Learning Human Preferences for Physical Human-Robot Cooperation

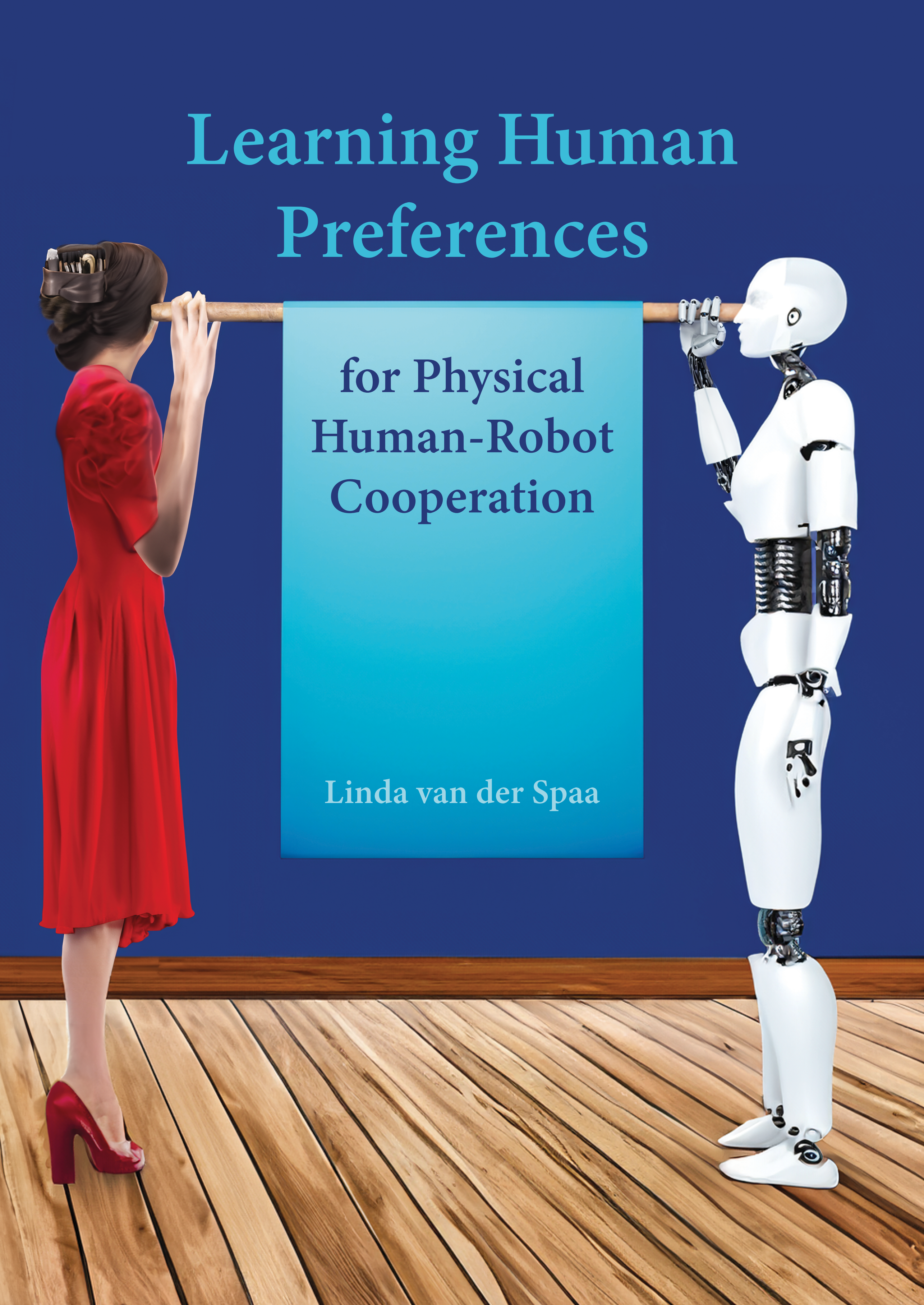
van der Spaa, L.F.

DOI
10.4233/uuid:90f0c7fe-34db-45f3-bd2b-7fec91075d20

Publication date
2024

Document Version
Final published version

Citation (APA)
van der Spaa, L. F. (2024). *Learning Human Preferences for Physical Human-Robot Cooperation*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:90f0c7fe-34db-45f3-bd2b-7fec91075d20



Learning Human Preferences

for Physical Human-Robot Cooperation

Linda van der Spaa

Learning Human Preferences for Physical Human-Robot Cooperation

Learning Human Preferences for Physical Human-Robot Cooperation

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 1 februari 2024 om 12:30 uur

door

Linda Fiona van der Spaa

Master of Science in Systems and Control &
Master of Science in Mechanical Engineering,
Technische Universiteit Delft, Nederland
geboren te Utrecht, Nederland.

Dit proefschrift is goedgekeurd door de

Promotor: dr.-ing. J. Kober
Promotor: prof. dr. R. Babuška

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Dr.-ing. J. Kober,	Technische Universiteit Delft
Prof. dr. R. Babuška,	Technische Universiteit Delft
Onafhankelijke leden:	
Prof. dr. ir. D.A. Abbink	Technische Universiteit Delft
Prof. dr. ir. A. Bozzon	Technische Universiteit Delft
Dr. ir. D.J. Broekens	Universiteit Leiden
Dr. S. Ivaldi	Inria Nancy
Overige leden:	
Dr.-ing. M. Gienger,	Honda Research Institute Europe

Dr. M. Gienger heeft in belangrijke mate aan de totstandkoming van het proefschrift bijgedragen.

Keywords: Physical Human-Robot Interaction, Human-Robot Collaboration, Human Preferences, Human-Centered Planning, Inverse Reinforcement Learning

Printed by: Ridderprint | www.ridderprint.nl

Front & Back: Linda van der Spaa

ISBN 978-94-6483-779-7

An electronic version of this dissertation is available at http://repository.tudelft.nl/.

“The path is the goal.”

Mahatma Gandhi

Contents

Summary

PHYSICAL human-robot cooperation (pHRC) has the potential to combine human and robot strengths in a team that can achieve more than a human and a robot working on the task separately. However, how much of the potential can be realized depends on the quality of cooperation, in which awarenes of the partner's intention and preferences plays an important role. Preferences tend to be highly personal, and additionally depend on the cooperation partner and the cooperation itself. They can be hard to define in terms a robot would understand, and may change over time. This thesis focuses on learning 'useful models' from observed behavior, to let our robot adapt its behavior to better match its human partner's preferences, and thus improve the cooperation.

The aim is to capture personalized approximate models of human preferences –*how* a person likes to do something– from very few interactive observations, providing only small amounts of imprecise data, such that the robot can use the model to improve each user's comfort. First, we learn a model to predict and optimize the human ergonomics in a pHRC task, such that our robot can propose a plan, for both the human and itself, to solve the task in a way that is more ergonomic for its human partner. However, people do not necessarily prefer to act ergonomically, nor do we want to impose on them what a robot thinks best. Therefore, next, we apply inverse reinforcement learning (IRL), to capture less restrictive preference models: 1) path and velocity preferences for motion planning, and 2) on a higher level of abstraction, which (grasp or motion) action to initiate for proactive physical support. For learning to take the correct action in cooperation, we developed the disagreement-aware variable impedance (DAVI) controller to smoothly transition between providing active guidance and allowing the human to demonstrate alternative behavior.

In Chapter 2, we present a model to predict the ergonomics of the human partner, for optimization during collaborative task planning, of sequential tasks involving pHRC over the entire duration of the task (not just handing something over). We consider tasks that are planned as a sequence of where a human and a mobile bimanual robot are holding an object. Specifically, we test our model on the task of cooperatively rotating a large box. From a small data set of poses observed when the user performed the task with another human, we train a rough inverse kinematics (IK) transformation to obtain a personal estimate of the full-body pose given where the hands will be. Additionally, we estimate the load on the hands from first principles. From the full-body pose and load estimates, the ergonomic cost is computed, using a metric widely accepted for ergonomic evaluation and validated by experts in the field. The resulting ergonomic cost term is added to the objective function of the existing planner to which we compare, both in simulation and in a small user study. We also verify the load estimation on the robot hands. For different users, the ergonomic planner proposed different solutions which considerably reduced the time spent in poses with "high ergonomic risk" as defined by the ergonomic cost metric. It is advised to re-evaluate with engineers and ergonomics experts which ergonomic cost metric should best be used.

