second attempt also explicitly noted that the robot head will move in all trials, as some participants reported that they only noticed the head moving after a few trials, and got distracted by trying to remember whether it had moved in all trials or not (the robot head always moved). As a technicality, the scale for the answers was reduced from range "1 to 10" to range "1 to 10", after participants commented that the perceived differences would be easier to classify on a shorter scale. The instructions and form used are listed in appendix B, Figures B.1, B.2, B.3, B.4, B.5, and B.6.

3.4.2 Second Evaluation

With the modifications to the velocity adaptation and evaluation procedure, a second attempt was made to evaluate the difference between cost models *Static* and *ContextCost*.

The participants were students working in the LAAS CNRS lab on computer science projects.

- Number of Participants: 2 female, 15 male
- Age range: 22 - 34
- Mother language: 11 French, 6 Other
- Education level: 11 Master, 5 PhD
- Years of robot experience: 11 None, 6 with less than 6 years

After each run, the participants were asked to rate the robot's performance according to predefined questions:

- Please rate the robot behavior (clear vs. confusing)
- Please rate the crossing situation (comfort):

The participants were given a Semantic Differential Scale from 1 to 5, the extremes labeled "clear", "confusing", "comfortable", "uncomfortable" respectively. A preliminary trial had asked the participants to rate their "surprise", but this word also had positive connotations and was also rated for the novelty of the whole situation. "clear" vs. "confusing" seemed better to express legibility issues.

Based on the design of the cost model *ContextCost*, the hypotheses for the outcome of the experiment were the following:

- H1: The robot behavior observed in simulation for *Static* also occurs in the experiment
- H2: Participants rate path adaptation as more confusing than velocity adaptation
- H3: Participants rate path adaptation as more uncomfortable than velocity adaptation

The alternatives are trivially negations of the given hypothesis. H2 and H3 would be confirmed if significant difference between the reported ratings were found.

3.4.3 Results

Data was collected from 170 valid trial runs as shown in Table 3.2. Figure 3.28 shows summary plots of a representative run using cost model *Static*, and Figure 3.29 shows the same plots for a representative run using *ContextCost*. Figures 3.28a and 3.29a show the positions of robot and human over time for two representative samples. The starting position that was outside the perception range is indicated as a single circle in the plot. Figures 3.28a and 3.28f shows how the robot at a distance of 1.2 m slightly deviates from the straight line towards its left and rotates.

The human passes during that time. Once the human has passed, the robot plans paths behind the human as shown in Figure 3.28b, thus turning to its right before eventually regaining the straight path. Compare that to Figures 3.29a and 3.29f, where the robot also decelerated to a minimal distance of 1.2 m, but never rotated, and never planned anything but a straight path as shown in Figure 3.29b.

	Static	*ContextCost*
robot start at (6, -0.5)	39	45
robot start at (6, -6.5)	41	45

Table 3.2: Conditions for the 170 valid trials.

The former behavior was explained in Subsection 3.3.1, and could now be observed on a real robot. The latter behavior is as intended by the directional cost model.

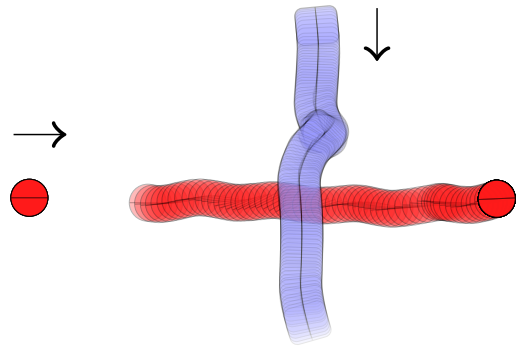
In Figure 3.29d at time 3.8 there is a small increase of velocity. The cause of this remains unknown, it could be with the prediction algorithm or robot wheel control. It is noteworthy here that such a small and short acceleration already counted as a cue to some participants that the robot was maybe re-accelerating.

Figure 3.30 shows the differences in reported discomfort. The visible difference as given by a one-way ANOVA indicates trials with cost model *ContextCost* were rated more comfortable (M=1.5, SD=0.723) than for *Static* (M=2.873, SD=1.265), $p < 0.001$. The same is shown for behavior clarity in Figure 3.30. The trials with cost model *ContextCost* were rated more clear (M=1.522, SD=0.707) than for *Static* (M=2.684, SD=1.215), $p < 0.001$. The ratings for clarity and discomfort were also correlated as revealed by Pearson's R test (R=0.771, p<0.001). In their remarks, some participants usually argued that both values were independent. They explained that they rated strangeness whenever the robot orientation changed (as it only did with cost model *Static*), while they rated discomfort high when the robot passed first (only happened with *Static*) or had jerky acceleration motions (only happened with *ContextCost*). However, given the strong correlation possibly most participants did not clearly separate both concepts.

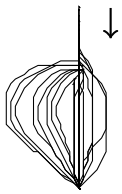
The hypotheses were thus validated follows:

- H1: Yes, the robot shows swerving path behavior with *Static* also occurred in the experiment
- H2: Participants rated path-adaptation as more confusing than velocity adaptation
- H3: Participants rated path-adaptation as more uncomfortable than velocity adaptation

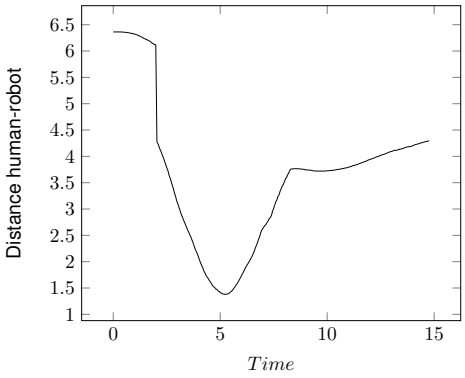
Hypothesis H1 is validated by the robot path behavior, as displayed in Figures 3.28 and 3.32, in comparison to Figure 3.29a. The distinct responses for clarity and comfort validate Hypotheses H2 and H3.



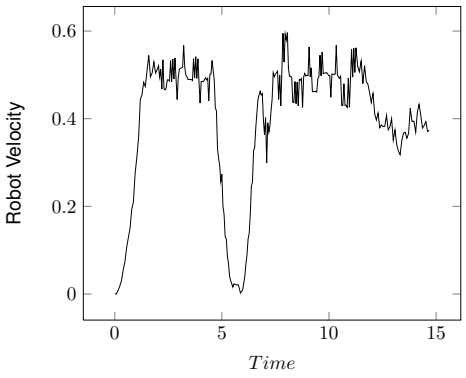
(a) Paths of human and robot



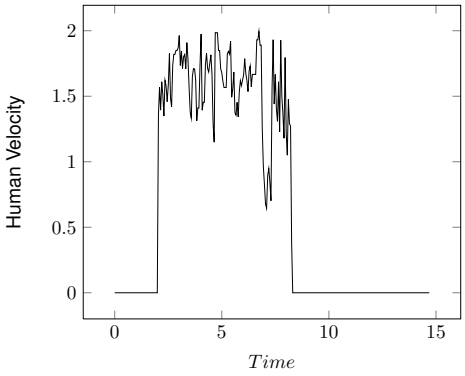
(b) Robot plans over time



(c) Distance human to robot center



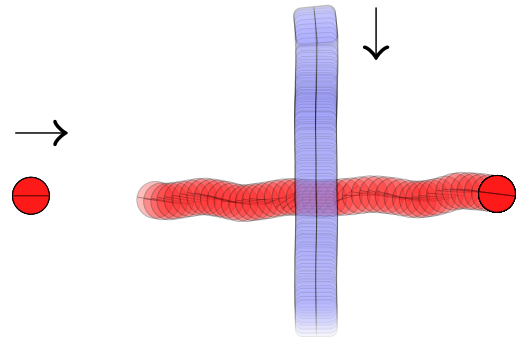
(d) Robot velocities



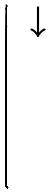
(e) Human velocitities

(f) Position at minimum distance

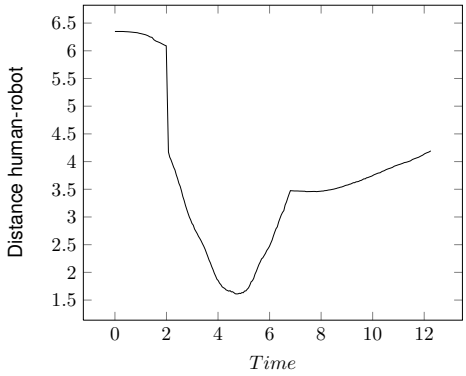
Figure 3.28: Sample 1 with cost model *Static*. In (a) the round positions are the human moving from left to right. (b) shows what the diverse plans the robot generate while moving.



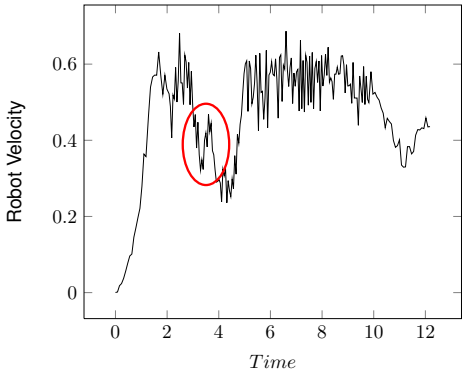
(a) Paths of human and robot



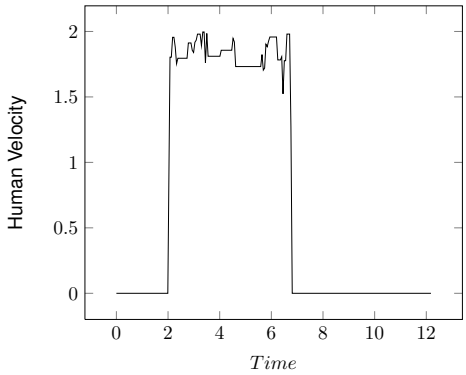
(b) Robot plans over time



(c) Distance human to robot center



(d) Robot velocities



(e) Human velocitities

(f) Position at minimum distance

Figure 3.29: Sample 2 with cost model *ContextCost*. Same plots as in Figure 3.28.

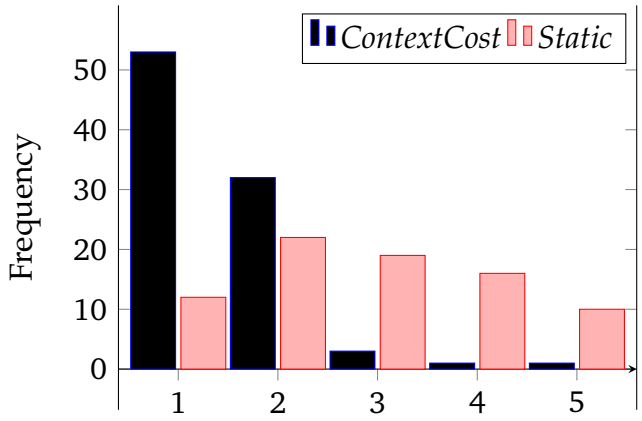


Figure 3.30: Participant ratings of situation: (1=Comfortable, 5=Uncomfortable).

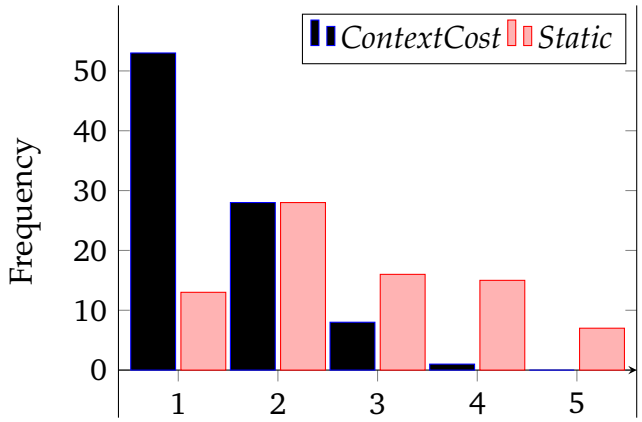
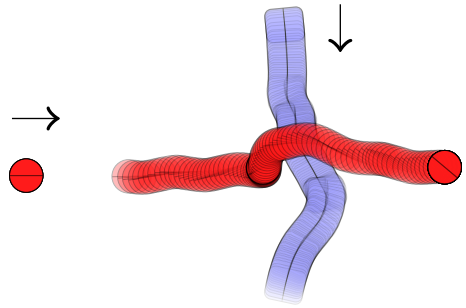
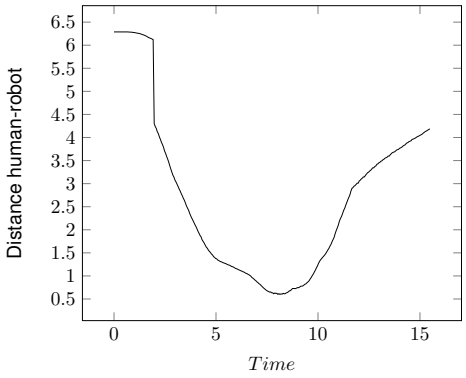


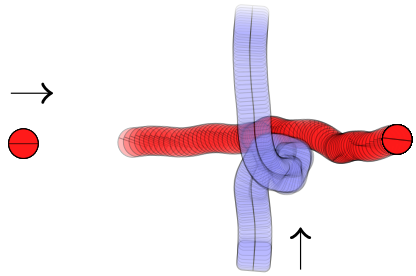
Figure 3.31: Participant ratings of robot behavior: (1=Clear, 5=Strange).



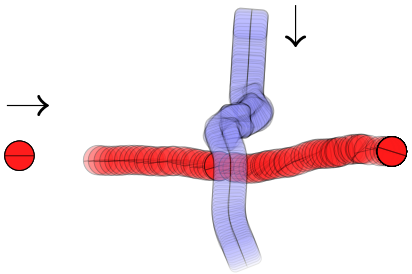
(a) Sample 3, human stops to let robot pass



(b) Distance human to robot center in sample 3



(c) Sample 4, robot (moving bottom to top) makes a small circle



(d) Sample 5, human hesitates multiple times

Figure 3.32: Additional Examples with cost model *Static*, showing variants of robot behavior reacting to different human behaviors

While the figures for strategy *ContextCost* are very similar over all 90 runs, there was a lot of variation over the runs with *Static*. This was due to the different exact circumstances of crossing, which depended on the motion of the participant relative to the robot. To show some of the observed variety, Figure 3.32 shows the paths for additional samples. In Figure 3.32a the participant let the robot move first. This happened 20 out of 80 times with cost model *Static*, but never with *ContextCost*. The minimal distance of 0.5 m was a result of the participant passing closely behind the robot, and such low distances were consistently observed when participants let the robot pass first. Figure 3.32c shows a more extreme robot motion deviation as the robot retraced its step following a path that led away from the human. Finally Figure 3.32d shows a sample where the participant hesitated several times before passing the robot.

No statistically significant differences were found comparing the robot approaching from the right or left-hand side of the human.

The observed human average walking velocities mainly varied between 1.5 m/s and 1.7 m/s with no clear trend detected over trials. While Figures 3.29e and 3.28e appear different, no pattern seemed evident over all 170 runs.

3.4.4 Discussion

The charts in Figure 3.30 and 3.31 still show some degree of variation over the robot clarity of motion and discomfort. This is partially due to the fact that some participants preferred to use the full answer scale while others used only low numbers. However from the remarks we also know that even with the cost model *ContextCost*, there were symptoms the participants disliked. This is mainly due to remaining confusing symptoms in the robot acceleration behavior. As the robot velocity depends on a linear projection of the human motion and the robot, and the sensory data is noisy, the observable robot behavior for slowing down had tiny moments of acceleration, which were very visible to participants and caused mild confusion and discomfort.

Other things learned from doing this user study and the participant remarks: A first attempt at the experiment failed because robot acceleration was uncomfortable, which could have been detected earlier by testing the setup with non-technical people first. Three trial runs for warm-up may not generally be sufficient for this kind of study, for non-technical people it may even be necessary to invite them to sessions on consecutive days to allow for a long adaptation period to the situation as a whole.

Reported discomfort by participants seemed to always be related to uncertainty about future collisions. Participants felt generally disturbed when the robot base moved (or rotated) while the robot appeared to be "waiting for them to pass". Legibility of robot motion may be a quality that only becomes relevant once humans feel sufficiently reassured about their safety in the presence of a robot. Most participants chose a very straight line to their goal, as opposed to curving their path to accommodate the robot, very few participants however

consistently walked on paths curved away from the robot approach direction. Two Participants reported that when the robot base rotated (in the case of cost model *Static*), they felt as if the robot tried to follow them or join them, three participants felt that the robot was trying to cut their way in such cases. Base rotations were reported as uncomfortable when it happened very close to the participants (due to the square shape of the robot, a rotation could also cause a collision with the person's legs), but when it was at least 1.5 m away, no discomfort was reported.

The robot passing first (with cost model *Static* and the participant walking slowly) as in Figure 3.32a was accepted by some participants, others felt the robot should have stopped for them in any case. In such cases it is also notable that human participants pass very close behind the robot, almost touching the robot, so there was no visible notion of social self-distancing from the robot. This could mean that social distancing is generally not required for such situations because the duration where two agents are very close is also very short. However social distancing is likely to still be relevant for situations of following one another, or walking side by side, when the duration of proximity is longer. The observation that humans pass very close behind crossing robots, but do not like being passed close by the same robot in the inverse situation, hints at a difference in socially normative behaviors for humans and robots as also found in [126].

As additional remark on predictability, using the adapted cost model *ContextCost*, the robot behaved very similar for each run, even when participants walked at different speeds or hesitated. This consistency made the robot behavior also very predictable, as consistency is generally a factor increasing predictability. The predictability of the whole situation for third party observers may also be considered, in 20 of 80 runs with *Static*, the participant let the robot go first, but never did so in 90 runs with *ContextCost*, meaning the latter cost model leads to better predictability of the situation as a whole.

The study does not validate this approach for other conflict situations such as crossing at different angles or following at different speeds. Such studies would require a larger experiment area than available this time. Also the study does not compare the straight path behavior of a robot to paths curved "behind" the crossing human, which requires temporal planning to be found by a path plan-

ner. However the findings of our study can still be applied to other planning strategies.

It is also worth considering how the navigation approach scales to situations with more humans. While the robot is prone to the "freezing robot" problem [142], because in case of doubt it reduces velocity to zero, for moderately crowded situations this behavior may still help to avoid making a complex situation even more complex, given the insecurity our participants reported to even slight changes in robot velocity. For densely crowded situations, a fully reactive navigation mode seems preferable.

The study has shown several cues that human observers use to guess the intentions of the robot. While the focus was on the path behavior of the robot, minor accelerations were also used as a significant cue by humans, so those have to be avoided to prevent confusion. Possibly the robot might require a social commitment behavior such that it commits to stopping or moving for at least a given timespan of one or two seconds. As other cues the PR2 caster wheels were making hectic motions when giving certain commands at low velocity, this resulted in sounds that draw attention and suspicion from participants. Similarly in one trial the robot cooling vents went off at full power just when the robot started to move, which visibly startled the participant who then circumvented the robot largely. Such cues may not clearly indicate a specific robot intent, but are confusing as cues.