Next, in Chapter 3, we present a framework to capture both path and velocity preferences, on a trajectory level, from very few physical corrections. We consider tasks in which a robot arm is to meet a user's preferences while transporting an object around an obstacle. The user can correct the robot by kinesthetically demonstrating an alternative trajectory. We apply IRL to learn parameterized path and velocity preferences from these demonstrations. The parameterization makes our model independent of the specific context (e.g. between which points to move). The key of our method being able to learn both path and velocity preferences is the separation of the two in the optimization phase: first, the path is optimized; then, the velocity is optimized along the path. It offers users the flexibility to demonstrate their path and velocity preferences either simultaneously or in separate demonstrations. The robot has optimization objectives of its own, separate from the learned human preference model. These two parts of the cost function, which provides robot's optimization criterion, counterbalance each other. In a user study, users were free to chose their own preferences. Regardless of their choice, they felt the robot understood both their path and motion preferences, and could generalize them well over different contexts, requiring low effort. The users confirmed the consistency of the preferences in a separate test testing the generalization. An additional comparison study, supported by simulation results, discusses the structural differences of the proposed method compared to two methods from literature.

To learn human preferences during physical cooperation, we developed the Disagreement-Aware Variable Impedance (DAVI) controller presented in Chapter 4. It allows the robot to transition smoothly between providing active guidance and being passive with zero stiffness, except for gravity compensation, allowing the human partner to demonstrate new behavior. The algorithm detects disagreement based on the interaction forces. The impedance that made the robot track its trajectory is gradually decreased as long as a disagreement is detected. It is increased again if the user stops disagreeing, and kept at zero once it reaches zero. This allows both haptic negotiation of where to go between user and robot, and teaching the robot alternative behavior in case of disagreement. We demonstrate smooth interaction with a small group of users, although preferences differ about how fast the robot should give in, among other things.

This DAVI controller is then used in Chapter 5, where we present a method for intention-aware preference learning during cooperation; how to solve a sequential task that requires simultaneous physical action of both human and robot, without explicit communication of the human intention, i.e., what specific goal the human wants the team to achieve. Modeling the task as a Markov Decision Process (MDP), the robot starts off with a nominal policy and an initial estimate of the human preferences. The human preference model (reward function) is IRL updated after every episode in which the human-robot team try the task, corrected for the influence of the robot. The robot solves a Partially Observable (PO) MDP, as it has no access to the human intention. The intention is estimated at runtime from the observed human actions. The method is tested in simulation and with a group of novice users, on a cooperative carrying task defined in a discrete state space connected by pre-programmed primitive robot actions. The presented Learner outperformed the baseline taking fewer wrong actions and providing support in previously unseen states generalizing over intentions. Users felt that their preferences and intentions were better understood, found it easier to work with the learner, and trusted the robot more. An additional simulation study provided additional insights in the effect of learning parameters and increased task complexity.

Overall, this thesis presents a number of methods that were successful in capturing personalized models of people for improved human-robot cooperation, from very little data. The closing outlook presents a vision of how the separate contributions could be combined in the future in a single framework that would enable robots to provide personalized physical assistance to improve people's ergonomics as well as their perceived comfort.

SAMENVATTING

FYSIEKE mens-robot samenwerking (pHRC) heeft de potentie om de sterke punten van een mens en een robot te combineren in een team dat meer kan bereiken dan een mens en een robot die afzonderlijk aan de taak werken. Hoeveel van het potentieel gerealiseerd kan worden hangt echter af van de kwaliteit van de samenwerking, waarbij bewustzijn van de intentie en voorkeuren van de partner een belangrijke rol speelt. Voorkeuren zijn vaak zeer persoonlijk en bovendien afhankelijk van de samenwerkingspartner en de samenwerking zelf. Ze kunnen moeilijk te definiëren zijn in termen die een robot zou begrijpen, en kunnen in de loop van de tijd veranderen. Dit proefschrift richt zich op het leren van 'bruikbare modellen' uit geobserveerd gedrag, zodat onze robot zijn gedrag kan aanpassen om het beter overeen te laten komen met de voorkeuren van zijn menselijke partner en zo de samenwerking te verbeteren.

Het doel is om gepersonaliseerde benaderende modellen van menselijke voorkeuren vast te leggen –*hoe* iemand iets graag doet– op basis van zeer weinig interactieve observaties, die slechts kleine hoeveelheden onnauwkeurige gegevens opleveren, zodat de robot het model kan gebruiken om het comfort van elke gebruiker te verbeteren. Eerst leren we een model om de menselijke ergonomie in een pHRC-taak te voorspellen en te optimaliseren, zodat onze robot een plan kan voorstellen, voor zowel de mens als zichzelf, om de taak op een manier op te lossen die ergonomischer is voor zijn menselijke partner. Mensen geven er echter niet noodzakelijk de voorkeur aan om ergonomisch te handelen en we willen hen ook niet opleggen wat een robot het beste vindt. Daarom passen we invers versterkend leren (IRL) toe om minder beperkende voorkeursmodellen vast te leggen: 1) pad- en snelheidsvoorkeuren voor het plannen van bewegingen, en 2) op een hoger abstractieniveau, welke (grijp- of bewegings)actie te initiëren voor proactieve fysieke ondersteuning. Om de juiste actie te leren kiezen tijdens het samenweren, hebben we de onenigheid-bewuste variable weerstandsregelaar (disagreement-aware variable impedance, DAVI, controller) ontwikkeld om soepel te schakelen tussen het geven van actieve ondersteuning en het toelaten dat de mens alternatief gedrag demonstreert.

In Hoofdstuk 2 presenteren we een model om de ergonomie van de menselijke partner te voorspellen, voor optimalisatie tijdens gezamenlijke taakplanning, van opeenvolgende taken waarbij gedurende de hele taak fysieke interactie plaatsvindt (niet alleen iets overhandigen). We beschouwen taken die gepland zijn als een reeks handelingen waarbij een mens en een mobiele tweearmige robot een object vasthouden. Specifiek testen we ons model op de taak: het samen roteren van een grote doos. Op basis van een kleine dataset van houdingen die zijn geobserveerd toen de gebruiker de taak uitvoerde met een andere mens, trainen we een ruwe invers kinematische (IK) transformatie om een persoonlijke schatting te verkrijgen van de volledige lichaamshouding gegeven waar de handen zich zullen bevinden. Daarnaast schatten we de belasting op de handen vanuit natuurkundige basisprincipes. Op basis van de schattingen van de lichaamshouding

en de belasting wordt de ergonomische last berekend met behulp van een maatstaf die algemeen geaccepteerd is voor ergonomische evaluatie en gevalideerd is door deskundigen op dat gebied. De resulterende term voor de ergonomische last wordt toegevoegd aan het optimalisatiecriterium van de bestaande planner waarmee we vergelijken, zowel in simulatie als in een klein gebruikersonderzoek. We verifiëren ook de schatting van de belasting van de robothanden. Voor verschillende gebruikers stelde de ergonomische planner verschillende oplossingen voor die de tijd die werd doorgebracht in houdingen met een "hoog ergonomisch risico", zoals gedefinieerd door gebruikte de ergonomische maatstaf, aanzienlijk verminderden. Er wordt geadviseerd om samen met ingenieurs en ergonomie-deskundigen opnieuw te evalueren welke ergonomische maatstaf het beste gebruikt kan worden.