3.4.5 Conclusions

The user study was performed to validate the improvement of perceived robot motion quality when using an adapted cost model *ContextCost* for path planning. The study shows that using a real robot and uninstructed participants, a robot will show confusing behavior when using a simple proxemic path planner, behavior that humans will rate as strange or confusing. For crossing situations, the study shows that if a robot instead discards proxemic costs in the path planner and chooses a direct line to the goal, but also uses a local planner that adapts velocity to maintain a certain distance, the robot behavior becomes more stable,

and humans rate the robot behavior as clearer. The study also shows that for such situations of conflict, robot acceleration is an important source of confusing robot cues, even if it happens for very short times and at small amounts. The study hints at the possibility that for short durations, a close encounter may not require social distancing at all, but safety distancing. Also the study has revealed that for legibility, it is not only crucial to provide useful cues to the robot internal state (such as its intention), but to also avoid accidentally creating misleading cues. This leads to the notion of "perspective taking" in path planning and local planning, such that a robot needs not only to reason about the effects of a *planned* trajectory, but also about the effects of the *imaginary trajectories* that observers predict for the robot, based on cues. So the implications of this user study on legibility extend to all other motion planning problems involving the presence of humans.

Chapter 4

Contributions to action selection

The contents of this chapter have partly been published in [83].

The previous chapter gave results improving the robot navigation quality by producing more legible behavior. This chapter considers the problem of action selection in the presence of humans.

Reviewing the entry example scenario of a robot serving in a household next to humans (Section 1.1), the action selection problem for a robot named B21 helping to set a table in the presence of human relates to the choices of what items to pick up, put down, and at what time, for the best acceptability by the persons present.

The challenges in human-friendly action selection derive from the same principles as for human-friendly navigation. Humans in the environment must accept the robot, so the robot should avoid behavior that annoys or discomforts the present persons. So the action selection needs to follow social constraints in addition to efficiency and correctness of plans. Also present humans are dynamic and change the environment in unpredictable ways. So if a robot plan requires an available item, a human may at any time make that item unavailable to the robot, but also vice versa. Items that have been unavailable can become available to the robot. Finally there are indirect quality aspects like legibility to consider that are not consequences of individual actions, but of many behavior aspects during the course of one or many actions.

The key concept in this thesis for legible behavior is opportunism. An increase of legibility for robot actions means that its internal state (its beliefs and inten-

tions), can be well-estimated by observing the robot behavior. As for navigation, this does not merely imply producing sufficient signals, but also avoiding misleading behavior. Opportunism can help avoiding misleading behavior.

The word "opportunistic" has positive and negative connotations. As a negative connotation it describes character traits that make an agent behave against conventions of loyalty and commitment. The positive connotation describes a mental flexibility to perceive the potential for optimization and act upon it. For an assistant robot opportunism in the second sense is most appropriate, of course. Opportunism requires thinking ahead several steps (the informal definition of planning), and selection of the best plan according to cost models (or utility models).

In an otherwise static environment that is completely known (the classical environment for a robot), the concept of opportunities has no purpose. Given the complete knowledge of the environment, an optimal plan can be created, and during the execution of the plan, no state can occur that allows for a better plan.

But in environments that are either unpredictably dynamic, or not completely known, opportunities arise during plan execution. Human environments usually have both traits, it is difficult using typical robot sensors to completely know the state of the environment, and acting persons constantly change the environment in an unpredictable way, even by just moving from one place to another.

A different kind of "opportunism" is defined outside the context of achieving given immediate goals. Humans can have very long-term goals, such as traveling to a certain country once in their lifetime, or becoming employed at certain job positions in their career. In that context, an opportunity is an unexpected event that can help fulfill one of those long-term goals. Somewhat similarly, humans can gain emotional benefit out of indulgences like eating cake. In that way, a shop offering a product at a lower price can be described as a rare opportunity to get some benefit at lower costs than usual. However this thesis only considers opportunities in the context of task execution in goal-driven behavior. Within that context, the effort and benefit of achieving goals via a given plan can be estimated by projection, and an alternative plan that has a better cost-to-benefit-ratio marks an opportunity. In the following we consider the problem of order-

ing tasks within a larger plan for opportunistic robot behavior in the presence of humans.

4.1 Opportunistic plan execution

When humans perform tasks, they adapt their behavior to the state of the environment by exploiting opportunities. This means when several steps are required to achieve one or several current goals, humans have a tendency to order steps to reduce overall effort, or to increase expected gain. This is apparent in idioms like "To kill two birds with one stone", meaning two goals are achieved with a single action, which is less effort than spending two actions.

A different way to express this is to consider the concept of "intelligent agents" in general. While many definitions for that concepts exist, consider for now the one offered by Patty Maes: "Autonomous agents are computational systems that inhabit some complex, dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed." [96]. The topic of this thesis is to increase the acceptability of robots in human environments, and doing so by making them realize their goals and task in a more human-friendly way. The method this thesis focuses on is planning. This gives a robot plan-driven behavior, in order to be a goal-directed agent. The plan is a means to achieve complex goals in the absence of a better method. The theory of intelligent agents however requires more than mere plan driven agents. As proposed by Cavendon et al.: "an agent ought to be more committed to his (ultimate) goal than to any specific means" [20]. A given plan is a mean, an opportunistic behavior means giving up this mean for behavior quality, but staying committed to ultimate goals.

Opportunistic behavior of a robot can benefit its acceptance in two ways: (1) the achievement of goals can be expected to often be more efficient (2) the behavior of the robot will reveal the robot internal state in a clearer way and thus allow easier cooperation with humans. This section deals mostly with the second aspect, although an efficiency gain is also possible as a result of opportunistic behavior.

In order to be an acceptable partner for a human, a robot should avoid action sequences that are considered very inefficient by present humans, as a human would be left wondering about the motive of the robot to chose an inefficient option, and have to assume about hidden motives or reasoning errors.

In the context of plan-driven behavior, an opportunity can be described as an alternative action to the robot's current action with an expected superior yield of benefits. When considering the next action to take, the different choices can be considered as different opportunities.

The concept of opportunistic planning relies on the detection of opportunities. In principal, opportunities could be defined by a variety of models, which might depend on the execution context. A model for recognizing opportunities for pick-and-place tasks is the "reachability" of an object as a measure to compare the current desirability of an (alternative) action. Outside HRI, reachability is often a predicate describing that an object is either physically reachable or it is not. It may also describe how much effort for a robot is expected to make a reaching action. Within HRI, instead reachability is a continuous function that can express additionally how much social costs a reaching action would incur, if an object is likely to be physically reachable.

For this section, the criterion of whether an object can easily be reached is determined with respect to the positions of humans in the world. This means that an object, which is spatially near the robot, but which is blocked by a human, is ranked as less reachable than an object to which a free path can be found. This phenomenon is illustrated in Figure 4.1. Although the robot ("Auto") is nearer to place A, the object lying at place B are ranked as better reachable, because the "human" blocks the way to the objects at A. So in the presence of humans, the reachability of an object can be described as a continuous function, and that function definition can include the position or other state of present humans to indicate how acceptable a reaching motion would be.

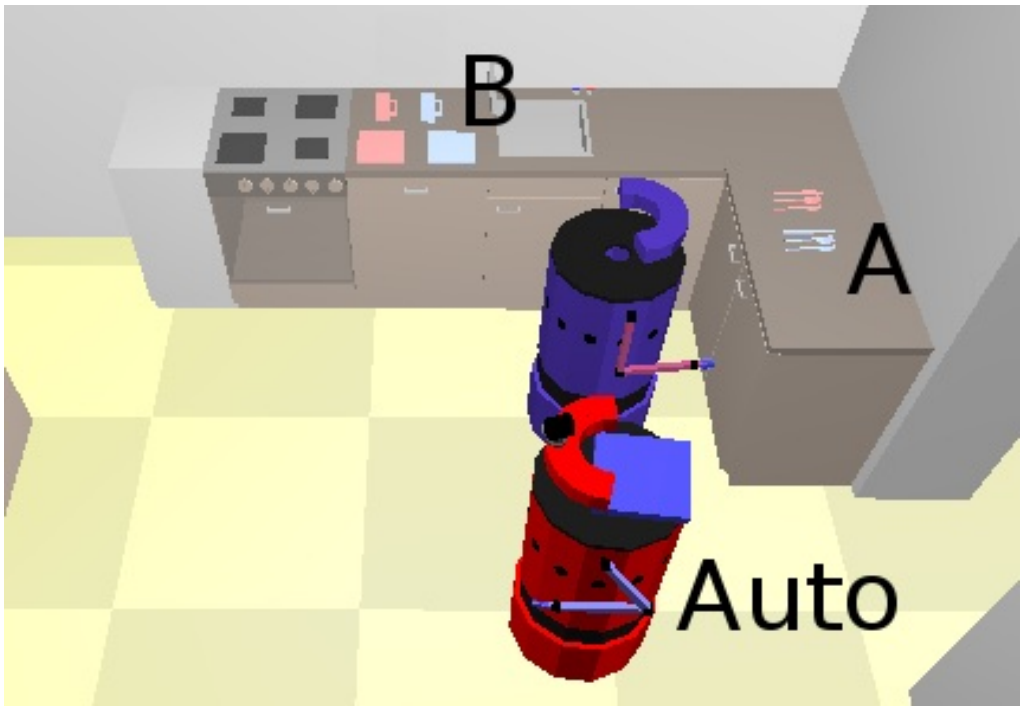


Figure 4.1: Robot "Auto" decides between picking up knife at A or plate at B.

4.1.1 Related Work on opportunistic behavior

A general survey on action selection architectures is given by Brom and Bryson [14].

What can be observed in humans is that often, when faced with a small set of alternatives, humans will make a choice based on decision theory [54]. These are generic models for cognitive agents like humans, animals and robots. Applying decision-theory to planning is discussed by Blythe [11], pointing out the common weakness of modeling real-world domains realistically enough to obtain useful results.

The conflicting needs of planning versus reactivity is present in many dynamic domains. This is shown in the classification by Hayes-Roth [56], who describes a continuous space of agent control modes. Within that space, agent design needs to balance early commitment on actions and sequences with reactivity. Early and persistent commitments break opportunism.

The same argument is made by Schut and Woolridge [131] who describe a framework that allows agents to make commitment choices at runtime rather than at design time, hence they show this approach to outperform agents with fixed policies. They use the well known BDI framework [13, 44] for their agent design, to embed reactive planning as intention reconsideration in a given control loop of the BDI framework. They embed the intention reconsideration into a loop, which in sequence reconsiders actions and then fully executes one action. However robotics and HRI require even more opportunism than that, because during the execution of an action opportunities can also occur, such as a human giving way to a previously blocked position. So additional legibility can be achieved by also recognizing opportunities that occur during the execution of one action. The additional efficiency gains to be expected from that depend on the duration of actions and the frequency of such spontaneous opportunities.

The computation of the utility of actions just before execution is effectively a reactive task planner, whose role is to decide which of the currently possible next actions in the plan should currently be pursued. Hayes-Roth has formulated that this way: "At each point in time, many actions may be logically possible, given the current state of the task environment. An intelligent agent must choose among them, either implicitly or explicitly." [57].

This kind of parallel reconsidering of choices has also been researched in [123] using goodness, competence and conflict as categories for action selection criteria. The robot Milou in that work had a controller which would review alternative navigation plans (what could be considered actions in action selection) every 100ms with regard to three fuzzy measures: goodness, competence and conflict. Goodness is an a priori priority ranking of actions, competence the degree by which the actions are currently possible, and conflict the degree by which one action conflicts with parallel actions. Their robot however has no notion of humans in the environment or of human comfort.

There is also similar work that is less directly related to our work but still worth considering in the context of reactive planning: The general benefits of planning at run-time have led to multiple approaches to achieve an interleaving of planning and execution, e.g. [61] for a truck-world simulation, or in [104] for an office robot.

An early paper on mobile robot software architectures, Gat introduces ATLANTIS as an architecture to deal with contingencies [41]. As a novelty Gat suggests a "sequencer" as a module that controls the robot action by executing tasks, calling a high-level deliberation layer as advice only. This design does not merely embed reactive task planning within a global plan, but it makes reactive task planning the default, with global planning being an alternative for exceptional circumstances.

Michael Georgeff and Felix Ingrand also contributed early on deliberation during execution [43] for the control of spacecraft systems. More recent works by Felix Ingrand extend those ideas [27, 94]. Georgeff on the other hand later contributed to the BDI model for agents [44].

A very recent work on mobile robot autonomy specifically is given by Schermerhorn et al. [129], finding benefits to joint tasks in human-robot teams, however it did not investigate robots acting in the same physical space as humans nor specific cost functions to act human-aware in such spaces, also it did not consider object manipulation tasks.

4.1.2 Formalization of Opportunism

Given changing environments with changing cost estimations of plans, it can be expected that a robot behavior will be more acceptable if the robot is able to change a currently ongoing plan whenever the expected quality of that plan changes.

Opportunities can be defined based on the concept of plans and cost functions estimating the costs a plan will incur.

Definition. *Given a current plan P_0, a cost function for plans and states c an* opportunity *is an environment state S_1 under which a different Plan $P1$ is optimal according to the cost function c:*

$$c(P_1, S_1) < c(P_0, S_1)$$

In other words, S_1 is an opportunity to gain benefits by abandoning P_0 in favor of P_1.

Definition 4.1.2 defines an opportunity as an environment state in which abandoning the current plan for a different one is expected to be beneficial.

A global planner will of course always attempt to create a plan P_0 that will be cost-optimal through all the states the world is expected to be in. So in a static and predictable worlds, opportunities can never arise if a plan was proven to be optimal once.

In unpredictably dynamic environments though, the future states of the world can cause any plan that was expected to be cost optimal to become worse than an alternative plan. This may also just be temporarily the case. As a real world scenario, imagine a robot having a goal of fetching some water. There is a close-by water tab and another tap further away. The close tap however is occupied by a human, who is busy cleaning dishes. Using a cost function that penalizes disturbing the human, a plan P_0 is made for the robot to fetch water from the far water tap. While the robot starts moving to that tap however, the human moves away from the tap, maybe to pick up some item. In this changed state of the world, the originally cost-optimal plan P_0 now has worse cost than an alternative plan P_x that would make the robot use the close tap. Now consider the robot takes a moment to calculate the costs, and during that time the human returns to the dish cleaning. Now plan P_0 is cost-optimal again, and the window of opportunity has passed without the robot exploiting it.

The example shows that in order to effectively exploit opportunities, it is often required to make decisions quickly. The example also shows the dilemma of the motion under uncertainty of opportunities. When a human moves in the environment, should a robot continue to move towards its current goal while evaluating alternatives, or should it stop and evaluate alternatives (because moving can move the robot away from a location of opportunity).

Such aspects are difficult to tweak, the optimal balancing of planning time and action during replanning depends too much on the environment and robot tasks. Ultimately there can be further aspects to consider, such as the likelihood of opportunities arising in the near future, and plans which are suboptimal under current assumptions, but have a larger likelihood to allow optimizations for

likely events in the future. E.g a robot could make a detour to possibly detect a change in the environment that could be an opportunity.

This thesis however only attempts to exploit opportunities that have arisen, and that can be detected without replanning very far into the future. This restriction is required for the robot to detect and exploit opportunities sufficiently early. Investigating complete action plans for alternatives with better expected qualities would often take too long.

4.1.3 Local task planner heuristics

Local planning always implies selecting actions to improve behavior in a small scope without considering the global scope during planning. This means that local planning can always result in globally suboptimal behavior, unless a mechanism exists that prevents this. In navigation, local planning is usually done in the context of a fixed global plan. The global plan then is the mechanism that prevents the robot to get stuck in cul-de-sacs.

For local *task* planning the same principle holds, if actions are selected to be optimal within only a local scope, in the global scope the behavior may become suboptimal. However the benefits of local planning can outweigh the sacrifice of potential global optimality. For robots acting as assistants, there are diverse quality criteria that could be observed. The robot behavior could be efficient, comfortable, legible, predictable, consistent, natural, etc. In an unpredictably changing world, a robot is forced to change plans during execution, so the overall behavior of the robot is not the result of executing a single plan, but of partially executing several plans.

As a simplified model, consider that a robot needs to achieve a goal by performing several atomic tasks in sequence. Before acting, the robot creates a global plan with an optimal action sequence, and after each tasks, if that the environment has changed invalidating the rest of the plan, the robot replans. In this model, the global robot behavior can be described as the sequence of tasks it eventually performed.

So if the robot iteratively created these plans P_0, P_1, P_2, ...

- $P_0 = (A_{0,0}, A_{0,1}, \ldots, A_{0,n_0})$
- $P_1 = (A_{1,0}, A_{1,1}, \ldots, A_{1,n_1})$
- $P_2 = (A_{2,0}, A_{1,1}, \ldots, A_{2,n_1})$
- ...

Assuming for the simplicity that the world changed sufficiently every time the robot acted to invalidate the rest of each plan, the observable robot behavior is $Beh_{global} = (A_{0,0}, A_{1,0}, A_{2,0}, \ldots)$ meaning the robot always executes just the first action of a plan, before switching to the next plan. In HRI, what we want to optimize is the robot acceptability. This means optimizing this observable robot behavior. A first step to optimize this behavior was to generate individual plans that are optimal according to social cost functions, and letting the robot execute those in a static environment, such that $Beh_{global} = P_0$. Given that $Beh_{global} = P_0$, and that P_0 has optimal social costs, Beh_{global} also has optimal social costs. But when deploy robots to dynamic worlds using replanning, the quality of the robot behavior is no more equivalent to the quality of a single plan. Indeed, as the definition shows, while all plans P_i have finite size, the global robot behavior may still never terminate due to oscillation.

In the simple model above, the robot would only replan if the current plan had been invalidated. This means if the world changes sufficiently little or mostly in ways that do not invalidate the plan, a robot may only need to switch plans a few times before achieving the goal. However HRI requires to consider many more qualities than just goal achievement, so replanning may become necessary as soon as a better plan is possible.

The point here is that opportunistic robot behavior is *not* the attempt to change each individual plans P_i such that they provide better robot acceptance, it is an attempt to improve the global behavior Beh_{global} to achieve better robot behavior. As such, the overall quality of any given individual plan P_i does not solely determine the robot behavior quality, but the ability and policy of switching plans and thus generating Beh_{global} also influences robot behavior acceptability.

Predicting the eventual robot behavior Beh_{global} before acting is generally not possible in unpredictably changing worlds. Rating the behavior afterwards is

possible. But it is difficult to determine whether a different ability and policy of changing plans would have produced a better alternative behavior. That's because the events that unpredictably happened (and triggered a change of plans) would be replaced by other unpredictable events if the robot behavior had been different.

So comparing robot plan switching policies in general is a difficult task that would require running a robot in many examples in a realistic setting using two different policies, and statistically determining the effect the policy has on the acceptance of the robot. Doing so would require a lot of effort and the benefits would be questionable, so instead it seems preferable to evolve robot plan switching policies by improving the likelihood of more acceptable plan switches.

The strategy of opportunistic robot behavior by local task planning can be called a "heuristic" in that it possibly sacrifices optimality of individual plans P_i in order to improve the quality of the global robot behavior Beh_{global}.

For the example of setting a table, a fixed plan could specify actions like "Pick up the green cup", "place the green cup", "pick up the green plate" "place the green plate". This plan would achieve the cup and the plate being placed at certain locations, but several other alternative plans could achieve equivalent goals. As an example moving the plate before the cup would not change the end result. But if a person currently occupies the place of the cup, this alternative could be more acceptable. After all, if the person knows the robot has to move both the cup and the plate, the person may feel best if the robot first moves the accessible plate, because the robot has to do so anyways, possibly allowing the person to finish whatever activity they currently perform. Similarly, an alternative plan may be to choose a different cup and plate, if the original goal does not require specific items on the table, just a cup and a plate.

Humans are able to perceive possibilities to switch plans for themselves and the robot, meaning humans regard it as a limitation of a robot if the robot is unable to change a plan or action once it has started executing it. Such a rigid commitment to a plan chosen earlier in time has even worse effect if the robot does not just act in human presence, but works on joint goals with humans. In

the latter kind of scenarios, a robot should be flexible with regards to which goals are achieved by the human and which by the robot, and a robot should be able to dynamically adapt to the human choices. This is of course also a challenge for perception, action recognition and action prediction. But for high-level task planning and execution, this also requires a paradigm shift away from rigid plans to plans that can reactively change.

The purpose of local planning for opportunism is to consider these alternatives within the boundaries set by a larger plan. The boundaries allow the robot to replan at a small cope and thus be reactive, without sacrificing global correctness of actions. This thesis uses the Reactive Plan Language (RPL) [102] for action selection. RPL already defines the concepts of designators to allow late binding of plan symbols to real world entities. This allows a robot to make decisions (like which cup to move) late. "Late" means during the plan of execution when the task "Pick up a cup" needs to be executed. This thesis extends this kind of reactivity to proactively look for alternative actions, even *after* such decisions have been taken, which means abandoning such decisions to provide opportunistic behavior.

4.1.4 Anti-opportunism

When using local task planning as a method for opportunistic robot behavior, it is inevitable that global quality may suffer. That sacrifice is made to allow reactive adaptive behavior. However opportunistic behavior can also in itself reduce robot behavior quality. It may be confusing to an observer that did not expect the robot to be opportunistic about a certain aspect. And opportunistic behavior can lead to oscillations. Both effects are generally possible when replanning during execution. Biological agents may face the same problems if they are opportunistic, and so biologically inspired solutions may be possible. For certain kinds of oscillations, biological agents learn to discard apparent opportunities if several attempts to exploit them fail. As an example, consider a person wanting to move to the other side of a road with some traffic. A pedestrian crossing with traffic light is available some distance ahead, but apparent gaps in the traffic appear as an opportunity to save time by crossing immediately. The person may step onto

the road, then perceive an approaching car, and retract. This may happen several times, until eventually the person may give up and try a different strategy like moving on to the traffic light.

When plan switching and oscillations of a robot appear to reduce acceptability, opportunism can be gradually reduced. Improvements to both effects are possible by adding some cost threshold to any switch of plans. This can be called "inertia", "hysteresis" or "commitment". But defining this threshold in a useful way, depending on the context, and validating that the threshold works well for a given robot task and situation is very difficult.

Another possible approach to negative impacts of plan switching is to do extended reasoning about the context. As an example, if a robot has visibly committed to a specific plan to some observer, it may have a penalty for abandoning the commitment. As an example, imagine a guide robot leading the person wanting to move to the other side of a road in the earlier example. If the robot committed to crossing the road at the traffic light (e.g. by saying so), then it may be best for the robot to ignore the opportunity of moving across the road, while without prior commitment, exploiting the opportunity may be more viable.

Finally reasoning about the nature of an opportunity can be used to reduce negative impacts of plan switching. Where the domain allows it, it may be possible to predict how stable an opportunity can be expected to be, and use this estimate to prevent plan switches in some cases.

Since proving the benefits of any of those approaches in HRI is very difficult, this thesis will not make use of them, they were only mentioned here for the sake of completeness. A formal classification of strategies is given by Cavendal et al. [20] as "precommitment" and "entrenchment".

4.1.5 Approach

This section will investigate how to realize Opportunistic plan-driven behavior in general, outside the context of specific plan execution frameworks. The next section will present a prototype realization based on the RPL plan language.

The general procedure for opportunistic plan-driven behavior is illustrated in Figure 4.2. After acquisition of a goal, a initial global plan is selected or created. While this plan is executed by one process, a parallel process monitors the environment with respect to the plan and goals to detect opportunities. An opportunity is found whenever an alternative plan exists that has a better rating. If an opportunity is found, a second deliberation process ("opportunity filter" in Figure 4.2) has to check whether the switch between the actions is worth the gain predicted by the model to evaluate opportunities. The previous section explained possibilities to prune opportunities. Finally a new plan exploiting the opportunity can be selected, and the plan execution process will be updated with a changed plan. This is constantly repeated until the goal is achieved.

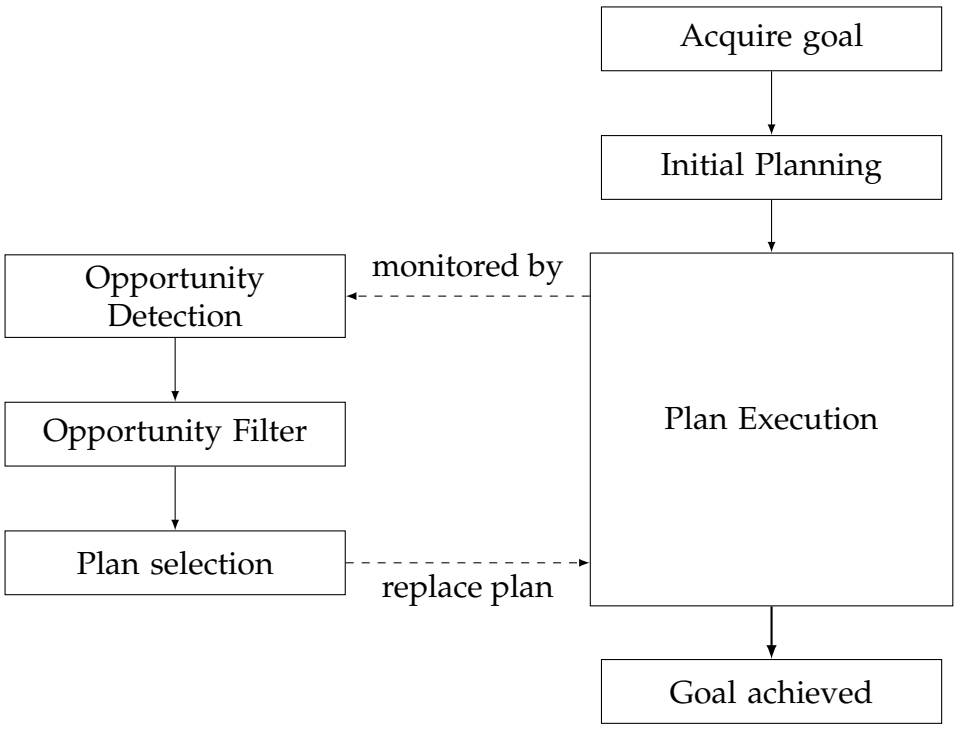


Figure 4.2: General schema for detecting and using opportunities during plan execution. Boxes represent abstract functionalities, which can be implemented as sequential functions or independent processes. Arrows indicate data flow, the dashed arrow a signal. Process flow not specified.

The schema does not define the process flow, because many alternatives can be viable depending on the kind of opportunities to detect and exploit. In particular the separation of opportunity detection, filtering and plan selection may be uni-

fied by a mere replanning step that has additional costs biased towards keeping the same visible intention.

This global schema can be realized by changing the entire plan if an opportunity exists that makes a very different plan more desirable. In many cases of robotics however finding a complete replacement plan that has better expected quality cannot be done in sufficient time to allow exploiting opportunities.

Another way to realize opportunism is to do local task planning as shown in Figure 4.3.

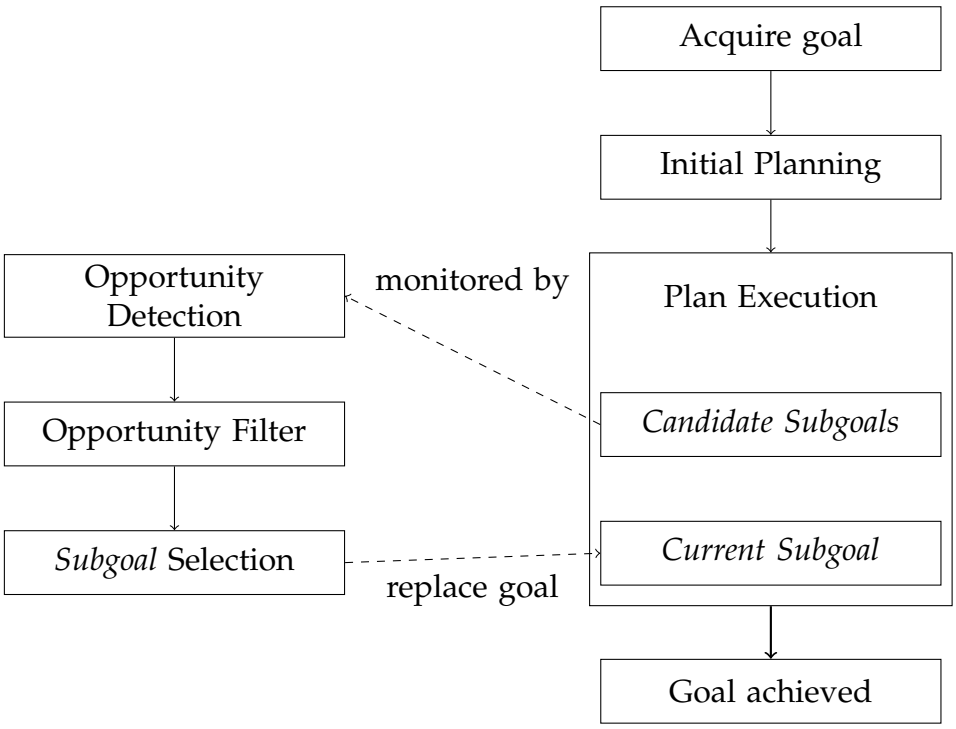


Figure 4.3: Specialized schema for local goal opportunism. Instead of exchanging the whole plan to exploit opportunities, only currently available subgoals can be switched the global plan defines the set of viable subgoals.

In local task planning, the global plan used is not changed in order to use opportunities. Instead, the global plan is constructed in such a way that at suitable times, more than one subgoal may be executed, without affecting the correctness or other relevant global properties of the plan. So an opportunity detection

can focus on evaluating whether switching to a different candidate subgoal has better expected quality or not without having to replan for the global goal.

This is similar to navigation where a local planner can select among several short trajectories to advance on, without affecting the global path and the global qualities granted by the global path.

Even in classical Planning, plans often have representations that allow variance in the execution. A typically partially ordered plan is only partially ordered, meaning several total orderings of the plan graph are valid execution sequences of the plan. So whenever such a plan is executed, there can be situations at runtime where more than one action can next be performed in the partially ordered plan. By considering the effect of any such action a subgoal, the situation shown Figure 4.3 is realized.

The next section looks how this local task opportunism can be used within structured reactive controllers introduced in Section 2.6.

4.1.6 Extending the Reactive plan language

The general task execution framework selected for this thesis are Structured Reactive Plans (SRP) [7] introduced in Section 2.6.

SRPs define plans to satisfy goals. In the plan language a definition of a goal always contains the plan for this goal. The plan for a goal may refer to subgoals to be achieved in modular ways. Subgoal execution can be choreographed using classical or concurrent principles.

The plan language RPL defines constructs seq, partial-order, par, pursue, try-all, try-in-order, try-one. Their properties are listed in Table 4.4.

Using these constructs directly, one may encode various strategies of achieving a parent goal by attempting to achieve subgoals. However, regarding optimal ordering, these constructs do not allow anything else than finding an optimal order of subgoals first, and then using the constructs above to achieve subgoals within that order.

	Concurrency	Fail when ...	Termination when ...
seq	sequential	any fails	all ended
try-in-order	sequential	all failed	one ended
partial-order	special	any fails	all ended
par	parallel	any fails	all ended
pursue	parallel	any fails	one ended
try-all	parallel	all failed	one ended
try-one	pick single goal by random	-	-

Figure 4.4: RPL / CRAM alternatives for choreographing achievement of multiple subgoals

A given plan may use the RPL plan language to reactively schedule subgoal achievement greedily, such as iteratively removing the best subgoal candidate from a set, and achieving this subgoal, and then continue with the next subgoal in the set, until all subgoals have been achieved.

This can already grant some degree of opportunism. However, the problem with this approach is that each subgoal in turn might contain subgoals, and so at a second level of hierarchy, subgoals cannot be changed opportunistically without some centralized scheduling.

So instead the prototype used for this thesis established a API layer between reactive plans and process modules, such that subgoals can be selected and switched opportunistically regardless of their occurrence in the goal tree.

The goal for this thesis is merely to create a prototype to validate the concept of opportunism in HRI, not to produce a feature-complete system to run on any robot. So the prototype is restricted in the scope of opportunism to subgoals of picking up and placing items, for convenience of research. The task primitives are given in a notation that specifies the entity to move and the target location, i.e. the place where it should be put down. This is expressed as an RPL goal $AT(A, B)$ indicating entity A should be a location B. This is similar to the notation used in classical AI for the blocksworld problems, such as $ON(A, B)$ defining that entity A is resting on top of entity B.

As a limitation, the prototype requires that the plan generated is correct and feasible, meaning that if there is a subgoal $AT(A, B)$ anywhere in the plan, then A is an entity the robot can grasp in its current state, and B is a location the robot can place A. Both A and B are uniquely identified. This means in particular that this notation of tasks cannot express that when one item is on top of another, the top one has to be moved before the lower one.

The action primitives available to the robot are picking items and putting them down. Each action primitive implies the robot base moving the robot base to a suitable position first.

In order to opportunistically execute all current subgoals of the robot of type $AT(A, B)$, a coordination layer is added to the design of Structured Reactive Controllers (SRC), as introduced in section 2.6. SRCs only define process modules as executive layer, and reactive plans for all higher level deliberation, in contrast to classical 3-layer architectures. In order to introduce opportunism, a middle layer is required to be added to SRCs, and it is called "sequencer" layer here. The names sequencer was chosen following the names in the ATLANTIS 3-layer architecture as presented in the landmark paper by Gat [42].

> "The three-layer architecture consists of three components: a reactive feedback control mechanism, a reactive plan execution mechanism, and a mechanism for performing time-consuming deliberative computations.[...] In ATLANTIS these layers are called the controller, the sequencer, and the deliberator."

The sequencing module monitors the set of subgoals maintained by higher layers of action selection and at any given time attempts to select a current action to eventually achieve all the subgoals in the set. Should several such actions be possible, it selects the one action which has least costs according to social cost functions.

No assumptions are made about the deliberation layer that adds or removes subgoals to the sequencer being e.g. an actual classical planning algorithm or a mere script. For this thesis structured reactive plans will be used to manage the subgoals, but the sequencer is not aware of that.

The interface between the sequencer and any other processes is based on the definition of primitive goals and actions. The calls by which other processes interact with the sequencer are the following: After being started, the sequencer uses the eventhandler and processmodules to read updates of events, such as a stop signal or the current set of goals.

- eventhandler.stopSignal() : boolean
- eventhandler.readGoals() : primitive-goals
- eventhandler.readReset() : boolean
- monitors.notifyComplete(primitive-goal)
- monitors.notifyFail(primitive-goal, failure)
- processmodules.execute(action)
- processmodules.preemptAction(action)
- processmodules.readFailures(action)
- actionselector.select(primitive-goals, action) : action, primitive-goal

(For uniformity and the purpose of readability, all interactions follow an object-oriented approach in this pseudocode, even if a functional style is applied in the implementation). The implementation in RPL requires a considerable understanding of the underlying RPL language, so the core listings have only been listed in the appendix for completeness sake in Listings 5.1 and 5.2.

Since achievement of goals happens asynchronously, monitors are notified asynchronously of the result of goal achievement. In the case of this thesis, the reactive plan that adds primitive goals will be the only monitor. Goals are achieved by executing actions, the sequencer module has access to a lower layer of control called process modules (equivalent to "controller" layer in ATLANTIS [42])

The sequencer then is implemented as a state machine depicted in Figure 4.5 driven by events such as adding goals, or goals having been achieved.

The execute state of the sequencer is a loop that runs infinitely, and in each iteration selects an action to eventually achieve all goals in its set. This is shown in Listing 4.1.6.

In the Pseudocode, the state machine is implemented by a variable which has one of three constant values, IDLE, EXECUTE and FAILURE.

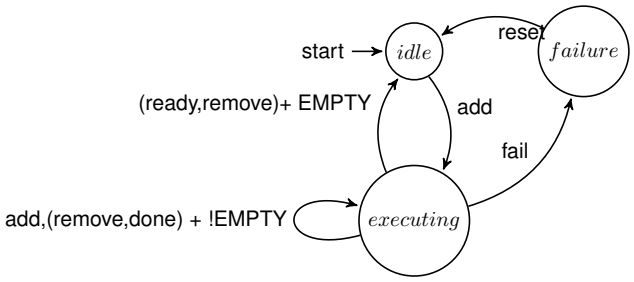


Figure 4.5: State diagram for the sequencer layer, basically the sequencer keeps executing goals until all goals are finished or a failure occurs.

```
1  def sequencer.loop(monitors, processmodules, actionselector, eventhandler):
2    currentAction = nil
3    currentGoal = nil
4    updatedGoals = nil
5    state = IDLE
6    while not eventhandler.stopSignal():
7      # remain in state fail until reset
8      if state == FAILURE:
9        if eventhandler.readReset():
10         state = IDLE
11       else:
12         continue
13     # update received goals
14     updatedGoals = eventhandler.readGoals()
15     if state == IDLE:
16       if updatedGoals != nil:
17         state = EXECUTING
18       else:
19         continue
20
21     ## state is EXECUTING
22     failures = processmodules.readFailures(currentAction)
23     if failures:
24       state = FAILURE:
25       monitors.notifyFail(currentGoal,
26                           failures)
27       continue
28     # check if current goal has succeeded
29     success = checkGoal(currentGoal)
30     if success:
31       monitors.notifyComplete(currentGoal)
```

```
32          updatedGoals.remove(currentGoal)
33
34        if updatedGoals == nil:
35          state = IDLE
36          continue
37
38        # select new action to execute and a goal it serves
39        {newaction, currentGoal} =
40            actionselector.select(updatedGoals, currentAction)
41
42        # if a different action seems more promising, opportunistically switch
43        if newaction != currentAction
44          if currentAction != nil:
45            processmodules.preemptAction(currentAction)
46          # asynchronous call
47          processmodules.execute(newaction)
```

The variables `currentAction` and `currentGoal` hold the information about what the process modules are currently supposed to do. In each loop iteration a possible replacement of the `currentAction` is calculated, using the `actionselector` module.