Vervolgens presenteren we in Hoofdstuk 3 een raamwerk om zowel pad- als snelheidsvoorkeuren vast te leggen op bewegingsniveau, op basis van zeer weinig fysieke correcties. We beschouwen taken waarbij een robotarm moet voldoen aan de voorkeuren van een gebruiker terwijl hij een object rond een obstakel verplaatst. De gebruiker kan de robot corrigeren door kinesthetisch een alternatief traject aan te tonen. We passen IRL toe om geparametriseerde pad- en snelheidsvoorkeuren te leren op basis van deze demonstraties. De parametrisatie maakt ons model onafhankelijk van de specifieke context (bijvoorbeeld tussen welke punten te bewegen). De sleutel van onze methode om zowel pad- als snelheidsvoorkeuren te leren, is de scheiding van de twee in de optimalisatiefase: eerst wordt het pad geoptimaliseerd, daarna wordt de snelheid langs het pad geoptimaliseerd. Het biedt gebruikers de flexibiliteit om hun pad- en snelheidsvoorkeuren gelijktijdig of in afzonderlijke demonstraties te tonen. De robot heeft zijn eigen optimalisatiedoelen, los van het geleerde menselijke voorkeursmodel. Deze twee delen van de kostenvergelijking, die het optimalisatiecriterium van de robot vormt, houden elkaar in evenwicht. In een gebruikersonderzoek waren gebruikers vrij om hun eigen voorkeuren te kiezen. Ongeacht hun keuze hadden ze het gevoel dat de robot zowel hun pad- als bewegingsvoorkeuren begreep en deze goed kon generaliseren over verschillende contexten, waarbij weinig inspanning nodig was. De gebruikers bevestigden de samenhang van de voorkeuren in een afzonderlijke test waarin de generalisatie werd getest. Een aanvullende vergelijkende studie, ondersteund door simulatieresultaten, bespreekt de structurele verschillen van de voorgestelde methode vergeleken met twee methoden uit de literatuur.

Om menselijke voorkeuren tijdens fysieke samenwerking te leren, hebben we de onenigheid-bewuste variable weerstandsregelaar ontwikkeld die in Hoofdstuk 4 wordt beschreven. Hiermee kan de robot soepel schakelen tussen actieve sturing en passief-zijn zonder stijfheid, behalve voor zwaartekrachtcompensatie, zodat de menselijke partner nieuw gedrag kan laten zien. Het algoritme detecteert onenigheid op basis van de interactiekrachten. De stijfheid die de robot zijn traject liet volgen wordt geleidelijk verlaagd zolang er een onenigheid wordt gedetecteerd. De stijfheid wordt weer verhoogd als de gebruiker het niet langer oneens is, en wordt op nul gehouden wanneer deze eenmaal nul is. Dit maakt zowel haptische onderhandeling tussen gebruiker en robot mogelijk, als het aanleren van alternatief gedrag aan de robot in het geval van onenigheid. We demonstreren een soepele interactie met een kleine groep gebruikers, hoewel de voorkeuren verschillen over onder andere hoe snel de robot zou moeten toegeven.

Deze DAVI-controller wordt vervolgens gebruikt in Hoofdstuk 5, waar we een methode presenteren voor het intentiebewust leren van voorkeuren tijdens samenwerking; hoe los je een taak op die bestaat uit een opeenvolging van handelingen die gelijktijdige fysieke actie vereisen van mens en robot, zonder expliciete communicatie over de menselijke intentie, d.w.z. welk specifiek doel de mens wil dat het team bereikt. Door de taak te modelleren als een Markov-beslissingsprobleem (Markov decision process, MDP), begint de robot met een nominaal beleid en een initiële schatting van de menselijke voorkeuren. Het menselijke voorkeurenmodel (beloningsfunctie) wordt bijgewerkt door middel van IRL na elke episode waarin het mens-robotteam de taak probeert, gecorrigeerd voor de invloed van de robot. De robot lost een gedeeltelijk waarneembaar (Partially Observable, PO) MDP op, omdat het geen toegang heeft tot de menselijke intentie. De intentie wordt tijdens de uitvoering van de taak geschat op basis van de geobserveerde menselijke acties. De methode is getest in simulaties en met een groep beginnende gebruikers, op een samenwerkingstaak voor het dragen van een object, gedefinieerd in een discrete toestandsruimte verbonden door voorgeprogrammeerde basisrobotacties. De gepresenteerde leerder presteerde beter dan de alternatieve basis door minder verkeerde acties uit te voeren, en biedt ondersteuning in voorheen ongeziene toestanden door te generaliseren over intenties.

Over het geheel genomen presenteert dit proefschrift een aantal methoden die succesvol waren in het vastleggen van gepersonaliseerde modellen van mensen voor verbeterde samenwerking tussen mens en robot, uit zeer weinig gegevens. De afsluitende vooruitblik geeft een visie op hoe de afzonderlijke bijdragen in de toekomst gecombineerd zouden kunnen worden in één enkel raamwerk dat robots in staat zou stellen om gepersonaliseerde fysieke hulp kunnen bieden om de ergonomie en het comfort van mensen te verbeteren.

1

INTRODUCTION

"All models are wrong,
but some are useful."

George Box

COOPERATION has the potential to combine multiple actors' strengths, such that the overall achievement is more than what can be achieved when the actors are acting separately. For example, a single person may be able to push a sofa, if it is not too heavy and the friction between the sofa and the floor is low enough. A threshold may be a serious obstacle, one that may not be possible to overcome if it crosses an opening that is not much wider than the sofa and does not allow walking around to lift first one side through and then the other. Stairs are even worse. With two people, one on each end, it becomes much easier to maneuver the sofa, and previously insurmountable obstacles may become negotiable.

Introducing a robot into a cooperative team potentially allows the combination of strengths of the human and the robot. Humans are typically flexible, good at overseeing the bigger picture, and finding creative solutions (just to name a few), while robots can be built to provide strength, precise coordination, and repeatable reliable behavior. To return to the example of the sofa, the robot may do the heavier lifting or very precise avoidance of doorposts.

However, both the achievement and the experience of the people involved depend very much on the quality of the cooperation. A key factor in that is consideration of each other's preferences. Some preferences in the sofa example could be: to hold it below or by the sides, thus holding it at a different height; rotating it when passing through doorways; steering more or less wide around obstacles; the speed at which to move it. If the partner on the team does not understand that you want to do something differently, it may lead to annoyance, increased effort of solving the task, or even failure to complete the task.

Different actors tend to have different preferences. Additionally, in cooperative scenarios, preferences also depend on the partner(s) in the cooperation and on the cooperation itself. For example, during the first time cooperating as a team, you might want to take it slowly. Probably, you would grasp an object somewhere else if your partner is much taller, shorter, or stronger than you. Preferences may also be not so easy to define or explain explicitly. People generally find it hard to define what they prefer specifically. Even if they would know, it would be hard to describe their preferences in a way that a robot would 'understand'. Instead, it is much easier for people to demonstrate, or try out, how they would like to do something.

The challenge taken on in this thesis is to make a robot learn a 'useful model' from observed behavior, that allows it to adapt its cooperative behavior to better match its partner's preferences and thus improve the overall cooperation. Specifically, the focus of this thesis is on *physical* human-robot cooperation (pHRC), more specifically, on cooperatively moving around large and/or bulky objects. It would be really helpful to have a robot that could take most of the weight of an object, and actively help maneuver it, while the human determines where it should go and has a preference on how to move it there. The earlier example of moving a sofa is just one example. If the object is a (grand) piano, it is even more important that the team moving it works well together. Yet also for smaller and lighter objects, physical help may be very welcome, for example to people who lack the necessary strength.

Before further introducing the topics researched in this thesis, I will first clarify some important terminology. Then, after posing the research questions, I will discuss the rel-

Figure 1.1: Two motivating examples, of cooperatively moving respectively a glass panel (left) and a sofa as part of loading a van for moving (right)[1].

evant general background on which this thesis continues to build, before elaborating on the taken approaches. The section closes with a summary of the contributions and the outline of the remainder of the thesis.

1.1. Terminology

Even when all appear to speak the same language, different people use the same words for different things depending on context and professional or personal background, among (probably many) other things. Therefore, before elaborating any further on the content of this thesis, this section is an attempt to get all readers on the same page regarding a number of concepts that are fundamental to the work.

Agents

The term ‘agent’ is used quite generally as someone or something that acts, in the world or on something more specifically. (It can be an animal hunting for food, or a chemical eroding some material.) In its most general meaning, this is very abstract. In this thesis ‘agent’ refers to a robot or human involved in a task (which can be cooperative). Whenever the theory might apply regardless of the agent being a human or a robot, I will use the word ‘agent’.

Tasks and Contexts

Tasks can be considered at many levels of abstraction, from tightening a bolt to cooking dinner. In general, we could say that a task involves one or more actors who act in or upon their environment in order to achieve some change, whether that may be a tightened bolt or ingredients turned into dinner.

In this thesis, I consider tasks the same if the differences can be parameterized. E.g., “tightening a bolt” remains the same task, regardless of the position of the bolt, its size or the shape of its head. Parameters could capture task specifics like position, orientation, head shape, size, tightening torque. We call these *context parameters*. By varying such parameters, we can change the context within a task and test how well an agent is able to do a task independent of the specific context.

[1] Pictures fall under a CC NC license.

PREFERENCES AND INTENTIONS

While working on a task, agents can have preferences and intentions. In literature, the terms *preference* and *intention* are used interchangeably to point to the same concepts. In my work, I make the following explicit distinction:

- **Intention** – the thing an agent wants to achieve, *what* someone is trying to do. This may be a long-term goal, or just a first objective in a chain that may eventually lead to a higher goal.
- **Preference** – the way to go about fulfilling the intention, *how* someone likes to do something. Generally, one goal can be achieved in multiple ways. The specific way someone likes to go about it can be highly personal.

Considering the task of moving a box, the intention would be where to move the box to, and a preference to slide it until it needs to get lifted onto or over something. Considering the task of restacking a stack of boxes, the intention would be how to have them restacked in the end, and preferences would be in what order to move them and through what intermediate configurations. Comparing these two examples, we see that a preference in a more complex task can be viewed as the intention of a subtask.

A task with discrete actions could model any task of which the actions can be considered subtasks. Tasks at the trajectory level with preferences in the continuous space and time domain, such as over which path and how fast to slide the above-mentioned box, are considered fundamental in this thesis, as they are non-trivial to break down further into subtasks. By looking into tasks both at this fundamental level and at a level above, I aim to prepare the way to learn more complex multi-layered preferences in future work, including the corresponding inference of intentions on the multiple levels.

In this thesis, preferences are seen as personal traits, whereas we assume intentions are universal. The human with whom we want to make our robot cooperate will have certain preferences, and act (in an attempt) to fulfill a certain intention. For every intention, the human will feel most comfortable if they can act fully according to all their preferences. The robot objective is to let the human act as close as possible to this (very subjective) optimum.

In Chap. 2, we assume the intention (task goal) to be known, and we assume the human prefers to perform the task ergonomically. In later chapters, I drop these assumptions. In Chap. 4 and 5, neither preferences nor the intention will be shared explicitly with the robot. –For more complex tasks, people would not want to communicate their intention for every subtask.– Then, the robot additionally needs to correctly infer the human intention, to match its task goal before a mismatch can cause any discomfort.

MODELS AND SYSTEMS

Few words are used to mean so many different things. Role models and scale models aside, even the variety of mathematical models is large. Models can be as diverse as the systems they describe. From a system perspective, a good model captures the relevant system properties. This may serve further analysis of the system to obtain a deeper understanding and discover additional properties, or it may serve to predict the behavior of the system, whether that system is a fundamental particle, local legislation, a robot arm, or even a person.

It depends very much on what we want to do with, or know about, our system, what kind of model is useful. If we want to know if a chair will hold a person, an approximate model of the person's geometry and weight will suffice. If we want to predict how the person will respond to something we say, we need a very different kind of model.

On the other hand, it depends on what we can measure/compute what model we can construct. If we know the geometry, materials, and actuator properties of our robot arm, we can construct a model of what signals we need to send to the actuators to let the arm move the way we want. Some measurements may be necessary to infer additional properties (e.g., friction in the joints) to make the model sufficiently accurate. (What is "sufficiently accurate" depends on the application.) However, if we again want to predict a person's reaction to spoken text, our knowledge of the human brain is insufficient to construct a model the way we did for the robot arm.

Every model is an approximation. Sometimes, the approximation can be very accurate, but usually at the cost of increased complexity. Complex models tend to get costly in terms of time and resources, or even mathematically impossible, to verify or solve. Mistakes are easily made and hard to find. Nor does increased complexity guarantee improved accuracy. From the engineering perspective in this thesis, the model we use only needs to be as accurate as is useful for the application.

In this thesis, I seek to model the tasks, the robot, and the human/partner agent involved in such a way that a robot optimizing its actions according to the models increases the human comfort on the task. In different chapters, we try out different metrics to measure the human comfort, ranging from computing ergonomics scores and measuring interaction forces to questionnaires. In what way we model the human depends on which aspect(s) of comfort we try to optimize for.

As people are very diverse and act on all kinds of personal preferences, we cannot model them following "laws of mechanics of human behavior". Instead, we seek to *learn* a model of the person whose comfort we want to optimize for, solely for that purpose within the objective of solving the (cooperative) task.

1.2. PROBLEM DEFINITION AND CHALLENGES

IN this thesis, my focus is to learn a model that captures human *preferences* in such a way that the user comfort increases, and users need no prior training or knowledge on robotic systems. Specifically, I focus on tasks which involve a physical robot, preferably physically cooperating with a human partner in order to achieve a goal which the human decides on.

The main research question of this thesis is how to learn a useful model that captures human preferences already from very little data, to improve the physical cooperation of a robot with this person. We learn personal models, capturing the preferences of one specific person, for tailored cooperation behavior.

Standard machine learning methods generally require a lot of data, orders of magnitude more than can feasibly be provided by a human. This is even more the case if we want to learn personalized behavior for a single person. When we want to learn cooperation preferences during cooperation, collecting data becomes even harder and thus more expensive. Next to that, it is very hard to verify high-dimensional or continuous-domain preferences with high precision. In many cases, people themselves are already not very precise in their preferences. In contrast to the robot learning (or control) problems of “just” solving some task, without caring about what a human may feel about it, there is no analytical, or even pre-trained, model available for human behavior and preferences that I could apply to the pHRC setting I consider in this thesis. Even if we would pre-train such a model, or construct it from principles in psychology, it would not be personal, and preferences can vary a lot between different people. Therefore, I choose to focus on learning approximate models that let the robot improve its behavior from a similar number of iterations as we would expect an average fellow person to need to learn, generalizing between similarly structured contexts. I take on the challenges of learning from imprecise data, as well as having very little data available to learn from.

First, we sought to improve the physical cooperation by modeling and optimizing the ergonomics of the human partner (Chap. 2). Here, we looked into how to learn a personalized model that allowed us to predict the human posture and load on the hands due to the task. Especially predicting the posture is a challenge, as many different postures can be assumed without seeing any difference on the object that is moved (to name an example of a cooperative task). The mapping from holding an object in a certain way to full-body pose is very redundant, and therefore very personal. Furthermore, again, people do not necessarily show precise, or even consequent, behavior when it comes to choosing a posture to hold something. This research focused on learning a personalized model capturing the human posture and load on the hands in such a way that it could be used for optimal planning: allowing our robot to compute a solution for a task requiring four hands, minimizing the ergonomic cost for its human partner.

However, while testing the developed method, we obtained evidence supporting the suspicion that many people’s preferences cannot be explained by an ergonomic model. People do not necessarily prefer what is healthy for them, nor do they like to be forced to make large changes. Habits can be strong (Verplanken and Orbell, 2022). Whether or not an aim for the future is to subtly influence people’s behavior for the benefit of their health –with careful consideration of the moral implications and risks of misuse–, the first step is to gain a better understanding of people’s personal preferences. Therefore, the rest of the PhD focused on learning a model of people’s preferences with fewer underlying assumptions. I approached this on two different levels of abstraction:

1. learning path and velocity preferences for improved motion planning to move between arbitrary locations in space, in the presence of an obstacle (Chap. 3);

2. learning which preferred (grasp or motion) action to initiate next for proactive object carrying support (Chap. 5).

Although literature discusses various methods for learning trajectory preferences, it is non-trivial to additionally learn velocity preferences. Rather than merely allowing different user speeds, or learning a frequency to a periodic motion, we wanted to be able to learn velocity preferences such that they would generalize over paths and contexts. We took on the challenge to incorporate learning of such velocity preferences into an inverse reinforcement learning (IRL) framework, to make our robot learn a combination of path *and velocity* preferences from a small number of feedback iterations, where we let users provide corrective feedback showing their preferences in an intuitive way, by physically moving the robot arm (Chap. 3).

On the higher level, we assume the robot knows how to do “primitive actions”, e.g. grasp or move from some point A to another point B. In a cooperative setting, where both actors (human and robot) are involved to complete a task, we need the robot initially already to be capable to help solving the task. Starting from a solution that is safe, or perhaps conceived by the robot as optimal, we want the robot to improve its behavior to better match its partner’s preferences, learning from its partner while cooperatively executing the task.