The `actionselector` module select function takes all open goals and the current situation, and returns an action to perform next, based on models of costs and utility. Key here is that the sequencer may then preempt the running action to perform a different action.

Whenever the situation changes, due to the current robot action or external events, the `actionselector` module may generate a different action. That way the robot becomes adaptive to human even if the environment does not change in a way that means a failure to the plan.

The function `actionselector.select` shown in Listing 4.1 shows such an action generator for a pick and place context. This context allows to reduce the set of actions to "atomic" pick and place tasks, which means that action selection narrows down to the choice of the object to manipulate next. In the general case, the granularity for opportunistic selection is a design choice that affects how

much the robot can exploit opportunity as well as how much computation is required at run-time to compare potential actions.

It generates actions for picking and placing objects based on the state of the robot actuators. While the robot is not carrying a given item and has a free arm, the next possible action is to pick the item up, and when it is holding an item, the next possible action is to put it down at the target destination. While the robot carries multiple items, it generates only two possible next actions of dropping either at its target locations. This can be extended to allow for stacking items, or for other mutually compatible actions.

Listing 4.1: Human-aware action selector

```
1  def actionselector.select(goals, currentAction):
2    actions = nil
3    actionCandidate = nil
4    # generate all possible actions
5    for goal=(At x, y) in goals do:
6      # specific part for pick and place
7      if robot carries (x):
8        add {x, loc(y), goal} to actions
9      else:
10       if robot actuators are free to carry x:
11         add {x, loc(x), goal} to actions
12   # critic, select best action
13   minimalCost = infinity
14   bestAction = nil
15   matchingGoal = nil
16   for action = {item, location, goal} in actions do:
17     cost = calculate expected costs of action
18     if action != currentAction
19       # constant threshold
20       costs += HYSTERESIS
21     if costs < minimalCosts:
22       minimalCost = cost
23       bestAction = action
24       matchingGoal = goal
25   return bestAction, matchingGoal
```

The set of possible actions is then rated, in this case according to a cost function. To implemented prototype uses the path costs of a human-friendly navi-

gation planner to compare the social costs of possible actions. The best action is selected for execution. A hysteresis threshold reduces oscillations. Adjusting the parameter for hysteresis was done manually for each experiment. Various other methods are possible to adjust the parameter, but like many other parameters in HRI, it is difficult to evaluate the usefulness of any method in such a complex setting.

If the set of possible actions grows very large, a complete evaluation of all options can make the robot become less able to seize opportunities in a timely manner. Several approaches can be applied to mitigate this problem, such as applying cheaper heuristics to prune the set of actions early before applying more complex cost function. The importance here is for the robot to exhibit reasonably acceptable opportunistic behavior, such that a human observer will not judge the robot actions unreasonable. The estimation of costs done here does not need to outperform that of a human observer, it needs not be perfect or optimal.

In the prototype social costs for actions are based on path planning alone. This works sufficiently well because the nature of action primitives are relatively comparable, picking up and placing items. Given a broader range of possible actions, a more general model would be required, e.g. comparing the social costs of the arm motion while picking up an item. Many other cost or utility concerns could be used here, the main problem is validating any such social cost calculation for actual benefits when acting near humans. Simple models are useful in research as they can help detect individual factors that improve the robot acceptability.

The prototype uses the modified HANP planner presented in the previous chapter to estimate action costs. As shown in Figure 4.6, HANP plans a path to locations considering social cost functions, but also it returns the costs for this path, which is a weighted sum of social costs and the path length. So by the paths to different locations can serve as a model of action desirability.

Using HANP rather than the Euclidean distance or other simpler heuristics has the benefits of both considering the spatial geometry of the environment (walls and furniture) as well as the comfort of the humans present. Also consider the situation in Figure 4.1, where spot A is closer to the robot "Auto", but obstructed by the other agent. Whether at any given time a special opportunity exists for the

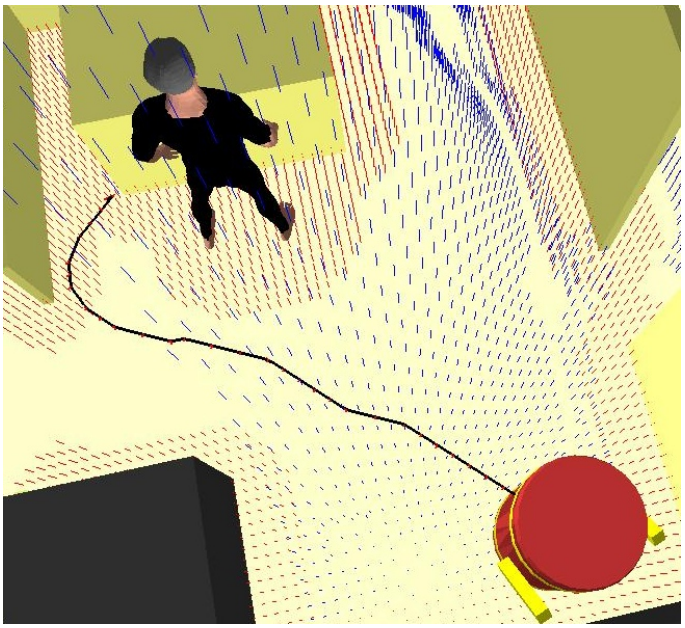


Figure 4.6: Move3d path planning with social costs

robot to save overall time can thus be determined as a function of the minimal navigation costs to move to a location of action.

This can be extended to also allow for e.g. stacking, by generating actions accordingly.

4.2 Validation in with simulated partner

This section shows that given the approach described in the previous sections, a robot shows believably more acceptable behavior in a common household situation. This needs to demonstrate two things: (1) the robot chooses actions suiting the assumed social costs, and (2) the robot is able to adapt to a change of the world by reordering actions, such that the new behavior is more suitable than the previous one.

This approach can be compared to choosing a sequencer which does not apply social cost functions nor change a decision that has been taken.

The demonstration is done in a simulated kitchen environment. It uses the Gazebo simulator [78], which allows defining realistic robot actuators and sen-

sors and uses a rigid body physics engine. The Gazebo simulator visualization window is shown in Figure 4.7. HANP is part of the Move3d software developed at LAAS-CNRS [132], which is shown in Figure 4.6. As agents there a model of the RWI B21 robot with a differential wheel drive, and attached idealized robotic arms. The perception used ground truth for human positions. The prediction of human motion was done by linear interpolations of several positions over time. The experiments were performed on a computer having a Core i7-720QM processor and 4GB of RAM.

The following validation situation is simple but can occur as such often in a joint household task. A rigid situation is chosen to allow for repeatability of the results.

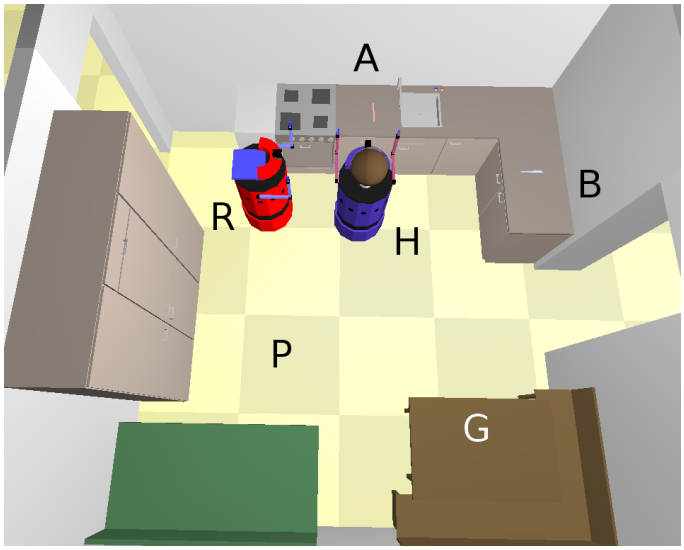


Figure 4.7: Evaluation scenario. Robot *R* has to move knife from *A* and fork from *B* to location *G*. Robot *H* simulates a human which is busy at the current location for an unknown time and then moves away to position *P*.

In the situation shown in Figure 4.7, the robot R is initially idle, and there is another robot H in the environment representing a human. At the beginning of the experiment episode, the robot sequencer is given the following subgoals as part of its plan:

- AT(knife-B, G)
- AT(knife-A, G)

The robot H representing the human runs one of three scripts shown Listing 4.2.

Listing 4.2: Simple behaviors for mocked human, being busy at current location for different times

```
def W5:
   wait 5s
   moveTo(P)

def W30:
   wait 30s
   moveTo(P)

def W80:
   wait 80s
   moveTo(P)
```

The waiting times in the scripts for the human robot were adapted manually to produce meaningful results in the validation.

The robot R would run with one of two reactive plans, one that achieves the subgoals in the given fixed order, and one that used the sequencer described in the last section. The fixed order of tasks was chosen here as if the robot had selected this order based on the reasoning that item B is more available than item A, so that it may be more appropriate to manipulate it first. so opportunistic behavior is not merely compared to fixed behavior, but to any early commitment behavior where decisions, once taken, will only be reconsidered after failures.

For the situation shown in Figure 4.7, observable sequences of actions can be defined based on common sense of what behavior is optimal. While appropriate behavior may differ between contexts and cultural backgrounds, the point here is only to demonstrate the benefits of opportunism itself, without normative claims that the situation should be solved in that way by a robot.

Given that the robot has no knowledge of how long the human will be busy where he is, the robot should as long as possible prefer to execute tasks in different places. Also where possible the robot should avoid unnecessary travel

distances by using both its arms. When the situation changes, the robot should adapt to the new situation.

For case *EARLY*, the optimal sequence of step is then

1. `pick item at A`
2. `pick item at B`
3. `place both at G`

For case *MEDIUM*, the optimal sequence of steps is

1. `pick item at B`
2. `pick item at A`
3. `place both at G`

For case *LATE*, the optimal sequence is

1. `pick item at B`
2. `place item at G`
3. `pick item at A`
4. `place item at G`

The simulation is started with the given inputs for both robots above, and the behavior of the robot *R* is observed.

4.2.1 Result

The table in Figure 4.8 shows the strategy the result of the automatic observation of robot movements. Each row lists the result for 50 runs with a given robot strategy and human behavior. In each column it is counted in how many of the 50 trials for this combination the robot behaved in a specific way. The labels for the columns indicate where the robot stopped, see Figure 4.7.

The failure cases such as when the robot failed to grasp an item were not counted. This could happen as the Gazebo simulator has physics, and so grasping objects always has the risk of virtual slippage.

As specified the robot chose the same strategy in all cases when following the fixed plan, namely to pick and place the item at B first, and then the item at A.

With the adaptive strategy, the robot shows a different behavior. When the human moved early, the robot chose the optimal sequence of Locations A-B-G in all 46 successful cases. This means once the human moved away from location a, the robot chose to pick the item at A rather than to move to location B. Similarly, when the human chose to move after 30 seconds, while the robot was in the process of picking up the item at location B. When the human moved after 80 seconds, the robot usually was already placing the item from location B at G, except in 2 cases, so that in those, the robot adaptively decided to pick the item at location A before moving to G carrying both items.

	B-G-A-G	**B-A-G**	**A-B-G**
S-80	47	0	0
S-30	46	0	0
S-5	43	0	0
A-80	45	2	0
A-30	0	47	0
A-5	0	0	46

Figure 4.8: Observed Robot behavior, classified by where the robot performed pick and place actions

The statistic plots in Figure 4.9 also show that by doing so, the robot saved some travel distance by being adaptive, as expected. In the general case, this might not happen, i.e. it may well happen that by being adaptive, the robot moves a further distance in dynamic situations.

The robot achieved a more efficient behavior with the opportunistic approach, but in this case of course this is a result of the situation being set up for validation. So indeed when the situation is favorable to opportunistic behavior, a sequencer as described in the last section can make the robot behave more acceptably than if it stuck with any early committed sequence of actions.

So the experiment demonstrates that opportunistic behavior creates more believably intelligent behavior in dynamic worlds than any behavior where early commitments are not subject to reconsideration. As opposed to robot behavior that is based on commitment and changing plans only on failures, the robot pro-

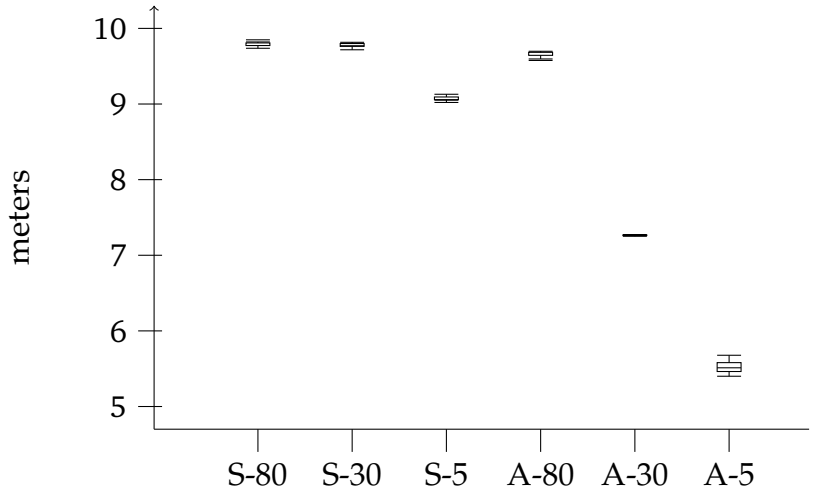


Figure 4.9: Distance traveled by Robot until tasks were achieved. Labels indicate whether action selection was Adaptive or reactive, and how long the simulated human remained in position before moving to location *G*

duces legible behavior by exploiting opportunities. Legibility indicates how well an observer can estimate aspects of internal state of the robot from observation of the robot behavior. In this case the aspect of internal state are the robot knowledge about the environment and its goals. Consider Figure 4.7, if robot R moves to pick up the item at B, an observer can infer that the robot is aware of the item at B, and has a goal concerning the item at B. But an observer cannot know if the robot is also aware of the item at A, and whether the robot has a goal concerning the item at A. If the "human" H in the figure moves away from item A, the space becomes free for the robot r to pick up the item at A. If the robot however continues to item B due to a prior commitment, then an observer may believe that the robot either is not aware of the item at A, or has no current goal concerning A. To avoid such misleading cues in the robot behavior, opportunism makes sure that the robot behavior is not just goal-driven but constantly attempting to optimize the robot behavior for all goals. This constant optimization provides additional cues to the robot internal states. If the "human" H moves away in the situation of Figure 4.7, and robot R stops moving towards B and moves towards A instead, an observer can see from that cue that the robot is also aware of the item at A and has a goal that refers to that item.

In general a robot in the situation given above would have another action option not discussed so far: To communicate its intent to try and make the "human" move away voluntarily. Other alternative approaches are also thinkable, such as using action and intention recognition of human activity to have a suitable estimate of how long the human will be staying in the current position.

Since this validation only shows the behavior that was specified, an additional validation was done with human participants presented in the next section.

4.2.2 Limitations

The prototype used to validate opportunistic behavior has several limitations that make it incomplete as a household assistant.

The Objects manipulated for the experiment were referenced by a unique ID. While the Structured Reactive Controllers Framework allows to use non-rigid designators instead of unique IDs to refer to objects, in joint cooperation with humans, non-rigid designators open a lot of additional problems. As an example, if the robot has the goal of moving any cup to a table, the robot may chose a cup for that purpose. Shortly after that decision, a human might move a different cup to the table. This simple situation is difficult for the robot to interpret, was its task already achieved by the person, or was its task to bring a cup to the table regardless of what items other agents place on the table? So the use of non-rigid designators in HRI very quickly requires common-sense reasoning and natural language processing, possibly even communication.

A similar aspect not well researched here is joint decision making, such as where exactly to place the items on the table when setting the table. While it is not too harmful if the robot chooses other locations for items, the problem itself is complex. For the experiments, we used predefined positions for all dishes, which in HRI could be either default positions or negotiated by the robot via communication.

Figure 4.10: User-study simulation, user perspective, steered robot at bottom. Red and blue items on the kitchen appliance on the left had to be set for breakfast at the table on the right.

Another strong limitation here is that no prediction of human actions was used for deciding on future robot actions. Where such information is available, a robot could in theory behave in even more apparently intelligent ways.

Also the opportunistic framework presented here is greedy only with respect to the currently executed action. This limitation is only a practical one, the concept of local task planning can be extended in other ways to allow locally scheduling actions multiple steps ahead, at the cost of reactivity. It is thinkable that for certain tasks this will provide a noticeable improvement in robot behavior. In the context of the kitchen scenario, an immediate benefit is possible by making use of plate stacking actions, thus allowing a robot to move several items at once. This requires to integrate knowledge about which items can and should be stacked (e.g. plates can be stacked, but clean and used plates commonly belong in separate stacks).

4.3 Validation with human participants

An evaluation of techniques for the purpose of human-robot interaction benefits greatly from actually exposing uninstructed humans to robots. Only then can the inherent physical limitations of robots take effect. And the reactions of human subjects of experiments lead to deeper insights than other forms of evaluation.

So this section describes an attempt made to evaluate opportunistic robot behavior for an assistive scenario. The original intention was to gain empirical evidence that opportunistic robot behavior is rated better than behavior based on commitment. However the problems that surfaced early on during trials made this attempt seem futile with the available resources for this thesis. So instead the setup merely represents a demonstration of opportunistic robot behavior, not a comparison of strategies with respect to acceptability.

Real-world robot prototypes have the disadvantage of being unreliable and slow for complex manipulation tasks. Additionally often the industrial type robotic arms used on research prototypes do not suffice ethical and legal requirements for experiment safety in free settings. This limits strongly the type of experiments that can be performed. As a compromise a 3D simulator was chosen for this thesis, in an experiment with human participants. The simulator offered an interface to a virtual agent to be steered by a human. This is somewhat similar to use a Computer game and augment it with agents controlled by artificial intelligence algorithms, but our approach takes into account the particular constraints of robotics, which are discarded in computer games.

A simulation environment was used where a human can interact with a robot in a household task of setting the table [75]. In the simulation, two robots are able of picking and placing household items in a 3D kitchen environment, one of them would be steered by a human, the other by an autonomous robot controller.

The participants would use the simulator in an office on a 24 in screen, and operate the robot avatar using a standard keyboard and mouse. The human perspective when steering the robot is seen in Figure 4.10, it is somewhat behind the robot as suggested to be optimal in [16].

The maximal movement speed for both robots was set to 0.5 m/s. Grasping and dropping items in the simulator took about 15 s. Those times are constrained by the CPU load for simulation and the controllers of the virtual agents.

Several trials were performed with four different human subjects, male and female students of different disciplines. The participants had the task of setting the virtual table for 2 persons, by placing a plate, cup, knife, for and spoon on the table. The participants could manually navigate the robot model that represents a human. They could also initiate pick and place actions, which were then autonomously executed by the virtual robot. Their robot could pick up an item of cutlery or a cup in each hand, or grab a plate with both hands. Dropping of the items was also automated, the participants merely had to indicate which one of 2 seat positions they wanted to place the item, the robot would then correctly place the plate in a center position, and cutlery and cups in suitable positions as well.

The virtual environment contained a second robot model, driven by a controller that prompted the robot to achieve the same goal of setting the table, thus helping the human with the task. The items for setting the table were visibly laid out on the other side of the virtual room.

The subject's rating of the assisting robot was captured on a scale from 1 to 5, 1 meaning not helpful and 5 meaning very helpful.

The robot behavior was designed to be opportunistic. One of the trials, shown in Figure 4.11 illustrates the opportunistic behavior of the robot. The initial situation that was used was one where the robot R would be closer to items at A than items at B. Immediately after launching the robot controller, the Human H was moved towards the Items A. As expected, the Robot R changed its intention when the human presence to A made this paths more costly than a path to B. The independent looping cycles calculating the costs for all the available intentions (10 in this case) took 2 s.

In the experiments the robot calculated the attractivity of open tasks in 1-3 s, which seemed sufficient to react in suitable time, given the maximum robot velocity. When moving to the target, recalculating the attractiveness in parallel took up to 4 s, as the motion planner was then also used for actual motion plan-

Seconds		Log message
587	1	Executing plan
587	1 2	There are 10 **open** goals to **perform** next
588	1	recalculated costs
588	1 2 3	Performing intention: (PICK THE ENTITY (NAME SPOON-BLUE))
590	1	recalculated costs
591	1	recalculated costs
593	1	recalculated costs
595	1 2 3 4	recalculated costs
597	1	recalculated costs
597	1 2 3 4	A different intention has become more attractive: Item THE ENTITY (NAME CUP-BLUE).
597	1 2	Performing intention: (PICK THE ENTITY (NAME CUP-BLUE))
599	1	recalculated costs

Snapshots

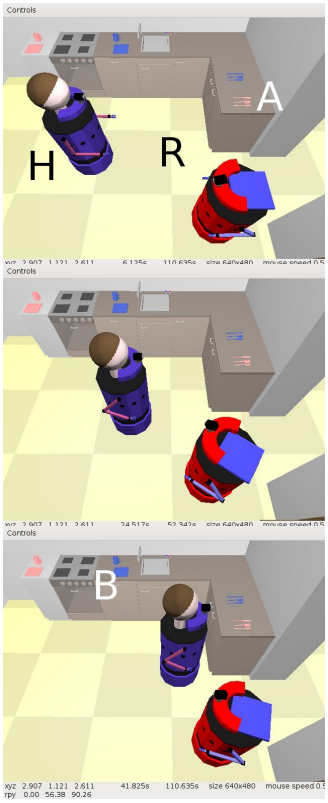


Figure 4.11: Logfile and snapshots showing opportunistic change of intention due to spatial circumstance. At time 588 the robot R decided to next pick item SPOON-BLUE shown as location A, and at time 597, the robot instead decided to pick item CUP-BLUE at location B, EVEN though SPOON-BLUE was still closer given Euclidean distance, but has higher path costs due to the human H having moved.

ning, not just for cost estimation. However the robot continuing to its currently committed goal was not adversely impacted by that calculation in parallel.

Over all trials,the robot moved items 48 times.

The experiment layout proved to be unsuitable to analyze the benefits pre-empting running actions opportunistically, because only in 3 of those 48 moves did the robot change its choice of what item to move next. In most cases when the robot made an initial choice for an item to pick, the human was already busy in a location of the kitchen and did not move during the robot approach of its target, only in 3 instances did the human move and thus made a different item

more attractive. So the experiment layout only used opportunism in the sense of local action selection, not action reconsideration.

The participants could rate the helpfulness of the independent robot on a scale from 1 to 5, and the average rating was 4.2. The subjects were also given the opportunity to give general comments, and the only comments given were about the simulator slowness and of failures due to dropped items, not about the robot behavior. From the comments and the observation of the experiment it became apparent that no meaningful comparison between robot action selection behaviors could be gained other than by forcibly making the robot select bad actions, in effect creating an adversarial situation, which is of no interest to assistive HRI. The participants were too easily frustrated by usability aspects of the simulation, quickly bored with the task, and frustrated over occasional grasping failures of the robot. Their attention was not focused enough on the helper robot to notice differences in action selection. A worse problem was that due to the duration of robot grasping actions, the behaviors of the human and the robot quickly became synchronized in that one would be picking up items while the other was placing items. In such a synchronized joint acting however the opportunistic action selection is very similar to non-opportunistic action selection where the robot selects arbitrary available actions.

So the experiments could not serve as a basis for further statements about the algorithms, but the mere lack of complaints about the robot behavior may serve as an indication of successful cooperative behavior. The subjects did also not seem to be distracted by the fact that no explicit communication was possible between the autonomous robot controller and the human.

The performance of the plans in the scenario varied a lot due to diverse factors, so that even when the robot changed plans, no benefit in terms of efficiency was measured. It was observed in trials that the hysteresis parameter had a beneficial effect on the action selection, when the robot was in a place where two actions in different places had rather similar costs. The hysteresis prevented minor variations causing the robot to switch its intention several times rapidly.

As a consequence the attempt to compare opportunistic robot behavior to other strategies in a user study using the simulator was abandoned. Several lessons

were learned however that may serve future research. A simulator-based evaluation still seems as a useful tool due to the reduced effort in setting up an experiment, and because less infrastructure is required. However the usability of the simulator in the sense of tele-operation of a virtual human needs to be of very high quality to keep the participants engaged. The individual manipulations of items should be as quick as in reality and not require much additional cognitive effort spend on the user interface. The experience of seeing a scene on the screen breaks immersion, and when participants were looking at a scene for manipulating an item, the other robot was not visible. So experiments should provide a wide angle view and allow quick "head motions" to be able to survey the virtual environment and the assistant robot actions. In order to evaluate the benefits of opportunistic behavior, care needs to be taken in designing the task such that action synchronization cannot emerge easily. Avoiding symmetry of tasks can help with that, i.e. the places to visit should not be exactly the same for human and robot, or the action durations should vary significantly.

4.4 Conclusion

This chapter introduced a general framework for detecting and using opportunities in collaborative human-robot tasks. Opportunities were defined as situations during execution of a plan in which a different plan exists that has better estimated costs or utility than the current plan. The framework was implemented with a prototype for opportunistic pick and place tasks in HRI.

When performing pick and place tasks in HRI, a robot has to avoid actions and action sequences that lead to annoying or confusing behavior. Using a human-aware navigation planner, a robot may chose actions that minimize negative effects on humans caused by navigation. This allows the action selection choices of the robot to appear human-friendly. Opportunism then means that the robot adapts his action selection when a human moves unpredictably.

Since humans often unpredictably move in the environment, a robot needs to reconsider his actions. Since global replanning of action selection does not scale well, the chapter suggests local action reconsideration within the context of a correct global plan. This local action reconsideration produces greedy robot be-

havior that may not be globally optimal, but for an environment where global optimality cannot generally be achieved anyway, due to unpredictable dynamics.

The prototype is implemented as an intermediate reasoning module situated between high-level task deliberation and action sequencing. It chooses the next action to take and also preempts the current action when a different action became more attractive in the meantime.

For pick and place tasks, this allowed the robot to interleave actions for different independent tasks, and to exploit having two arms by carrying one item in each hand.

The effect of the opportunistic algorithm was observed in two validation settings, using a 3d robotic simulator with physics and a model of a kitchen environment. In one setting the robot acted in the presence of another agent being controlled by a simple script, to demonstrate how the behavior changes and validate the robustness.

The dynamic action selection worked as predicted and would cause the robot to show the optimal behavior. An alternative scheme that did not allow reconsidering decisions (other than after failure) produced less believably intelligent behavior.

As a second form of validation, a study with human participants was presented, in a realistic joint table setting scenario in simulation, in which the greedy action selection and opportunistic action switching worked as expected.

An observed effect of opportunistic robot behavior is that the robot actions become controllable in a natural way by "stepping in the way of the robot". Thus, when a robot has a goal and several alternative actions available, it may sometimes choose an action alternative that a nearby human stakeholder does not like. That person gains a natural way of coercing the robot to choose an alternative action just by increasing the social costs of the currently chose action, e.g. by stepping in the way. This is possibly a more intuitive method of control than using natural language or other explicit commands, in certain cases.

Chapter 5

Summary

This thesis focuses on challenges specific to planning for HRI, specified in Section 1.4. Those are the requirement to adhere to social conventions, to cope with the unpredictable events inherent to HRI, to react to such events in a timely fashion, and to plan with interaction.

Planning can informally be described as "thinking several steps ahead before acting". Planning can be performed for many aspects of robot behavior, in this thesis navigation and high-level action selection were considered. The thesis shows how the existing planning frameworks for autonomous robotics struggle with providing an infrastructure that satisfies the challenges in HRI. In particularly global planning, that is planning each and every step from the current situation up to an eventual desired state, does not comply well with the challenges. This is on the one hand due to the search space of planning being unmanageable when the world often changes unpredictably, and on the other hand because planning does not scale well, and thus does not permit a robot to react in a timely manner to events.

Planning however cannot be entirely substituted by other techniques such as purely reactive behavior and learning, because goal-driven behavior that adapts to the situation is only generally possible by thinking ahead. So solutions are required to allow plan-driven behavior of robot to adhere to the specific challenges of HRI.

A literature survey of human-friendly navigation research has categorized approaches and solutions. To make robots more human-friendly, most researchers have focused on improving the comfort for humans, making the robot behave

more “natural”, and adhering to specific cultural rules. The approaches in navigation focus on improved prediction of future human motions, path planning, behavior selection, or reactive control of the robot.

For navigation this thesis shows that human-friendly path planning methods that have good results for static humans cannot be transferred to situations with moving humans easily. Mere replanning alone does not produce acceptable behavior in many cases. When temporal planning is not practicable, heuristic approaches can nevertheless improve the robot behavior. In particular by changing the assumption about the human as a fixed obstacle to the assumption of the human as a cooperating intelligent agent, path planning can boldly explore solutions that seemingly go “through” the human, and rely on reactive methods to coordinate joint motion with the human once the robot has started moving and is close to the human. A new cost function has been developed that takes into account moving humans as obstacles only if their predicted path opposes the motion of the robot. For compliant motions, the path planner does not produce a usually unnecessary deviation in the path. Instead, a local planner executing the path adapts the robot velocity. The same strategy has been observed in a human-human experiment for 90° crossing situations. Finally the improved robot behavior has been confirmed in a user study for the same crossing situations. In the user study uninstructed participants were able to notice a difference in robot behavior and rated the new cost function better than the state of the art.

The thesis has applied a theory of legibility and cues to navigation. In that theory, legibility refers to how easily an observer can estimate internal robot states (such as beliefs or intentions) based only on observations of the robot behavior. An observer will always naturally use any cue available to make such estimates, in order to adapt to the robot. As a consequence, a robot needs not only actively produce suitable cues to communicate its internal state, but it also has to avoid accidentally producing conflicting cues.

For navigation the thesis shows how applying mere replanning of static plans to cope with human motion can lead to such undesirable conflicting cues that obfuscate the robot’s intention. A user study has shown this negative effect on acceptability of the robot is measurable empirically.

In action selection similar concepts of legibility apply, and the same challenges of HRI exist, such as the need to quickly adapt to unpredictable events. Since for action selection in this context global replanning is not viable, this thesis instead suggests to apply the concept of local planning to action selection. Using local planning in action selection, opportunistic robot behavior becomes possible. Opportunism is defined as the strategy of an agent to eagerly reconsider an earlier decision, in order to adapt to a change of circumstances for improved expected benefit. So opportunism is an opposite of commitment, which attempts to stick to earlier decisions even under changed circumstances as long as possible.

Opportunism has been implemented as an extension to the existing framework of Structured Reactive Controllers (SRC). By collecting current goals to execute driven by a global plan, and greedily selecting the best goal to execute at any given time, a robot in simulation produced opportunistic behavior in a table-setting scenario with another acting agent. Key to all opportunism is a model describing the expected costs or utility of plans ahead of time. For pick and place tasks in a household, this thesis suggests to use the result of the adapted human-friendly path planning algorithm to judge the current desirability of actions in different places. The costs returned from path planning vary as persons move, so opportunistic behavior is relevant. In this case the opportunistic behavior of a robot does not merely improve the comfort of persons by preferring actions that happen at a distance to humans, but also improves the robot legibility. Legibility is improved because the visible behavior of an opportunistic robot reveals the knowledge and intentions of a robot earlier. Without opportunism, a robot is committed to prior decisions about what action to perform, even if a better action became available in the meantime, so an observer reasoning about the robot under the assumption of intelligence may easily be confused over the lack of opportunism, and generate false expectations about the robot's beliefs and intentions.

The opportunistic behavior was also demonstrated in a user study. While the user study could not serve to quantify the improvement in robot behavior, it nevertheless showed the applicability of opportunism to household pick-and-place tasks.

5.1 Validation of Scenario

In the introduction on this thesis (Section 1.1) an example scenario was given for a robot serving in a household alongside human partners. Enacting this scenario in the real world for this thesis was not feasible, due to technical limitations of current robot hardware, and the sheer engineering effort that would be required to make such a demonstration. Still the experiments done on navigation and action selection in this thesis show how such a scenario could benefit from the approaches suggested in this thesis for HRI.

The scenario described a household scene of a fictional couple Jane and John and a robot named B21. In the scenario, B21 tasks was to assist with chores such as setting the table. This starts with clearing the table of objects, as an example a flower bouquet and a book. Using a Structured Reactive Controller, it is possible to define a plan for setting the table that relies on clearing the table of undesired objects first. The challenge in HRI is for the robot B21 to decide which action to do itself, and how that coordinates with the actions of the present humans Jane and John. This thesis suggests that B21 should follow a plan in which all objects to be removed are listed as subgoals. The robot can then reactively go to the table and pick up items one by one, while present humans join into the activity or not. No explicit communication is required, and when a person gets in the way of the robot, or picks up an item the robot intended to pick up, the opportunistic action selection will simply allow the robot to pick an alternative action, compliantly reacting to the human actions. B21 can also reactively combine actions, such as picking up two items at a time, if they are available and conveniently close to each other.

While moving through the apartment, the robot should also avoid to cause discomfort to the persons present. At the same time, a household being often constrained in space, B21 should not easily block and fail if a person is standing in the way. On the contrary, the robot should rather unambiguously convey its intention, where it wants to go. This implies also avoiding behavior cues that may indicate different intentions. And the even of a person moving at the same time as the robot is not unusual in this situation, and must be taken into account. The adaptations to the human-aware cost functions suggested in this thesis will

enable these properties. B21 can make useful path planning decisions based on the positions of humans, optimistically assuming that persons may move away for the robot, but not making other people move unnecessarily when a better alternative is available. As a result of this thesis it is also apparent that B21 must take care not to accidentally produce turning or acceleration motions that could indicate a different intention than the actual intention of the robot.

The scenario also describes the necessity of being able to reconsider decisions regarding the items to manipulate. While the chapter on action selection did not go into details about it, the general concept of non-rigid designators and local task planning allows to create such behavior. The key as for all planning in HRI is to make decisions quickly, avoiding the effort of global replanning. In the context of a correct global plan describing the constraints for open subgoals, it is possible to quickly replan at a local scope. Thus it becomes feasible to also reason about alternative items to chose to satisfy a goal, or to evaluate costs for n steps ahead, such as considering which pair of items to transport together best, based not only on their current location, but also on their respective target locations.

It is important to consider instantiating a real world situation with an autonomous and human participants similar to the scenario described above. There were several obstacles for achieving this in the context of this thesis. A main obstacle remains the reliable perception of persons and other objects in arbitrary environments with obstacles. The Vicon motion capture system used for the navigation user study requires a large room allowing for a large distance between the sensors and the markers, but at the same time only in a small portion at the center of that room would the markers be tracked reliably. The second main obstacle is robot speed and reactivity within safety limits. this refers both to hardware and software. In hardware, the challenge is for the robot to be able to make swift motions and maintain a relatively high speed, without the mass of the robot and the physical energy created by the motion being a lethal threat to present persons. While compliant robotic arms are being industrially produced for usage close to humans, complete mobile robot bases which suffice safety criteria are not available. And in the absence, research is limited to ensuring safety by either slowing down all robot motion, or strictly separating the operational workspaces of robots and humans, which prevents enacting scenarios as the joint table set-

ting described above. Another big challenge is in interpreting and predicting human motion. All planning activity means thinking several steps ahead, under assumptions of future events. The better model of the future is available to a robot, the more intelligent its planned behavior will appear.

So this thesis could only provide improvements to the planning activities on robots to enable scenarios like the one above, until such scenarios can be enacted reliably, more work on perception, safe and robust manipulation, and reasoning about human activities will be required.

5.2 Conclusion

Human-robot interaction faces many particular challenges in comparison to robotics without human interaction. Many simplifying assumptions about the environment being static or predictable become false when humans are present. With assumptions changing, the architectures of common practices of general robotics become unsuitable in the HRI context. Desirable robot behavior when humans are present cannot be judged in terms of efficiency and correctness, but must be measured in terms of robot acceptance. Predicting acceptance and planning for it requires model of human agreement or rejection of robot behavior. Such models are difficult to obtain in a generalized form, as psychological studies require a rather fixed repetitive structure to allow making inferences from patterns in measured data.

This thesis points out several limitations of state-of-the art planning technologies when used for HRI purposes. The introduction of robotics into the human-centered environments faces obviously many further challenges than those of plan-driven action. But for plan-driven action, this thesis shows that robotic frameworks and architectures will have to put more focus on the dynamical aspects of plan-driven behavior. The assumption that the environment will not change does not hold in HRI. And while machine-learning and behavior based architectures can solve several use cases, plan-based action remains necessary to produce goal-directed behavior in many contexts. So the assumption of an unpredictably changing environment needs to be included into plan-based frame-

works, such that robots can becoming suitable assistants to humans in their homes or living spaces.

The behavior quality cannot merely be defined in terms of simple measures like minimum distances or absence of collisions, but robot behavior quality also requires aspects that are more difficult to measure, such as legibility. Such aspects must be a focus of research on planning in HRI, because planning with simpler cost models is not new to the research area of planning, but does not produce sufficient robot behavior quality.

The paradigm of extended local replanning within a robust global plan seems to be very promising for achieving robot behavior that is both goal-directed and sufficiently reactive in dynamic environments. So in a way, the solution to intelligent robot behavior is to avoid planning most aspects in the long term, but to leave most decisions related to social qualities to a quick planner that only considers a limited scope of time. Global planning is required to avoid behavior that is incorrect or does not lead to the goal, local planning is required to continuously interact with human partners.

The user study on navigation in crossing situations shows how the abstract concept of legibility can be measured empirically. By giving participants a fixed assumption about the robot intention and measuring their confusion over the robot behavior, the legibility of the robot behavior becomes measurable without addition of the concept of predictability. The study also shows that for seamless navigation among humans, a robot needs to avoid small motions that are in themselves harmless, but give rise to fears and doubts in present humans. In the sense of perspective taking, a robot needs not only to think about the actual consequences of motions, but also about the imaginary consequences that observers of the robot may generate.

The benefits of robots helping in households are obvious. To make robots acceptable as household companions, robots have to be intuitively understood by the human stakeholders around them. This does not merely imply actively expressing knowledge and goals, but also suppressing behavior that masks knowledge and goals, and conveys false beliefs about the robot. To achieve this, all research on HRI must assume a vivid interaction with humans that includes plenty

of unexpected events, rather than a basing results on simplifying assumptions of humans being static or predictable like robots. This approach will produce a new generation of robots that can work autonomously in human environments without requiring much supervision effort.