For this, we need to control the robot actions in such way that it will recognize when it is being corrected by the human. In physical cooperation, haptic communication, via interaction forces, is much more direct and intuitive than explicit communication (Pezzulo et al., 2021). Therefore, we set to develop a controller that would smoothly and safely transition between providing active guidance and allowing the partner to demonstrate alternative (more desired) behavior, in response to haptically detected disagreement (Chap. 4).

This controller allowed us to then further address the problem of learning from haptic feedback on the task: which action to take to best support the human-preferred way of solving the task (Chap. 5). Again, we want to learn from only a handful of interactive trials. Adding to the challenge is that the robot needs to learn the human preferences from observations it is influencing by its own actions. Additionally, we consider that the human in question does not explicitly communicate the intended task goal, e.g. where to ultimately place the object that is being moved cooperatively. The intention has to be inferred from the observed actions, so the robot can match it.

In all cases, we learn task-parameterized models, such that the learned preferences transfer between contexts. We test this by changing the start, obstacle (if applicable), and/or goal locations in between task episodes.

1.3. The Robots

During my research, I tested the developed methods on the following two robots (depicted in Fig. 1.2).

HRI Bimanual Dexterous Cooperation Robot

This robot, stationed at the Honda Research Institute Europe (HRI-EU) in Offenbach, consists of two Kuka LBR iiwa 820 arms mounted on a vertically actuated slide, which is in turn mounted on a mobile platform. The mobile platform is omni-directional, it can instantaneously translate and rotate on the horizontal plane. Each robot arms has a payload capacity of 14 kg, which includes the six-axis force torque sensor and seven DoF Schunk dexterous 3-finger hand with tactile pads mounted on each of the arms. More information on the robot can be found in Gienger et al. (2018).

With this robot, we tested the ergonomics prediction and optimization presented in Chap. 2. In Chap. 5, the robot reappears as an actor in the more complex of the two presented scenarios, the one on which the presented method was tested only in simulation.

Franka Emika Panda

A couple of these Panda robots are owned by the research group at the department of Cognitive Robotics at the mechanical Engineering faculty of the Delft University of Tech-

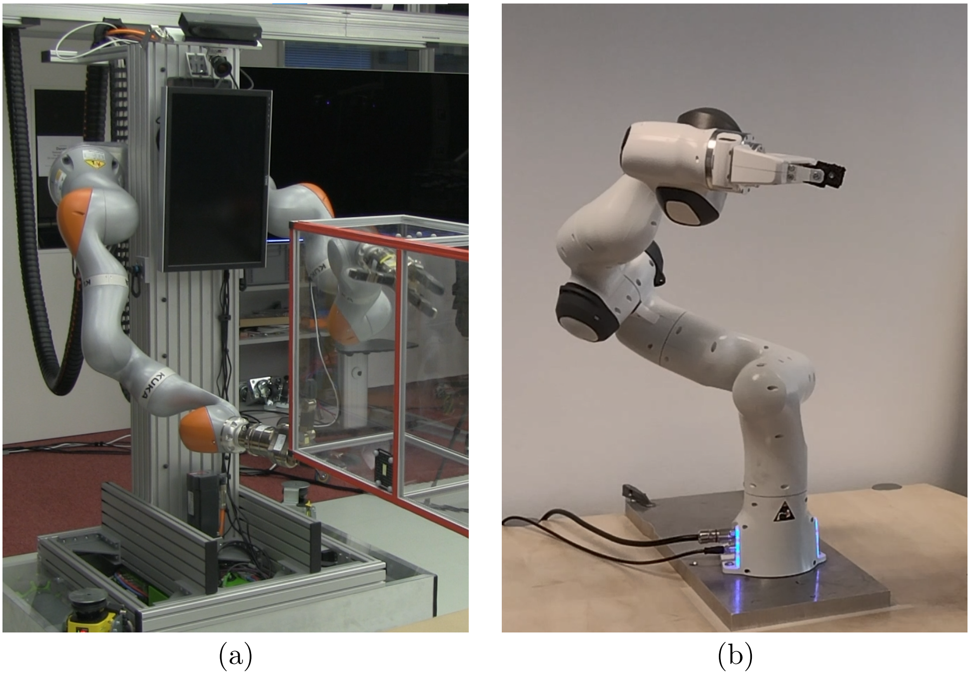


Figure 1.2: The two robots: (a) the HRI Dexterous Cooperation Robot, and (b) the Franka Emika Panda at TU Delft, with custom fingertips for improved grip.

nology. The 7 DoF arm, developed by Franka Emika, is certified to be safe in a workspace shared with humans. All joints have torque sensors and can be directly torque controlled. The control interface allowed us easy integration with Robot Operating System (ROS). The version of the hardware I used had to remain mounted standing on its base. It has a reach of 0.855 m and can carry a payload up to 3.0 kg.

This type of robot was used to conduct the user studies in Chap. 3 to 5.

1.4. CONTRIBUTIONS AND OUTLINE

IN the papers presented in the following chapters, the following contributions were made in modeling different subjective aspects of a human partner, all with the purpose of improved planning or optimization in control:

- In Chap. 2, we 1) develop a model that predicts the ergonomics of the human within a human-robot collaborative task, and 2) integrate the predictor in sequential task planning, resulting in a joint plan for the human and robot movements optimized for human ergonomics. Ergonomic optimization is applied to the task of manipulating, specifically rotating, a large object, which requires continuous interaction between the robot and the human partner.

- In Chap. 3, we propose a novel framework for optimizing trajectories in object-transportation tasks that meet the user's path and velocity preferences, where we first optimize the path and then the velocity on the path. Learning the path and velocity separately provides users with the option to avoid the challenge of providing a temporally consistent demonstration at each iteration. It offers users the flexibility to demonstrate their path and velocity preferences either simultaneously or in separate demonstrations. IRL is applied to update a task-parameterized preference model of the human from these demonstrations. Given a general task (e.g., move an object), the model is parameterized such to allow learning and generalizing preferences across different contexts (i.e., arbitrary but known start, goal, and obstacle locations).

- In Chap. 4, we present a control framework which modulates the robot stiffness as a function of perceived disagreement of the human, for learning high-level target policies. The control of the task is traded between the human and the robot, and when the robot is passive, it is interactively updating its belief of the desired policy. We focus on tasks where human and robot move an object to a new location in space, only communicating intuitively through the interaction forces.

- In Chap. 5, we present a novel method for learning a human preference model for intention-aware cooperation, from collaborative episodes in which the human does not explicitly communicate their intention. The method 1) learns a personalized model of a human partner from physically cooperating with this partner, from scratch or improving a nominal model; 2) models human preferences as an explicit function of intention, enforcing inherent intention awareness; 3) applies second order Theory of Mind (ToM) reasoning to model the human's preferences separate from the robot's, resulting in explicit partner awareness. This allows the robot to optimize an objective different from the human for improved cooperative behavior. We consider the problem of cooperatively moving an object, on the abstract level where the two agents choose their actions from a predefined set of primitive actions (e.g., "grasp object", "pull towards position ... in space"). The robot infers at runtime what is the most likely intended goal where its human partner wants to place the object.

Additionally, the models and derived optimized robot plans, trajectories, and policies have been tested in user studies of different sizes:

- All methods are tested with a real robot and real people.

- The participants of the user studies conducted in Chap. 3 to 5 were free to choose their own preferences.

- The quality of the preference learning, in Chap. 3 and 5, is evaluated with a sufficiently large group of mostly novice users to yield significant subjective results from questionnaires on the user experience.

Chap. 6 closes this thesis with an overarching conclusion and discussion of the presented research, followed by an outlook that presents a vision of how separate contributions might be combined in the future into a single common framework.

2

Predicting and Optimizing Ergonomics in Physical Human-Robot Cooperation Tasks

This chapter presents a method to incorporate ergonomics into the optimization of action sequences for bi-manual human-robot cooperation tasks with continuous physical interaction. Our first contribution is a novel computational model of the human that allows prediction of an ergonomics assessment corresponding to each step in a task. The model is learned from human motion capture data in order to predict the human pose as realistically as possible. The second contribution is a combination of this prediction model with an informed graph search algorithm, which allows computation of human-robot cooperative plans with improved ergonomics according to the incorporated method for ergonomic assessment. The concepts have been evaluated in simulation and in a small user study in which the subjects manipulate a large object with a 32 DoF bimanual mobile robot as partner. For all subjects, the ergonomic-enhanced planner shows their reduced ergonomic cost compared to a baseline planner.