Appendix

A List of Prior Publications

The work presented in this document is partly based on prior publications. Sections of this work that drew upon content from prior publications cited the respective publications where appropriate. A complete list of publications that were (co-)authored during my research as a doctoral candidate is provided below.

Journal Articles

Alexandra Kirsch, Thibault Kruse, E. Akin Sisbot, Rachid Alami, Martin Lawitzky, Dražen Brščić, Sandra Hirche, Patrizia Basili, and Stefan Glasauer. Plan-based Control of Joint Human-Robot Activities. *Künstliche Intelligenz*, 24(3):223–231, 2010.

Patrizia Basili, Murat Saglam, Thibault Kruse, Markus Huber, Alexandra Kirsch, and Stefan Glasauer. Strategies of locomotor collision avoidance. *Gait & Posture*, 37(3): 385–390, 2013.

Thibault Kruse, Amit Kumar Pandey, Rachid Alami, and Alexandra Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726 – 1743, 2013.

Conference and Workshop Papers

Thibault Kruse, Alexandra Kirsch, E. Akin Sisbot, and Rachid Alami. Dynamic Generation and Execution of Human Aware Navigation Plans. In *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010.

Thibault Kruse, Alexandra Kirsch, E. Akin Sisbot, and Rachid Alami. Exploiting Human Cooperation in Human-Centered Robot Navigation. In *RO-MAN, IEEE*, 2010.

Thibault Kruse and Alexandra Kirsch. Towards Opportunistic Action Selection in Human-Robot Cooperation. In *33rd Annual German Conference on Artificial Intelligence (KI 2010)*, 2010.

Thibault Kruse, Patrizia Basili, Stefan Glasauer, and Alexandra Kirsch. Legible robot navigation in the proximity of moving humans. In *Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on*, pages 83–88, 2012.

Thibault Kruse, Harmish Khambhaita, Rachid Alami, and Alexandra Kirsch. Evaluating directional cost models in navigation. In *HRI, ACM/IEEE*, page to appear, 2014.

Other

D. Brščić, M. Eggers, F. Rohrmüller, O. Kourakos, S. Sosnowski, D. Althoff, M. Lawitzky, A. Mörtl, M. Rambow, V. Koropouli, J.R. Medina Hernández, X. Zang, W. Wang, D. Wollherr, K. Kühnlenz, C. Mayer, T. Kruse, A. Kirsch, J. Blume, A. Bannat, T. Rehrl, F. Wallhoff, T. Lorenz, P. Basili, C. Lenz, T. Röder, G. Panin, W. Maier, S. Hirche, M. Buss, M. Beetz, B. Radig, A. Schubö, S. Glasauer, A. Knoll, and E. Steinbach. Multi Joint Action in CoTeSys — Setup and Challenges. Technical Report CoTeSys-TR-10-01, CoTeSys Cluster of Excelence: Technische Universität München & Ludwig-Maximilians-Universität München, Munich, Germany, 2010.

B Supporting Materials

Participant Instructions

This is an experiment about robot navigation. We want the robot to explore different navigation strategies. We use participants who have no knowledge of the robot behavior, to evaluate whether the robot behavior is natural or confusing.

The Robot is a US-manufactured PR2, designed to be safe when acting near humans. There are no sharp edges, and the arms cannot produce strong movements. In the experiments, the robot arms are deactivated, and cannot move.

The Robot can perceive the humans via the Motion Capture System, the system uses 10 ceiling-mounted cameras emitting infrared light and detecting passive markers. You are required to wear the helmet with 4 passive markers on it for the robot to be able to reliably perceive him/her.
The robot motion should be safe, please still do not jump recklessly in front of the robot.

To start a trial, the robot needs to be in the waiting state, on either side of the room. Before the trial, you go to the starting point marked by an X on the floor. Then you wait for the robot to say “Go”, triggered by the experiment instructors. Once you hear “Go”, walk towards the goal without hesitation. When you then walk towards the goal area, the robot will move across your path. This is done to create a crossing situation. The robot's task is to move friendly and naturally. The robot will try different programs to act, so sometimes it will look confused, but this is on purpose. The robot will move to the other side of the room, say “Finished”, and wait for the next trial.

You should go to the goal area at a normal speed and acting as if this were a normal walk in the office. At the goal region, the questionnaire will be waiting on the table, you should give a rating about how surprised and uncomfortable you felt as indicated on the form, after each trial.

You will perform 3 test trials first to become used to the robot and the experiment setup. Those 3 first trials will not be evaluated. Then 8 trials will be performed capturing position data. After that the experiment ends.

The instructors will be present at the experiment but only to stop the robot in case of an emergency. The instructors can clarify these instructions, but shall not answer general questions about the experiment before the end of it.

Thank you.

Figure B.1: Instruction form used in user study on navigation behavior

Instructions pour les participants

Vous allez participer à une expérience sur la navigation autonome de robots dont le but est d'analyser plusieurs stratégies de mouvement interactif. Durant cette étude certaines questions vous serons posées afin de rassembler des avis concernant le comportement du robot.

Le robot est un PR2 fabriqué aux USA. Il a été conçu spécialement pour ne pas être dangereux pour l'homme, notamment il ne possède aucunes parties tranchantes ou pointues et ses bras (qui ne bougeront pas durant l'expérience) ne peuvent causer le moindre mal.

Durant cette expérience, vous serez perçu à l'aide du système de Motion Capture installé dans la salle. Les 10 caméras émettent des rayonnements infrarouges et captent ce qui est reflété par les billes réfléchissantes. Vous allez donc devoir porter le casque, muni des 4 billes, afin que le robot puisse vous voir. Le comportement du robot et son mouvement sont sûrs, cependant, merci d'éviter tout comportement inconsistant comme de sauter devant le robot.

Afin de commencer un essai, le robot doit être en état d'attente. Avant l'essai, placez vous sur le « X » indiqué sur le sol de la salle et attendez que le robot dise « Go », puis allez vers la zone finale, sans hésitation.

Lorsque vous commencerez votre mouvement vers la zone finale, le robot commencera, lui aussi, à se déplacer, croisant ainsi votre chemin. Cette situation est créée intentionnellement, afin que le robot puisse agir de manière polie et naturelle. Le comportement du robot peut varier entre les passages, et pourra avoir l'air confus, mais cela fait parti de l'expérience. Le robot finira par s'arrêter de l'autre côté de la salle, en disant « Finished », puis il se positionnera en attendant du prochain essai.

Vous devez aller à la zone finale à vitesse normale et en agissant comme si vous étiez à la maison par exemple. Après chaque passage, vous devrez répondre à 2 questions concernant ce passage, afin de donner votre avis sur le comportement du robot.

Vous allez effectuer 3 essais de test pour vous habituer au robot et à l'expérience. Ces 3 essais ne seront pas pris en compte pour l'évaluation des données. Ensuite vous ferez 8 passages que vous devrez évaluer comme décrit précédemment. L'expérience pourra alors prendre fin.

Les opérateurs du robots seront présent durant l'expérience, mais uniquement pour agir en cas d'urgence. Les opérateurs peuvent clarifier ces instructions, mais ne répondront à vos questions qu'après l'expérience afin de ne pas introduire de biais.

Merci pour votre participation.

Figure B.2: French Instruction form used in user study on navigation behavior

Human-Robot Interaction Experiment

Participant Form

Personal Details			
Participant ID: _____	Date: _____	Time: _____	
Age: _____	Gender: [ ] Female	[ ] Male	
Education: [ ] Bachelor	[ ] Master	[ ] PhD	
Field of Work: _____			
How much experience you have with robots? _____ years			
How afraid are you with robots? (on a scale of 1 to 10) _____			
Your mother tongue : _____			
Do not film: [ ]	(We will be capturing visual data about the movements of the robot and the participant during the experiment. This data is gathered only for evaluation and presentation purpose and will not be published openly. Check this box if you do not want us to capture the visual data involving you)		

Questions to Participant																				
Trial No.	How **surprised** were you with the robot behavior?										How **uncomfortable** was the crossing situation?									
	not surprised									very surprised	comfortable									uncomfortable
1	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
9	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
11	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10

Participant Remarks on robot or experiment

Figure B.3: Questionnaire used in first user study on navigation behavior

Human-Robot Interaction Experiment

Participant Form

Personal Details		
Participant ID: _____	Date: _____	Time: _____
Age: _____	Gender: [ ] Female	[ ] Male
Education: [ ] Bachelor	[ ] Master	[ ] PhD
Field of Work: _____		
How much experience you have with robots? _____ years		
How afraid are you with robots? (on a scale of 1 to 10) _____		
Your mother tongue : _____		
Do not film: [ ]	(We will be capturing visual data about the movements of the robot and the participant during the experiment. This data is gathered only for evaluation and presentation purpose and will not be published openly. Check this box if you do not want us to capture the visual data involving you)	

Questions to Participant

Trial No.	How **surprised** were you with the robot behavior?										How **uncomfortable** was the crossing situation?									
	not surprised									very surprised	comfortable									uncomfortable
1	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
9	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
11	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10

Participant Remarks on robot or experiment

Figure B.4: French questionnaire used in first user study on navigation behavior

Human-Robot Interaction Experiment

Participant Form

Personal details			
Participant Nr: _____	Date: _____	Time: _____	
Age: _____	Gender: [ ] Female	[ ] Male	
Education: [ ] Bachelor	[ ] Master	[ ] PhD	
How much experience you have with robots? _____ years			
Your mother tongue: _____			
Do not film: [ ]	(We will be capturing visual data about the movements of the robot and the participant during the experiment. This data is gathered only for evaluation and presentation purpose and will not be published openly. Check this box if you do not want us to capture the visual data involving you)		

Questions to participant											
Trail No.		Please rate the robot behavior (clear vs. confusing):						Please rate the crossing situation (comfort) :			
		clear				confusing		comfortable			uncomfortable
1		1	2	3	4	5		1	2	3	4	5
2		1	2	3	4	5		1	2	3	4	5
3		1	2	3	4	5		1	2	3	4	5
4		1	2	3	4	5		1	2	3	4	5
5		1	2	3	4	5		1	2	3	4	5
6		1	2	3	4	5		1	2	3	4	5
7		1	2	3	4	5		1	2	3	4	5
8		1	2	3	4	5		1	2	3	4	5
9		1	2	3	4	5		1	2	3	4	5
10		1	2	3	4	5		1	2	3	4	5
11		1	2	3	4	5		1	2	3	4	5
12		1	2	3	4	5		1	2	3	4	5
13		1	2	3	4	5		1	2	3	4	5

Participant remarks on robot or experiment
(E.g. What distracted you most during the crossing situations ?)

Figure B.5: Questionnaire used in second user study on navigation behavior

Human-Robot Interaction Experiment

Participant Form

Personal details			
Participant Nr: _____	Date: _____	Time: _____	
Age: _____	Gender: [ ] Female	[ ] Male	
Education: [ ] Bachelor	[ ] Master	[ ] PhD	
How much experience you have with robots? _____ years			
Your mother tongue: _____			
Do not film: [ ]	(We will be capturing visual data about the movements of the robot and the participant during the experiment. This data is gathered only for evaluation and presentation purpose and will not be published openly. Check this box if you do not want us to capture the visual data involving you)		

Questions to participant											
Trail No.		Please rate the robot behavior (clear vs. confusing):						Please rate the crossing situation (comfort) :			
		clear				confusing		comfortable			uncomfortable
1		1	2	3	4	5		1	2	3	4	5
2		1	2	3	4	5		1	2	3	4	5
3		1	2	3	4	5		1	2	3	4	5
4		1	2	3	4	5		1	2	3	4	5
5		1	2	3	4	5		1	2	3	4	5
6		1	2	3	4	5		1	2	3	4	5
7		1	2	3	4	5		1	2	3	4	5
8		1	2	3	4	5		1	2	3	4	5
9		1	2	3	4	5		1	2	3	4	5
10		1	2	3	4	5		1	2	3	4	5
11		1	2	3	4	5		1	2	3	4	5
12		1	2	3	4	5		1	2	3	4	5
13		1	2	3	4	5		1	2	3	4	5

Participant remarks on robot or experiment
(E.g. What distracted you most during the crossing situations ?)

Figure B.6: French questionnaire used in second user study on navigation behavior

Participant ID	Age	Male	Education	Language
02	31	True	Master	French
03	26	True	Master	Italian
04	34	True	PhD	English
05	24	False	Master	French
06	32	True	Master	French
08	25	True	PhD	French
09	25	True	Master	French
11	26	True	Master	Portugese
12	23	True	Master	Chinese
13	31	False	Phd	French
14	22	False	Master	Arabic
15	25	True	Phd	English
16	42	True	Master	Italian
17	27	False	Phd	French
19	25	False	Master	English
20	22	True	Master	French
22	37	True	Master	Japanese
23	24	True	Master	French

Figure B.7: Participants for first (failed) attempt at evaluation of cost model *ContextCost*

P-ID	Trial	Surpr.	Disc.	*ContextCost*	Side	minDist	hAvgV
02	4	5	8	True	left	1.52	1.27
	5	3	4	False	right	1.04	1.43
	6	4	5	True	left	1.66	1.58
	7	3	3	True	right	1.48	1.61
	8	4	5	False	left	1.01	1.40
	9	3	3	True	left	1.51	1.18
03	4	2	2	False	right	1.43	1.36
	5	2	2	False	left	1.34	1.48
	6	2	2	True	right	1.33	1.13
	7	2	2	True	right	1.36	1.35
	8	2	2	True	left	1.41	1.42
	9	3	3	False	right	1.37	1.48
	10	2	2	True	left	1.35	1.47

04	4	2	3	False	right	1.00	1.06
	5	2	3	False	left	1.39	1.40
	6	1	4	True	right	1.00	1.23
	7	1	3	True	left	1.22	1.40
	8	1	2	False	right	0.92	1.05
	9	1	3	True	right	1.07	1.35
	10	1	3	True	left	1.09	1.28
05	4	8	1	False	left	1.34	1.34
	5	1	1	True	right	1.66	1.58
	6	6	4	True	left	1.13	1.73
	7	8	4	False	right	1.24	1.52
	8	3	1	True	left	1.42	1.34
	9	6	2	True	left	1.41	1.76
06	4	2	1	True	right	1.30	1.61
	5	2	1	False	left	1.52	1.82
	6	1	1	True	right	1.54	1.80
	7	1	1	False	left	1.75	1.70
	8	1	1	False	right	1.46	1.62
	9	2	2	True	left	1.45	1.81
	10	2	2	True	right	1.22	1.85
	11	1	1	False	left	1.60	1.76
08	4	3	1	True	right	1.18	1.47
	5	3	3	False	left	0.93	1.16
	6	6	4	False	right	1.01	1.31
	7	2	2	True	left	1.22	1.64
	8	1	1	True	right	0.94	1.62
	9	1	1	False	right	1.17	1.39
	10	2	4	True	left	1.14	1.55
	11	1	2	True	right	1.34	1.66

09	4	1	2	True	right	1.14	1.77
	5	2	2	False	left	1.27	1.42
	6	1	1	True	right	1.28	1.75
	7	1	2	True	left	1.18	1.77
	8	1	1	False	right	1.56	1.74
	9	1	2	True	right	1.12	1.77
	10	1	1	False	left	1.38	1.73
	11	1	1	True	right	1.03	1.75
11	4	3	4	False	right	1.18	1.47
	5	3	4	True	left	1.09	1.57
	6	3	3	True	right	1.04	1.65
	7	3	3	False	left	1.21	1.60
	8	3	3	True	right	1.06	1.53
	9	3	3	True	left	1.24	1.53
	10	3	4	False	right	1.49	1.69
	11	3	3	False	left	1.35	1.71
12	4	2	4	True	left	1.22	1.72
	5	2	5	False	right	0.96	1.47
	6	1	2	False	left	1.01	1.20
	7	7	7	True	right	0.90	1.42
	8	5	4	True	left	1.31	1.88
	9	3	2	False	right	1.05	1.59
	10	1	1	False	left	1.50	1.61
	11	3	6	True	right	0.87	1.44
13	4	2	2	False	right	1.04	1.55
	5	1	1	True	left	1.40	1.81
	6	1	1	False	right	1.31	1.62
	7	1	2	False	left	1.09	1.50
	8	1	1	True	right	1.82	1.60
	9	1	1	True	left	1.71	1.51
	10	1	2	False	right	1.20	1.73
	11	1	1	True	left	1.52	1.60

14	4	1	1	True	right	1.19	1.38
	5	1	1	False	left	1.83	1.40
	6	1	1	True	right	1.18	1.49
	7	1	1	True	left	1.35	1.37
	8	1	1	False	right	1.12	1.23
	9	1	1	False	left	1.19	1.40
	10	1	1	True	right	1.29	1.45
	11	1	1	False	left	1.15	1.59
15	4	1	2	True	right	1.15	1.46
	5	2	2	True	right	1.19	1.73
	6	4	5	True	left	0.74	1.48
	7	1	3	False	right	1.03	1.53
	8	1	3	False	left	1.04	1.56
16	4	3	5	False	right	1.03	1.62
	5	4	3	True	left	1.45	1.66
	6	7	7	False	right	0.87	1.46
	7	6	5	True	left	1.17	1.62
	8	3	2	True	right	1.16	1.81
	9	6	6	False	left	1.24	1.73
	10	4	4	False	right	1.32	1.74
	11	6	5	True	left	1.18	1.82
17	4	2	3	False	left	1.36	1.52
	5	1	1	False	right	1.20	1.53
	6	1	1	False	right	1.65	0.48
	7	2	2	True	left	1.27	1.73
	8	1	1	False	left	1.73	1.60
	9	2	2	True	right	1.24	0.68
	10	2	2	True	left	1.55	1.75
19	4	2	3	True	right	1.20	1.67
	5	2	3	True	left	1.33	1.72
	6	3	6	False	right	1.17	1.56
	7	3	5	True	right	0.90	1.61
	8	2	3	True	left	1.36	1.30
	9	4	5	False	right	0.61	0.80
	10	5	6	False	left	1.03	0.80

20	4	2	3	True	right	0.94	1.64
	5	1	1	True	left	1.45	1.50
	6	4	2	False	right	1.35	1.62
	7	1	1	True	left	1.34	1.72
	8	3	3	False	right	1.15	1.27
	9	5	6	False	left	1.20	1.42
	10	2	2	True	right	1.15	1.68
	11	5	7	False	left	1.23	1.38
22	4	3	4	False	right	1.28	1.43
	5	5	7	True	left	0.91	1.49
	6	4	5	False	right	1.13	1.60
	7	3	5	True	left	1.08	1.60
	8	3	6	False	right	1.19	1.47
	9	4	7	True	left	1.13	1.25
	10	4	5	True	right	0.85	1.47
	11	8	9	False	left	1.08	1.61
23	4	2	2	False	right	1.15	1.31
	5	1	1	True	right	1.38	1.66
	6	2	1	True	left	1.41	1.62
	7	1	2	False	right	0.97	1.79
	8	2	2	False	left	1.10	1.64
	9	1	1	True	right	1.23	1.59
	10	2	1	True	left	1.55	1.34

Figure B.8: Participant replies in trials of first (failed) attempt at evaluation of cost model *ContextCost*. P-ID: Participant ID, Surpr.: Reported surprise, Disc.: Reported Discomfort, Side: from which participants die robt approached for crossing, minDist: Minimal Distance between human and robot (center to center), hAvgV: Average human velocity

Participant ID	Age	Male	Education	Language
25	23	True	Master	French
26	30	True	Master	French
27	34	True	Master	French
28	28	True	Master	French
29	27	True	Master	Romanian
31	22	False	Bachelor	Romanian
32	24	True	Master	French
33	27	False	Master	French
34	29	True	Master	French
35	29	True	Phd	Italian
36	28	True	Master	French
37	27	True	Master	German
38	28	True	Phd	Arabic
39	29	True	Bachelor	French
40	25	True	Phd	Arabic
41	26	True	Phd	Chinese
42	27	True	Phd	Indonesian

Figure B.9: Participants for second (successful) attempt at evaluation of cost model *ContextCost*

P-ID	Trial	Surpr.	Disc.	*ContextCost*	Side	minDist	hAvgV
25	4	2	3	False	right	0.72	1.17
	5	1	2	True	left	1.59	1.51
	6	1	1	True	right	1.35	1.66
	7	1	2	False	left	1.37	1.63
	8	1	1	True	right	1.39	1.66
	9	3	3	False	left	0.60	1.05
	10	1	2	False	right	0.77	1.23
	11	2	2	False	left	0.57	0.95
	12	2	1	True	right	1.30	1.65
	13	1	1	True	left	1.39	1.59

26	4	4	4	False	right	0.85	1.60
	5	2	1	True	left	1.43	1.61
	6	2	2	True	right	1.16	1.66
	7	4	4	False	left	1.05	1.65
	8	1	1	True	right	1.18	1.58
	9	3	3	False	left	1.18	1.38
	10	2	4	False	right	0.71	1.41
	11	3	2	False	left	1.26	1.57
	12	1	1	True	right	1.17	1.68
	13	1	1	True	left	1.34	1.72
27	4	5	5	False	right	0.75	1.31
	5	3	1	True	left	1.30	1.53
	6	4	5	False	right	0.58	1.17
	7	2	2	True	left	1.43	1.54
	8	5	5	False	right	0.46	1.07
	9	4	4	False	left	0.88	1.46
	10	4	5	False	right	0.38	1.01
	11	1	2	True	left	1.34	1.49
	12	3	2	True	right	0.82	1.16
	13	2	3	True	left	1.30	1.47
28	4	1	1	True	right	1.30	1.71
	5	4	4	False	left	0.77	1.57
	6	1	2	True	right	1.31	1.70
	7	3	4	False	left	0.80	1.50
	8	5	5	False	right	0.38	1.41
	9	3	3	False	left	0.90	1.18
	10	1	1	True	right	0.91	1.59
	11	1	1	True	left	1.46	1.57
	12	1	1	True	right	1.25	1.59
	13	1	1	True	left	1.45	1.60

29	4	2	2	False	right	0.65	1.80
	5	1	1	True	left	1.60	1.50
	6	2	2	False	right	0.69	0.87
	7	1	1	False	left	1.54	1.72
	8	1	1	True	right	1.13	1.86
	9	1	1	False	left	1.37	1.65
	10	1	3	False	right	0.86	1.30
	11	1	1	True	left	1.54	1.81
	12	1	1	True	right	1.21	1.84
	13	1	1	True	left	1.45	1.79
31	4	2	2	True	right	1.21	1.16
	5	2	1	False	left	1.30	1.27
	6	3	2	False	right	1.01	1.26
	7	2	1	True	left	1.31	1.50
	8	1	1	True	right	0.93	1.16
	9	2	1	False	left	0.81	1.26
	10	3	2	False	right	0.60	1.03
	11	2	2	False	left	1.04	1.32
	12	1	1	True	right	1.30	1.65
	13	1	2	True	left	1.20	0.92
32	4	5	5	False	right	0.59	1.31
	5	2	2	True	left	1.42	1.39
	6	2	3	True	right	1.19	1.72
	7	3	4	False	left	0.66	0.70
	8	2	2	True	right	1.19	1.63
	9	4	4	False	left	1.00	1.55
	10	1	1	False	right	0.80	1.24
	11	1	2	True	left	1.41	1.31
	12	3	3	True	right	1.25	1.70
	13	3	2	True	left	1.37	1.33

33	4	3	3	False	left	0.79	1.57
	5	2	2	True	right	0.46	1.26
	6	2	2	False	left	0.42	1.33
	7	2	2	True	right	0.66	1.21
	8	2	2	False	left	1.58	1.59
	9	3	3	False	right	0.47	0.95
	10	2	2	True	left	1.31	1.07
	11	2	2	True	right	1.08	1.45
	12	2	2	True	right	1.11	1.46
34	4	2	2	False	right	1.01	1.85
	5	1	1	True	left	1.64	1.47
	6	2	1	True	right	1.30	1.49
	7	2	3	False	left	1.35	1.44
	8	2	2	False	right	0.70	1.73
	9	1	1	False	left	1.42	1.79
	10	2	1	True	right	1.40	1.81
	11	1	1	True	left	1.61	1.79
	12	1	1	True	right	1.28	1.87
	13	1	2	True	left	1.50	1.86
35	4	4	4	False	right	0.47	1.15
	5	1	2	False	left	2.00	1.64
	6	2	1	True	right	1.36	1.81
	7	2	1	False	left	1.60	1.72
	8	2	2	True	right	1.34	1.78
	9	2	3	False	left	1.17	1.67
	10	1	4	False	right	0.64	1.73
	11	1	1	True	left	1.56	1.62
	12	1	1	True	right	0.92	1.72
	13	1	1	True	left	1.65	1.67

36	4	1	2	False	right	0.96	1.21
	5	2	3	False	left	1.45	1.55
	6	1	1	True	right	1.36	1.61
	7	2	2	False	left	1.38	1.66
	8	1	1	False	right	0.96	1.70
	9	2	3	False	left	1.31	1.51
	10	1	1	True	right	1.33	1.73
	11	1	1	True	left	1.48	1.55
	12	1	1	True	right	1.27	1.70
	13	1	1	True	left	1.63	1.37
37	4	4	3	False	right	0.61	1.58
	5	4	3	False	left	0.90	1.46
	6	1	2	True	right	1.17	1.70
	7	3	4	True	left	1.40	1.81
	8	4	4	False	right	0.55	1.79
	9	5	3	False	left	1.21	1.66
	10	4	5	False	right	0.83	1.65
	11	3	2	True	left	1.48	1.70
	12	2	2	True	right	1.16	1.72
	13	2	2	True	left	1.45	1.75
38	4	4	4	False	right	1.12	1.69
	5	3	5	False	left	1.18	1.80
	6	4	5	True	right	1.30	1.59
	7	2	2	True	left	1.47	1.49
	8	5	5	False	right	0.81	1.40
	9	5	3	False	left	1.16	1.66
	10	2	4	False	right	0.76	1.27
	11	1	1	True	left	1.61	1.69
	12	1	2	True	right	1.36	1.86
	13	3	2	True	left	1.49	1.75

39	4	2	2	False	right	0.95	1.37
	5	1	1	True	left	1.40	1.69
	6	1	1	True	right	1.37	1.74
	7	2	1	False	left	1.33	1.64
	8	3	4	False	right	0.42	1.15
	9	2	1	False	left	1.08	1.39
	10	1	1	True	right	1.25	1.69
	11	1	1	True	left	1.36	1.47
	12	1	1	True	right	1.13	1.56
	13	1	1	True	left	1.51	1.65
40	4	2	2	False	right	0.82	1.44
	5	1	1	True	left	1.45	1.57
	6	2	2	False	right	0.69	1.22
	7	1	1	True	left	1.43	1.58
	8	1	2	False	right	0.71	1.14
	9	2	2	False	left	1.18	1.11
	10	1	1	True	right	1.11	1.51
	11	1	1	True	left	1.52	1.64
	12	1	1	True	right	1.14	1.59
	13	1	1	True	left	1.41	1.52
41	4	1	1	False	right	0.62	1.45
	5	2	2	True	left	1.30	1.57
	6	1	2	True	right	0.76	1.44
	7	3	3	False	left	0.87	1.53
	8	2	4	False	right	0.87	1.35
	9	2	1	False	left	1.28	1.78
	10	2	5	False	right	0.63	0.96
	11	1	1	True	left	1.24	1.09
	12	1	2	True	right	1.02	1.33
	13	2	2	True	left	1.16	1.00

42	4	4	3	False	right	0.90	1.73
	5	2	2	True	left	1.41	1.29
	6	2	2	True	right	1.42	1.35
	7	4	4	False	left	1.31	1.79
	8	3	3	False	right	0.92	1.41
	9	3	2	False	left	1.23	1.37
	10	3	3	False	right	0.75	1.80
	11	2	1	True	left	1.43	1.79
	12	2	1	True	right	1.36	1.79
	13	3	1	True	left	1.42	1.48

Figure B.10: Participant replies in trials of evaluation of cost model *ContextCost*. P-ID: Participant ID, Surpr.: Reported surprise, Disc.: Reported Discomfort, Side: from which participants die robt approached for crossing, minDist: Minimal Distance between human and robot (center to center), hAvgV: Average human velocity

Listing 5.1: LISP code to generate action for a pick-and-place goal with cost functions

```
(defmethod find-actions-for-goal ((goal pick-place-goal) belief-context)
  "returns a pick-place-action helping for this goal, considering belief state"
  (let*  ((left-hand-entity
             (robot-beliefs-left-hand-entity (belief-context-robot-beliefs belief-context)))
          (right-hand-entity
            (robot-beliefs-right-hand-entity (belief-context-robot-beliefs belief-context)))
          (holds-item-directly
           (robot-holds-entity-p (pick-place-goal-entity goal)))
         (has-one-free-hand (or (null left-hand-entity) (null right-hand-entity)))
         (has-two-free-hands (and (null left-hand-entity) (null right-hand-entity))))
    (unless (or (value (pick-place-goal-entity-unavailable-fluent goal))
                (value (pick-place-goal-success-fluent goal))
                (null (pick-place-goal-entity goal)))
      (cond
        ;; holds item -> PLACE
        (holds-item-directly
         (make-pick-place-action
          :plan (make-instance 'entity-on-entity
                               :top-entity  (pick-place-goal-entity goal)
                               :bottom-entity (pick-place-goal-chosen-bottom-entity goal)
                               :rel-location (pick-place-goal-relative-location goal))
          :name (list 'place (pick-place-goal-entity goal))
          :pose (perform-action-pose
                  (copy-entity-and-replace-pose
                    (pick-place-goal-entity goal)
                    (pick-place-goal-chosen-target-location goal)))
          :goal-success-fluents (pick-place-goal-success-fluent goal)
          :goals (list goal)
          :cost-map '(navigation-penalty)))
        ;; when action is possible and we do not hold entity yet
        ((or has-two-free-hands
             (and has-one-free-hand (not (eq 'plate
                 (value (entity-type (value (reference (pick-place-goal-entity goal)))))))))
         (make-pick-place-action
          :plan (make-instance 'entity-on-entity
                               :top-entity  (pick-place-goal-entity goal)
                               :bottom-entity (pick-place-goal-chosen-bottom-entity goal)
                               :rel-location (pick-place-goal-relative-location goal))
          ;;(make-instance 'entity-picked-up :entity (pick-place-goal-entity goal))
          :name (list 'pick (pick-place-goal-entity goal))
          :pose (search-perform-action-pose (value (reference (pick-place-goal-entity goal))))
          :goal-success-fluents (pick-place-goal-success-fluent goal)
          :goals (list goal)
          :blocked-fun (lambda () (setf (pick-place-goal-entity goal) nil))
          :cost-map '(hand-full-pickup-penalty navigation-penalty)))
        (T nil)))))
```

Listing 5.2: RPL code to continuously and opportunistically select and execute actions for a current set of subgoals that can concurrently change.

```
(def-interp-proc achieve-opp-goals ()
  (let* ((goals t)
         desires
         current-intention
         (finished nil))
    (with-failure-handling failure ()
      (recover
       ((typep failure 'intention-blocked-failure)
        (achieve (make-instance 'b21-at-pose :pose (value (getgv 'statevar 'b21-pose))))
        :retry)
       ((typep failure 'better-opportunity-failure)
        (achieve (make-instance 'b21-at-pose :pose (value (getgv 'statevar 'b21-pose))))
        :retry)
       (T
        (achieve (make-instance 'b21-at-pose :pose (value (getgv 'statevar 'b21-pose))))
        (fail failure)))
      (monitor
       (seq
        (nisp::loop
          until (null goals)
          (seq
            (wait-time 1)
            (nisp::loop
              until (null (slot-value (getgv 'global-valve 'b21-arms-busy-valve)
                                      'nisp::owner))
              (wait-time 1))
            (setf goals (getgv :reactive-tasks :open-reactive-goals ))
            (setf desires (find-intentions-for-goals goals))
            (let (new-intention)
              (setf new-intention (execute-in-parallel
                                   'select-kibo-intention
                                   goals
                                   current-intention))
              (unless (or (null new-intention)
                (intentions-similar-p current-intention new-intention)
                (slot-value (getgv 'global-valve 'b21-arms-busy-valve) 'nisp::owner))
                (setf current-intention new-intention)
                (fail :class better-opportunity-failure))
              (wait-time 1))))
        (setf finished t)))
      (perform
       (seq
        (nisp::loop
             until finished
                   (if (null current-intention)
                       (seq
                        (wait-time 1))
                       (seq
                        (achieve (pick-place-intention-plan current-intention))
                        (setf current-intention nil)))))))))
```

Bibliography

[1] S. Alili, R. Alami, and V. Montreuil. A task planner for an autonomous social robot. In *Distributed Autonomous Robotic Systems 8*, pages 335–344. Springer Berlin Heidelberg, 2009.

[2] P. Althaus, H. Ishiguro, T. Kanda, T. Miyashita, and H. Christensen. Navigation for human-robot interaction tasks. In *ICRA, IEEE*, 2004.

[3] D. Althoff, D. Wollherr, and M. Buss. Safety assessment of trajectories for navigation in uncertain and dynamic environments. In *ICRA, IEEE*, 2011.

[4] T. Asfour, K. Regenstein, P. Azad, J. Schroder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann. Armar-iii: An integrated humanoid platform for sensory-motor control. In *Proceedings of the IEEERAS/RSJ International Conference on Humanoid Robots (Humanoids06)*, pages 169–175, 2006.

[5] P. Basili, M. Huber, T. Brandt, S. Hirche, and S. Glasauer. Investigating human-human approach and hand-over. *Human Centered Robot Systems: Cognition, Interaction, Technology*, pages 151–160, 2009.

[6] P. Basili, M. Saglam, T. Kruse, M. Huber, A. Kirsch, and S. Glasauer. Strategies of locomotor collision avoidance. *Gait & Posture*, 37(3):385–390, 2013.

[7] M. Beetz. Structured Reactive Controllers — a computational model of everyday activity. In O. Etzioni, J. Müller, and J. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents*, pages 228–235, 1999.

[8] M. Beetz, D. Jain, L. Mösenlechner, and M. Tenorth. Towards Performing Everyday Manipulation Activities. *Robotics and Autonomous Systems*, 58(9):1085–1095, 2010.

[9] M. Beetz, D. Jain, L. Mosenlechner, M. Tenorth, L. Kunze, N. Blodow, and D. Pangercic. Cognition-enabled autonomous robot control for the realization of

home chore task intelligence. *Proceedings of the IEEE*, 100(8):2454–2471, 2012.

[10] M. Beetz, L. Mösenlechner, and M. Tenorth. CRAM – A Cognitive Robot Abstract Machine for Everyday Manipulation in Human Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1012–1017, Taipei, Taiwan, October 18-22 2010.

[11] J. Blythe. Decision-theoretic planning. *AI Magazine*, 20, 1999.

[12] C. Boutilier, T. L. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *J. Artif. Intell. Res. (JAIR)*, 11:1–94, 1999.

[13] M. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press, Cambridge, Massachusetts, 1987.

[14] C. Brom and J. Bryson. Action selection for intelligent systems. *European Network for the Advancement of Artificial Cognitive Systems*, 2006.

[15] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(10), 1986.

[16] D. J. Bruemmer, D. A. Few, and C. W. Nielsen. Spatial reasoning for human-robot teams. In B. N. Hilton, editor, *Emerging Spatial Information Systems And Applications*, pages 350–372, Hershey, PA, USA, 2006. IGI Publishing.

[17] J. T. Butler and A. Agah. Psychological effects of behavior patterns of a mobile personal robot. *I. J. Autonomous Robots*, 10(2):185–202, 2001.

[18] S. Cambon, F. Gravot, and R. Alami. asymov: Towards more realistic robot plans. In *International Conference on Automated Planning and Scheduling, (ICAPS 2004)*, 2004.

[19] D. Carton, A. Turnwald, D. Wollherr, and M. Buss. Proactively approaching pedestrians with an autonomous mobile robot in urban environments. In *13th International Symposium on Experimental Robotics (ISER)*, 2012.

[20] L. Cavedon, A. S. Rao, and W. Wobcke, editors. *Intelligent Agent Systems, Theoretical and Practical Issues, Based on a Workshop Held at PRICAI'96, Cairns, Australia, August 26-30, 1996*, volume 1209. Springer, 1997.

[21] N. Chan, J. Kuffner, and M. Zucker. Improved motion planning speed and safety using regions of inevitable collision. In *17th CISM-IFToMM Symposium on Robot Design, Dynamics, and Control (RoManSy'08)*, July 2008.

[22] S.-Y. Chung and H.-P. Huang. Incremental learning of human social behaviors with feature-based spatial effects. In *IROS, IEEE/RSJ*, 2012.

[23] W. Chung, S. Kim, M. Choi, J. Choi, H. Kim, C. bae Moon, and J.-B. Song. Safe navigation of a mobile robot considering visibility of environment. *Industrial Electronics, IEEE Transactions on*, 56(10):3941 –3950, October 2009.

[24] R. B. Cialdini and N. J. Goldstein. Social influence: Compliance and conformity. *Annual Review of Psychology*, 55(1):591–621, Feb. 2004.

[25] A. Clodic, S. Fleury, R. Alami, R. Chatila, G. Bailly, L. Brèthes, M. Cottret, P. Danès, X. Dollat, F. Eliseï, I. Ferrané, M. Herrb, G. Infantes, C. Lemaire, F. Lerasle, J. Manhes, P. Marcoul, P. Menezes, and U. P. Sabatier. Rackham: An interactive robot-guide. In *RO-MAN, IEEE*, 2006.

[26] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, C. L. Nehaniv, E. A. Sisbot, R. Alami, and T. Siméon. How may i serve you?: a robot companion approaching a seated person in a helping context. In *HRI, ACM/IEEE*, Utah, USA, 2006.

[27] O. Despouys and F. Ingrand. Propice-plan: Toward a unified framework for planning and execution. *Recent Advances in AI Planning*, 2000.