The contents of this chapter have been slightly modified from the paper of the same name, by Linda van der Spaa, Michael Gienger, Tamas Bates, and Jens Kober, published in *IEEE International Conference on Robotics and Automation, 2020.*

2.1. Introduction

Industrial robots have begun to leave their cages, but are not yet anywhere near the point where they can cooperate with humans at an equal level. Through physical assistance and cooperation, robots have a large potential to make human lives easier and prevent muskuloskeletal disorders (MSDs). However, physical Human-Robot Interaction (pHRI) is still largely restricted to handling, lifting, and positioning scenarios in which robots do not have the autonomy to plan their provided assistance themselves (Villani et al., 2018).

For example, consider the case of moving a large object that is too heavy or bulky to be safely and comfortably manipulated by one person. In such cases a cooperative robot providing physical assistance could take most of the weight, or at least support the object in a way that allows the human to remain in a comfortable, ergonomic posture. Non-ergonomic poses are very high on the list of causes of work-related MSDs (da Costa and Vieira, 2010), closely followed by heavy physical work and lifting. Product lifecycle management software, like Siemens Jack (Siemens PLM Software, 2019), is used in industry to optimize the ergonomics of products or processes, including collaborative robot arms (Maurice et al., 2017), in the design phase. However, such tools are typically useful for static environments or processes which do not involve any on-the-fly customization or adaptation to specific users. Incorporating human ergonomics measurements in the decision-making process of cooperative robots at run-time has the potential to improve the long-term impact on workers even when the task and physical environment may be highly dynamic or unknown in advance.

In this chapter, ergonomic optimization is applied to the task of manipulating a large object, which requires continuous interaction between the robot and the human partner. The presented ergonomic planner extends the sequential planner of Gienger et al. (2018) to select a sequence of states which is ergonomically optimal for the human partner. The new planner is applied to the task of rotating large objects (Fig. 2.1).

The contribution of this chapter is twofold: 1) We develop a model that predicts the ergonomics of a human within a human-robot collaborative task (Sec. 2.4). 2) This ergonomics predictor is integrated in sequential task planning (Sec. 2.5), resulting in a joint plan for the human and robot movements optimized for human ergonomics. Subsequently, Sec. 2.6 explains how this method is applied to our test case. The presented ergonomic planner is compared to a baseline planner which optimizes solely for a minimum time solution, without additionally optimizing for the partner ergonomics. The simulation results and user study evaluation are presented in Sec. 2.7. Our findings are concluded and discussed in Sec. 2.8. But first, related work is discussed in Sec. 2.2 and a system overview given in Sec. 2.3.

2.2. Related Work

Literature shows various research on physical human-robot cooperation (pHRC), improving ergonomic working conditions, and some steps towards integrating the two.

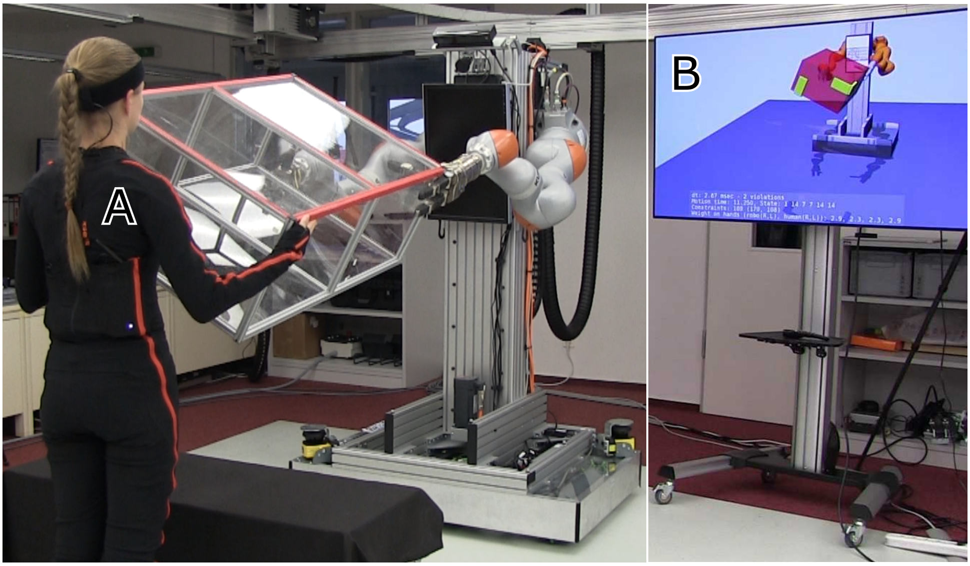

A
B
dt: 2.67 msec - 2 violations
Motion time: 11.250, State: 1 14 7 7 14 14
Constraints: 108 (170, 108)
Weight on hands (robo(R,L), human(R,L)): 2.9, 2.3, 2.3, 2.9

Figure 2.1: Robot test setup of the human-robot cooperative planner. A: The human partner wears a full-body motion capture suit (for validation). B: The screen displays the cooperative plan for both partners.

2.2.1. Physical Human-Robot Cooperation

An important area of pHRC research focuses on a human and robot jointly manipulating a single tool (Ficuciello et al., 2015; Nemec et al., 2018; Roveda et al., 2019). These problems require a form of impedance control, which allows stiffnesses to be varied to achieve improved trajectory tracking while decreasing the human task effort. This form of co-manipulation has been extended to cooperative carrying of objects, still using impedance control to manage the interaction forces between the robot and human (Agravante et al., 2019; Gribovskaya et al., 2011). Instead of using the position of the manipulator or manipulated object for impedance control, EMG signals have been successfully used as an input “intention estimate” for controlling cooperative object manipulation (DelPreto and Rus, 2019; Peternel et al., 2016).

A different form of cooperative physical interaction is observed in object hand overs. Much research in this area has focused on predicting where the human will move his/her hand (Bütepage et al., 2018) or, incorporating knowledge of a task model, classifying the intended (next) action (Hawkins et al., 2014; Maeda et al., 2017).

More complex, sequential, tasks that require regrasping during co-manipulation of objects have been addressed in Stouraitis et al. (2018), in which a planner was developed for dyadic collaborative manipulation, which includes a model of the human as an active agent who shares the task objective.

2.2.2. Optimizing Ergonomics in pHRC

Ergonomic optimization has made a recent appearance in the field of pHRC. So far two trends have been observed.

The first trend is the task of holding a workpiece in the optimal position in space while a human works on it. For example, in tasks like drilling in which a human applies a tool to an object, the position of the object held by a robot has been ergonomically optimized to minimize joint torques (Kim et al., 2018; Peternel et al., 2017), muscular effort (Marin et al., 2018) or the RULA score (McAtamney and Corlett, 1993; Shafti et al., 2019).

The second trend, in the domain of sequential tasks, is in optimizing the ergonomics for short moments of interaction during the handover of objects (Busch et al., 2018). In these cases the human pose only depends on the robot in the brief instants of the handover, and each handover pose is independent of the previous ones.

2.2.3. Ergonomic Measures

Extensive bio-mechanical models exist (e.g. AnyBody Technology A/S (2017)) which can be used to simulate human bodies to extract information that cannot be directly obtained from sensors. However, the complexity of full muskuloskeletal models makes them computationally expensive to use. In Marin et al. (2018), a full muskuloskeletal model is used to train a low dimensional latent variable model, which is then used for minimizing muscle activation in a bimanual drilling task. In Kim et al. (2018) and Peternel et al. (2017), a weighted sum of joint torques has been used as an ergonomic measure, in which the joint torques were obtained from the full-body pose combined with the estimated respectively measured center of pressure.

Many methods in practical use are based on tables and checklists (David, 2005) designed for manual evaluation of tasks by an ergonomics expert. A generally accepted and popular method for full-body evaluation, verified by ergonomic experts and easy to automate, is the Rapid Entire Body Assessment (REBA) (Hignett and McAtamney, 2000) (which is the full-body extension of RULA (McAtamney and Corlett, 1993)). REBA requires measurement of the full-body pose and estimation of the external forces acting on the body.