[28] A. D. Dragan, K. C. Lee, and S. S. Srinivasa. Legibility and predictability of robot motion. In *Proceedings of the 8th ACM/IEEE International Conference on Human-robot Interaction*, HRI '13, pages 301–308, Piscataway, NJ, USA, 2013. IEEE Press.

[29] B. Duffy. Anthropomorphism and the social robot. *Robotics and Autonomous Systems*, 42(3-4):177–190, 2003.

[30] K. Erol, J. Hendler, and D. Nau. Htn planning: Complexity and expressivity. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1123–1123. John Wiley & Sons LTD, 1994.

[31] D. Feil-Seifer and M. J. Matarić. Human robot interaction. In R. A. Meyers, editor, *Encyclopedia of Complexity and Systems Science*, pages 4643–4659. Springer New York, 2009.

[32] D. J. Feil-Seifer and M. J. Matarić. People-aware navigation for goal-oriented behavior involving a human partner. In *Proceedings of the International Conference on Development and Learning*, Frankfurt am Main, Germany, August 2011.

[33] R. O. Fikes and N. J. Nilsson. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Technical Report 43r, AI Center, SRI International, 1971.

[34] A. Foka and P. Trahanias. Predictive autonomous robot navigation. In *IROS, IEEE/RSJ*, 2002.

[35] T. Fong, I. R. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42, 2003.

[36] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 1997.

[37] M. Fox and D. Long. PDDL2.1: An extension of PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:61–124, 2003.

[38] T. Fraichard. A short paper about motion safety, 2007.

[39] C. Fulgenzi. *Autonomous navigation in dynamic uncertain environment using probabilistic models of perception and collision risk prediction.* PhD thesis, Institut National Polytechnique de Grenoble - INPG, June 2009.

[40] O. Gal, Z. Shiller, and E. Rimon. Efficient and safe on-line motion planning in dynamic environments. In *ICRA, IEEE*, 2009.

[41] E. Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, San Jose, CA, 1992.

[42] E. Gat. On Three-Layer Architectures. In P. Bonasso, D. Kortenkamp, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots*, pages 195–210. MIT Press, Cambridge, MA, 1998.

[43] M. Georgeff and F. Ingrand. Real-time reasoning: the monitoring and control of spacecraft systems. AI center technical note SRI AI 478, SRI International, Jan. 1990.

[44] M. P. Georgeff, B. Pell, M. E. Pollack, M. Tambe, and M. Wooldridge. The belief-desire-intention model of agency. In *ATAL*, pages 1–10, 1998.

[45] R. Gockley, J. Forlizzi, and R. Simmons. Natural person-following behavior for social robots. In *HRI, ACM/IEEE*, 2007.

[46] A. C. Gonzalez, M. Shiomi, T. Kanda, M. A. Salichs, H. Ishiguro, and N. Hagita. Position prediction in crossing behaviors. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010*, 2010.

[47] J. P. Gonzalez, A. Dornbush, and M. Likhachev. Using state dominance for path planning in dynamic environments with moving obstacles. In *ICRA, IEEE*, 2012.

[48] M. A. Goodrich and A. C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

[49] C. Granata and P. Bidaud. A framework for the design of person following behaviours for social mobile robots. In *Intelligent Robots and Systems (IROS)*, pages 4652–4659, 2012.

[50] F. Gravot, S. Cambon, and R. Alami. asymov: A planner that deals with intricate symbolic and geometric problems. In *Proceedings of the 11th International Symposium on Robotics Research (ISRR)*, pages 100–110, 2003.

[51] H.-M. Gross, H. Boehme, C. Schröter, S. Müller, A. Koenig, E. Einhorn, C. Martin, M. Merten, and A. Bley. Toomas: Interactive shopping guide robots in everyday use - final implementation and experiences from long-term field trials. In *IROS, IEEE/RSJ*, 2009.

[52] E. Hall. *The hidden dimension*. Anchor Books, 1966.

[53] S. T. Hansen, M. Svenstrup, H. J. Andersen, and T. Bak. Adaptive human aware navigation based on motion pattern analysis. In *Robot and Human Interactive Communication, 2009.*, Toyama, Japan, September-October 2009.

[54] S. O. Hansson. Decision theory: A brief introduction, 2005.

[55] K. Hayashi, M. Shiomi, T. Kanda, and N. Hagita. Friendly patrolling: A model of natural encounters. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[56] B. Hayes-Roth. Opportunistic control of actions in intelligent agents, 1992.

[57] B. Hayes-Roth. Intelligent control. *Artificial Intelligence*, 59:213–220, 1993.

[58] M. Heerink, B. J. A. Kröse, V. Evers, and B. J. Wielinga. Assessing acceptance of assistive social agent technology by older adults: the almere model. *I. J. Social Robotics*, 2(4):361–375, 2010.

[59] F. Hegel, S. Gieselmann, A. Peters, P. Holthaus, and B. Wrede. Towards a typology of meaningful signals and cues in social robotics. In *RO-MAN*, pages 72–78, 2011.

[60] D. Helbing. A mathematical model for the behavior of pedestrians. *BEHAVIORAL SCIENCE*, 36:298–310, 1991.

[61] J. Helwig and P. Haddawy. An abstraction-based approach to interleaving planning and execution in partially-observable domains. In *AAAI Technical Report FS-96-01*, 1996.

[62] P. Henry, C. Vollmer, B. Ferris, and D. Fox. Learning to navigate through crowded environments. In *ICRA, IEEE*, 2010.

[63] F. Hoeller, D. Schulz, M. Moors, and F. E. Schneider. Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps. In *IROS, IEEE/RSJ*, 2007.

[64] K.-C. Huang, J.-Y. Li, and L.-C. Fu. Human-oriented navigation for service providing in home environment. In *SICE Annual Conference 2010, Proceedings of*, August 2010.

[65] T. Imai. Interaction of the body, head, and eyes during walking and turning. *Experimental Brain Research*, 136(1):1–18, 2001.

[66] B. Jensen, N. Tomatis, L. Mayor, A. Drygajlo, and R. Siegwart. Robots meet humans – interaction in public spaces. *IEEE Transactions on Industrial Electronics*, 52(6), December 2005.

[67] E.-J. Jung, B.-J. Yi, and S. Yuta. Control algorithms for a mobile robot tracking a human in front. In *IROS, IEEE/RSJ*, 2012.

[68] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot hrp-2. In *Proceedings of the 2004 IEEE International*

Conference on Robotics and Automation, 2004.

[69] S. Kean, J. Hall, and P. Perry. *Meet the Kinect: An Introduction to Programming Natural User Interfaces*. Apress, Berkely, CA, USA, 1st edition, 2011.

[70] J. E. Kelley, Jr and M. R. Walker. Critical-path planning and scheduling. In *Papers Presented at the December 1-3, 1959, Eastern Joint IRE-AIEE-ACM Computer Conference*, IRE-AIEE-ACM '59 (Eastern), pages 160–173, New York, NY, USA, 1959. ACM.

[71] J. Kessler, C. Schröter, and H.-M. Gross. Approaching a person in a socially acceptable manner using a fast marching planner. In *ICIRA (2)*, pages 368–377, 2011.

[72] S. Kim, W. Chung, C. bae Moon, and J.-B. Song. Safe navigation of a mobile robot using the visibility information. In *ICRA, IEEE*, 2007.

[73] R. Kirby, R. Simmons, and J. Forlizzi. Companion: A constraint optimizing method for person-acceptable navigation. In *RO-MAN, IEEE*, 2009.

[74] A. Kirsch. *Integration of Programming and Learning in a Control Language for Autonomous Robots Performing Everyday Activities*. PhD thesis, Technische Universität München, 2008.

[75] A. Kirsch and Y. Chen. A testbed for adaptive human-robot collaboration. In *33rd Annual German Conference on Artificial Intelligence (KI 2010)*, 2010.

[76] B. Kluge and E. Prassler. Reflective navigation: Individual behaviors and group behaviors. In *ICRA*, pages 4172–4177, 2004.

[77] 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.

[78] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *In IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2149–2154, 2004.

[79] S. Kripke. *Naming and Necessity*. Basil Blackwell, Oxford, 1980.

[80] K. M. Krishna, R. Alami, and T. Siméon. Safe proactive plans and their execution. *Robotics and Autonomous Systems*, 54(3):244–255, 2006.

[81] T. Kruse, P. Basili, S. Glasauer, and A. Kirsch. Legible robot navigation in the proximity of moving humans. In *Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on*, pages 83–88, May 2012.

[82] T. Kruse, H. Khambhaita, R. Alami, and A. Kirsch. Evaluating directional cost models in navigation. In *HRI, ACM/IEEE*, page to appear, 2014.

[83] T. Kruse and A. Kirsch. Towards opportunistic action selection in human-robot cooperation. In *33rd Annual German Conference on Artificial Intelligence (KI 2010)*, 2010.

[84] T. Kruse, A. Kirsch, E. A. Sisbot, and R. Alami. Dynamic generation and execution of human aware navigation plans. In *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010.

[85] T. Kruse, A. Kirsch, E. A. Sisbot, and R. Alami. Exploiting human cooperation in human-centered robot navigation. In *RO-MAN, IEEE*, 2010.

[86] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726 – 1743, 2013.

[87] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard. Feature-based prediction of trajectories for socially compliant navigation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[88] A. Kushleyev and M. Likhachev. Time-bounded lattice for efficient planning in dynamic environments. In *ICRA, IEEE*, 2009.

[89] C.-P. Lam, C.-T. Chou, K.-H. Chiang, and L.-C. Fu. Human-centered robot navigation - towards a harmoniously human-robot coexisting environment. *Robotics, IEEE Transactions on*, 27(1):99 –112, February 2011.

[90] D. Lambert. *Body Language*. HarperCollins, 2004.

[91] F. Large, C. Laugier, and Z. Shiller. Navigation among moving obstacles using the *NLVO*: Principles and applications to intelligent vehicles. *I. J. Autonomous Robots*, 19(2):159–171, 2005.

[92] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[93] S. M. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[94] S. Lemai and F. Ingrand. Interleaving temporal planning and execution in robotics domains. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2004.

[95] M. Luber, L. Spinello, J. Silva, and K. O. Arras. Socially-aware robot navigation: A learning approach. In *IROS, IEEE/RSJ*, 2012.

[96] P. Maes. Artificial life meets entertainment: lifelike autonomous agents. *Commun. ACM*, 38(11):108–114, Nov. 1995.

[97] J. Mainprice, M. Gharbi, T. Siméon, and R. Alami. Sharing effort in planning human-robot handover tasks. In *RO-MAN, IEEE*, 2012.

[98] E. Martinson. *Acoustical awareness for intelligent robotic action*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2007. AAI3294521.

[99] E. Martinson. Hiding the acoustic signature of a mobile robot. In *IROS, IEEE/RSJ*, 2007.

[100] E. Martinson and D. Brock. Improving human-robot interaction through adaptation to the auditory scene. In *HRI, ACM/IEEE*, New York, NY, USA, 2007. ACM.

[101] J. Mccarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence*, pages 463–502. Edinburgh University Press, 1969.

[102] D. McDermott. A reactive plan language. Technical report, Yale University, Computer Science Dept., 1993.

[103] D. McDermott. The 1998 AI Planning Systems Competition. *AI Magazine*, 21(2):35–55, 2000.

[104] J. Miura and Y. Shirai. Parallel scheduling of planning and action for realizing an efficient and reactive robotic system. In *ICARCV*, pages 246–251, 2002.

[105] A. Müller. *Transformational Planning for Autonomous Household Robots using Libraries of Robust and Flexible Plans*. PhD thesis, Technische Universität München, 2008.

[106] A. Müller and M. Beetz. Towards a plan library for household robots. In *Proceedings of the ICAPS'07 Workshop on Planning and Plan Execution for Real-World Systems: Principles and Practices for Planning in Execution*, Providence, USA, September 2007.

[107] J. Müller, C. Stachniss, K. O. Arras, and W. Burgard. Socially inspired motion planning for mobile robots in populated environments. In *International Conference on Cognitive Systems (CogSys'08)*, Karlsruhe, Germany, 2008.

[108] V. Narayanan, M. Phillips, and M. Likhachev. Anytime safe interval path planning for dynamic environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2012.

[109] D. Nau, O. Ilghami, U. Kuter, J. W. Murdock, D. Wu, and F. Yaman. Shop2: An htn planning system. *Journal of Artificial Intelligence Research*, 20:379–404, 2003.

[110] T. Ohki, K. Nagatani, and K. Yoshida. Collision avoidance method for mobile robot considering motion and personal spaces of evacuees. In *IROS, IEEE/RSJ*, 2010.

[111] E. Pacchierotti, H. Christensen, and P. Jensfelt. Evaluation of passing distance for social robots. In *RO-MAN, IEEE*, September 2006.

[112] A. Pandey and R. Alami. A framework for adapting social conventions in a mobile robot motion in human-centered environment. In *Advanced Robotics, 2009. (ICAR)*, 2009.

[113] A. Pandey and R. Alami. A step towards a sociable robot guide which monitors and adapts to the person's activities. In *Advanced Robotics, 2009. (ICAR)*, pages 1 –8, june 2009.

[114] A. Peters, T. P. Spexard, P. Weiß, and M. Hanheide. Make room for me - a spatial and situational movement concept in hri. *Workshop on Behavior Monitoring and Interpretation*, September 2009.

[115] R. Philippsen and R. Siegwart. Smooth and efficient obstacle avoidance for a tour guide robot. In *ICRA, IEEE*, 2003.

[116] M. Phillips and M. Likhachev. Sipp: Safe interval path planning for dynamic environments. In *ICRA, IEEE*, 2011.

[117] E. Prassler, D. Bank, and B. Kluge. Key technologies in robot assistants: Motion coordination between a human and a mobile robot. *Transactions on Control, Au-*

tomation and Systems Engineering, 4, 2002.

[118] E. Prassler, J. Scholz, and P. Fiorini. Navigating a robotic wheelchair in a railway station during rush hour. *I. J. Robotic Res.*, 18(7):711–727, 1999.

[119] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[120] J. Rios-Martinez. *Socially-Aware Robot Navigation: combining Risk Assessment and Social Conventions*. PhD thesis, Universit'e de Grenoble, Atlanta, GA, USA, 2012.

[121] J. Rios-Martinez, A. Renzaglia, A. Spalanzani, A. Martinelli, and C. Laugier. Navigating between people: A stochastic optimization approach. In *ICRA, IEEE*, 2012.

[122] J. Rios-Martinez, A. Spalanzani, and C. Laugier. Understanding human interaction for probabilistic autonomous navigation using risk-rrt approach. In *IROS, IEEE/RSJ*, pages 2014–2019, 2011.

[123] P. P. Saffiotti, S. Parsons, O. Pettersson, A. Saffiotti, and M. Wooldridge. Robots with the best of intentions, 1999.

[124] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent asimo: system overview and integration. In *In IEEE/RSJ International Conference on In Intelligent Robots and System*, volume 3, pages 2478–2483, 2002.

[125] O. Sapena and E. Onaindía. Planning in highly dynamic environments: an anytime approach for planning under time constraints. *Applied Intelligence*, 29(1):90–109, 2008.

[126] A. Sardar, M. Joosse, A. Weiss, and V. Evers. Don't stand so close to me: users' attitudinal and behavioral responses to personal space invasion by robots. In *HRI*, pages 229–230, 2012.

[127] S. Satake, T. Kanda, D. F. Glas, M. Imai, H. Ishiguro, and N. Hagita. How to approach humans? strategies for social robots to initiate interaction. In *HRI, ACM/IEEE*, 2009.

[128] L. Scandolo and T. Fraichard. An anthropomorphic navigation scheme for dynamic scenarios. In *ICRA, IEEE*, February 2011.

[129] P. Schermerhorn, J. Benton, M. Scheutz, K. Talamadupula, and S. Kambhampati. Finding and exploiting goal opportunities in real-time during plan execution. *Robotics*, pages 3912–3917, 2009.

[130] M. Scheutz, P. W. Schermerhorn, J. F. Kramer, and D. Anderson. First steps toward natural human-like hri. *I. J. Autonomous Robots*, 22(4):411–423, 2007.

[131] M. Schut and M. Wooldridge. Principles of intention reconsideration. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, 2001.

[132] T. Siméon, J.-P. Laumond, and F. Lamiraux. Move3D: a generic platform for path planning. In *4th International Symposium on Assembly and Task Planning (ISATP 01)*, Fukuoka, Japan, May 28–29 2001.

[133] E. A. Sisbot. *Towards Human-Aware robot Motions*. PhD thesis, LAAS/CNRS, Universite Paul Sabatier, October 2008.

[134] E. A. Sisbot, A. Clodic, R. Alami, and M. Ransan. Supervision and motion planning for a mobile manipulator interacting with humans. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, HRI '08, pages 327–334. ACM, 2008.

[135] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon. A human aware mobile robot motion planner. *IEEE Transactions on Robotics*, 23, 2007.

[136] G. L. Steele, Jr. *Common LISP: the language (2nd ed.)*. Digital Press, Newton, MA, USA, 1990.

[137] M. Svenstrup, T. Bak, and H. J. Andersen. Trajectory planning for robots in dynamic human environments. In *IROS, IEEE/RSJ*, October 2010.

[138] S. Tadokoro, M. Hayashi, Y. Manabe, Y. Nakami, and T. Takamori. On motion planning of mobile robots which coexist and cooperate with human. *IROS, IEEE/RSJ*, 1995.

[139] L. Takayama and C. Pantofaru. Influences on proxemic behaviors in human-robot interaction. In *IROS, IEEE/RSJ*, 2009.

[140] S. Thompson, T. Horiuchi, and S. Kagami. A probabilistic model of human motion and navigation intent for mobile robot path planning. *4th International Conference on Autonomous Robots and Agents*, 2009.

[141] G. D. Tipaldi and K. O. Arras. Please do not disturb! minimum interference coverage for social robots. In *IROS, IEEE/RSJ*, 2011.

[142] P. Trautman and A. Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *IROS, IEEE/RSJ*, October 2010.

[143] J. P. van den Berg, M. C. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *ICRA, IEEE*, 2008.

[144] M. Walters, M. Oskoei, D. Syrdal, and K. Dautenhahn. A long-term human-robot proxemic study. In *RO-MAN, IEEE*, 2011.

[145] PR2 Robot, 2008. `www.willowgarage.com/pr2`.

[146] J. Young, Y. Kamiyama, J. Reichenbach, T. Igarashi, and E. Sharlin. How to walk a robot: A dog-leash human-robot interface. In *RO-MAN, IEEE*, 2011.

[147] F. Yuan, L. Twardon, and M. Hanheide. Dynamic path planning adopting human navigation strategies for a domestic mobile robot. In *IROS, IEEE/RSJ*, October 2010.

[148] H. Zender, P. Jensfelt, and G.-J. M. Kruijff. Human- and situation-aware people following. In *RO-MAN, IEEE*, 2007.

[149] B. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. D. Bagnell, M. Hebert, A. Dey, and S. Srinivasa. Planning-based prediction for pedestrians. In *IROS, IEEE/RSJ*, October 2009.