2.3. System Overview

The focus of this chapter is the design of a method for ergonomic planning of tasks in which both human and robot need to coordinate their movements to satisfy the constraints of the task (e.g. do not drop the object). To this end, we developed a method to estimate the ergonomic cost based on a prediction model of the human's pose and loads (Sec. 2.4). This predicted ergonomic cost is then incorporated in a sequential planner (Sec. 2.5). A schematic overview of the two components and their interaction is given in Fig. 2.2.

2.4. Ergonomics Predictor

In order to estimate the ergonomics (Sec. 2.4.1) of the hand poses, we need to predict the postures the human will use to obtain these hand poses, as well as the load that will act on the human in these postures (Fig. 2.3). For predicting the posture, we employ a learned pose predictor combined with an inverse kinematics correction to estimate the full-body pose of the human based on his/her hand poses (Sec. 2.4.2). The load is

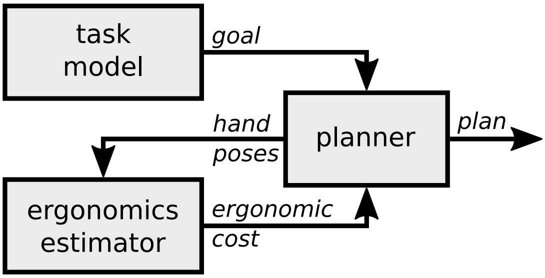


Figure 2.2: System overview: The task model provides a goal. In order to reach this goal, the planner proposes sequences of hand poses. An ergonomics estimator is developed to evaluate the ergonomic cost of these hand poses, such that minimizing this cost results in an ergonomic task solution.

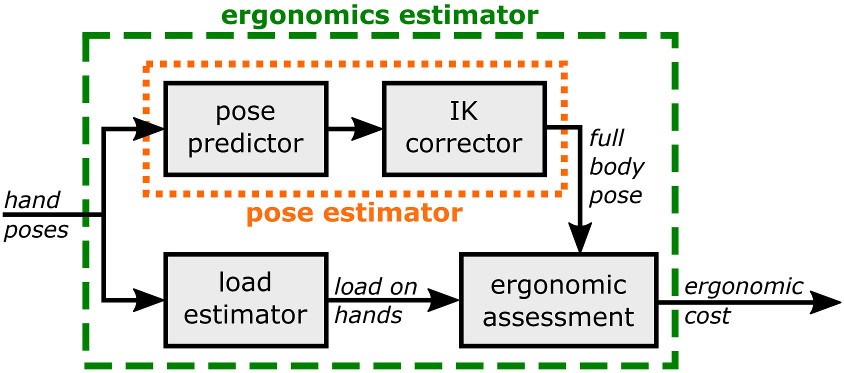


Figure 2.3: Predicting the ergonomic cost from hand poses.

estimated based on the hand poses of the human and the robot, combined with the physical properties of the manipulated object (Sec. 2.4.3).

2.4.1. Ergonomic Assessment

In this chapter, we chose REBA (Hignett and McAtamney, 2000) as the ergonomics assessment method. By using a standardized ergonomics metric, we expect that our results will be compatible with existing ergonomics practices. Even so, the metric is easily replaceable by any other measure derivable from pose and load information. The choice of another metric will likely lead to a different pose to be considered optimal. However, it will not influence the pose and load estimators and thus not impact their performance.

REBA applies a set of tables to evaluate the human posture from joint angles augmented with some additional information, e.g., external loads (Sec. 2.4.3) or whether the human is standing on one leg. The resulting score ranges from 1 (negligible risk) to a maximum of 12 (high risk: 8-10, very high risk: 11+).

The original REBA assessment considers only one active arm. For the bimanual tasks we are considering, we evaluate the arm that is the least ergonomic in order to compute the worst-case REBA score. The full-body pose score is integrated over time to evaluate and compare plans.

2.4.2. Full-Body Pose Estimator

For the ergonomic assessment, we need to infer a full-body pose given the hand poses. Mapping hand poses to a full-body pose is a redundant problem, which we chose to solve in the following two steps:

Learned Pose Predictor From Data

The pose predictor maps the two human hand poses to the human's full-body pose by means of supervised learning. Given that the training data does not always cover the whole input space, there might be small errors in these predictions.

Since the mapping from hand poses to full-body poses is not unique, we need to train a model to predict likely full-body poses. We train a separate model for each human, so we capture a personal model to predict the full-body poses each specific human typically employs. The datasets contain full-body joint angles (59 degrees of freedom) and a corresponding forward kinematic model. We train a nearest neighbor (NN) map and an LWPR model (Vijayakumar and Schaal, 2000) on the hand poses obtained by the forward kinematics model from the recorded poses. Evaluation and selection of the models is discussed in Sec. 2.6.

In any case, the hand positions and orientations of the estimated full-body pose will not exactly match the target. This is corrected in the next step:

Inverse Kinematic (IK) Pose Correction

The IK pose correction step corrects the pose such that the hands of the full-body pose match the given human hand poses. Additional constraints are applied to align the feet with the ground plane, and to keep the overall center of mass balanced. See Gienger et al. (2005) for details.

2.4.3. Hand Load Estimator

The load on each human hand is estimated based on the hand poses of the human and robot as well as physical properties (such as the mass, geometry, and friction parameters) of the manipulated object. To do so, we solve the following optimization problem: each of the hands is allowed to exert a normal force F_N and a tangential force F_T (as depicted in Fig 2.4). Static balance is assumed at all times. Two reasons support this assumption: all (allowed robot) motions are slow, and the planner only considers configurations in which it should be possible to halt the plan, for example, to wait for a confirmation to continue.

The optimization criterion is the square of the resultant forces summed over all hands in contact, which expresses the desire to hold the object as lightly as possible. Given which hands are in contact with the object, Eq. (2.1) describes the optimization problem:

$$\min \sum_{i=\text{hands\_in\_contact}} \left(F_{N,i}^2 + F_{T,i}^2\right) \tag{2.1}$$

subject to the following bounds for each of the hands:

$$-\sqrt{F^2_{\text{maxPull},i} - F^2_{\text{maxT},i}} \leq F_{N,i} \leq \sqrt{F^2_{\text{maxPush},i} - F^2_{\text{maxT},i}} \tag{2.1a}$$

$$-F_{\text{maxT},i} \leq F_{T,i} \leq F_{\text{maxT},i} \tag{2.1b}$$

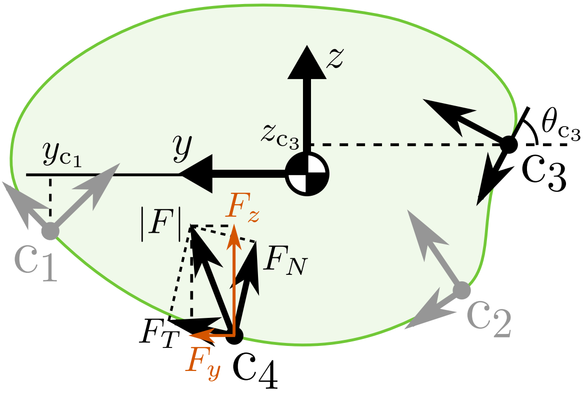


Figure 2.4: An object supported at four points, by: c_1: robot right, c_2: robot left, c_3: human right, and c_4: human left hand. At each contact point, the total force F is the vector sum of the normal force F_N, acting in direction θ, and a tangential force F_T, which can be decomposed into vectors pointing in y and z direction. The x direction is defined from the human to the robot.

and the static balance constraints,

$$\sum_i F_{x,i} = 0, \quad \sum_i F_{y,i} = 0, \quad \sum_i F_{z,i} = mg, \tag{2.1c}$$
$$\sum_i M_{x,i} = 0, \quad \sum_i M_{y,i} = 0, \quad \sum_i M_{z,i} = 0, \tag{2.1d}$$

where $F_{\text{maxPull},i}$, $F_{\text{maxPush},i}$, and $F_{\text{maxT},i}$ are the maximum pulling, pushing, and tangential forces for each of the (robot and human) hands in contact. Gravity g is acting on the object's mass m in the negative z-direction (Fig. 2.4). The solution of this optimization problem is the set of minimum hand forces required to hold the object in static balance.

2.5. ERGONOMIC PLANNER

FOR planning, the general planning architecture of Gienger et al. (2018) has been extended to include the ergonomic cost of the human partner. Given some goal state provided by the task model (Fig. 2.2), the state space is searched for a sequence of states to reach the goal with minimum cost. This plan can be converted to smooth motion trajectories for the robot's hands, and through full-body IK, motor commands for the robot hardware can be computed. For further details, the reader is referred to Gienger et al. (2018).

Given an object of known size and weight, the tasks we consider are formulated in terms of the desired position and orientation (pose) of the object. In order to plan how best to cooperatively manipulate the object to achieve this goal, the underlying state description comprises the object pose as well as the contact locations for robot and human hands. Fig. 2.5 depicts the discrete state description for the box object used in our study. However, the face of the object can have an arbitrary shape. Possible contact locations are defined evenly distributed around the graspable surface of the object, with the hand normal vector always perpendicular to the object surface. In this way, the index of the contact location suffices to describe the position and orientation of each hand.

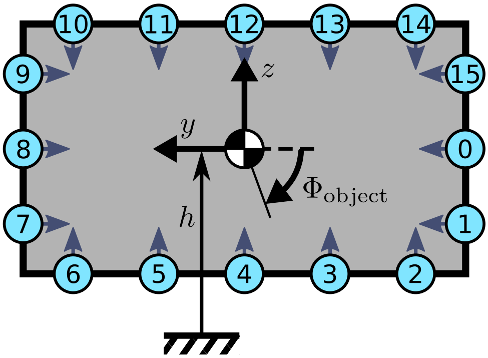


Figure 2.5: State description. The numbers in the circles enumerate the contact locations on one side of the object. Discretized height h (from ground) and angle Φ_{object} define the position and orientation of the object.

An A* graph search is applied to find a state sequence which is optimal with respect to some cost criterion. Costs accumulate as the planner explores possible next states in the sequence and each transition from one state to the next has an associated cost. In Gienger et al. (2018), this transition cost was proportional to the time the robot needed for the transition.

In the ergonomic planner, the transition cost is extended by the predicted ergonomic cost of the human partner. With the method presented in Sec. 2.4, the REBA score can be computed for each of the states proposed by the planner. It is assumed that subsequent states are close enough, and the change between them slow enough, that the properties (such as pose and load) during the transition between the states can be estimated sufficiently by linear interpolation between the enclosing states. Currently, we assume each state transition to have a fixed duration. Therefore, we assume the ergonomics of the transitions can be compared by taking the average REBA score of the enclosing states.

The human ergonomic cost term is scaled to dominate the robot cost described previously by one order of magnitude. When different possible successor states have the same cost, the planner is biased towards selecting the next state which requires the least movement. If the ergonomic cost has the same order of magnitude as the trajectory cost, the robot is less likely to plan a trajectory that involves more robot motion and takes longer to execute for the benefit of improved ergonomics for the human partner.

While not exploited in the experiments, it should be stated that our planning architecture allows for specifying an upper bound on the permissible ergonomic cost. This can easily be incorporated into the employed planner by rejecting states with an ergonomic cost larger than a given limit.

2.6. EVALUATION SCENARIO

In this chapter, the method is applied to a cooperative object rotation task in which the object is held on one side by the robot and on the other side by a human. A proof-of-concept user study with four subjects was conducted with a rectangular box of dimensions 0.63 × 0.36 × 1.0 m, weighing 10 kg (Fig. 2.1). The box is rotated in the 2D plane

Table 2.1: Model estimation errors, mean±std, for the NN and LWPR predictors with different angle weight factors.

Angle weight factor		0.0	0.1	0.2	0.3	0.4	0.5
NN	$\lvert\vec{x}\rvert$ in cm	13±8.2	13±8.2	13±7.5	14±7.6	15±8.3	15±8.7
	$\angle$ in deg	30±34	21±18	17±10	15±9.6	14±8.9	14±8.5
LWPR	$\lvert\vec{x}\rvert$ in cm	15±10	14±9.1	16±10	15±9.3	17±8.7	14±8.3
	$\angle$ in deg	39±35	36±31	30±37	22±26	26±34	16±22

around the axis pointing from the human to the robot. Translation is only allowed in the vertical direction, as specified by h. The state space is defined as in Fig. 2.5, with angles $\Phi_{\text{object}} \in \left\{0°, \Delta\Phi, \ldots, 360° - \Delta\Phi\right\}, \Delta\Phi = 30°$ and heights $h \in \{0.5, 0.5+\Delta h, \ldots, 2.0\}, \Delta h = 0.1$ in meters. This section presents the relevant implementation details to the simulation tests and user study discussed in Sec. 2.7.

2.6.1. Pose Estimation

Each participant was equipped with a motion capture suit for data collection, and was asked to move and rotate the box with a human partner. A corresponding dataset of about 1.5 min with a sampling time of 4 ms was recorded. Participants were guided to move the box through a large range of positions in the overall task space. The obtained dataset was used to train a subject-specific pose predictor which, after IK correction, provides a pose estimate that reflects the respective participant's personal pose behavior. The smaller the IK correction can be kept, the more human-like the predicted poses will be, and the more personal behavior is captured.

The pose predictor is trained to estimate the full-body pose from hand positions and orientations. A weight factor scales the importance of the orientations with respect to the positions. One set of training data (18k samples) was used to train the NN and LWPR predictors with different angle weight factors. The models were evaluated with a different dataset (25k samples) containing similar poses of the same person. The results are shown in Table 2.1.

In general, the NN prediction results in a lower mean error and smaller standard deviation compared to the LWPR. The differences are small for the position precision, but the NN predictor performs much better when it comes to predicting the correct angles of the hands. An angle weight factor of 0.3 was chosen for the subsequent experiments, trading off position and orientation.

The body pose estimation, especially the IK correction step, is too computationally expensive for the planner to run on every state evaluation. As the set of states considered during planning is discrete and finite, we generate a table which contains the full-body pose stored for every set of unique human hand states and object heights. Then, during search, only this table needs to be evaluated in order to obtain the pose for ergonomic evaluation.

Table 2.2: Parameters and constraint values for the load estimation

maximum human pulling force	$F_{\mathrm{h,maxPull}}$	0.0	N
maximum robot pulling force	$F_{\mathrm{r,maxPull}}$	0.0	N
maximum human pushing force	$F_{\mathrm{h,maxPush}}$	98.1	N
maximum robot pushing force	$F_{\mathrm{r,maxPush}}$	137.3	N
friction coefficient human	μ_{h}	1.1	
friction coefficient robot	μ_{r}	1.1	
object (box) mass	m	10.0	kg

2.6.2. Load Estimation

Currently, we disallow grasping (and hence pulling). For the robot this will always be the case, as it is not able to grasp the objects. The people in the study were instructed not to grasp, only support, the object. Thus, all forces in x direction, perpendicular to the object face (Fig. 2.4), are assumed to be zero. Therefore, the maximum allowed tangential forces depend on the normal forces and the friction coefficients, i.e., $F_{\mathrm{maxT},i} = \mu_i |F_{N,i}|$, with the friction coefficients μ_i.

The parameter values used in the presented cases are listed in Table 2.2. Since REBA considers a maximum load of 10 kg, this was set as the maximum allowed human pushing load. The robot arms have a maximum load specification of 14 kg. The friction coefficients are an estimate based on the friction coefficients of combinations of materials the human and robot hands and the box are made of.

As for the full-body poses, the loads are precomputed and stored in a table for fast retrieval during planning.

2.7. Experimental Results

The ergonomic planner is tested in a small user study of four people (1 female, 3 male), ranging in height between 1.70 and 1.95 m and in BMI between 20 and 27 $\mathrm{kg\,m^{-2}}$, on the task of collaboratively rotating the box 180°, clockwise and counterclockwise. The subjects were specifically chosen for their differences in size and build in order to prove the concept. The study was approved by the Ethics Committee of the Honda Research Institute Europe on 03/09/2019.

The ergonomic performance of the planner is compared to that of the planner without the ergonomic cost term. In order to evaluate the quality of the ergonomic planner, the participants need to follow the robot's collaborative plan for them. The plan is displayed on a large screen next to the robot.

The planner is evaluated in simulation and in an experimental setup with the robot depicted in Fig. 2.1. During the experiments, the participants wear an Xsens motion capture suit (Roetenberg et al., 2009) to measure their poses. The REBA scores are computed from the predicted, respectively measured, poses combined with the estimated loads. Measurements of the robot hand forces show an average force estimation error of 0.4 N, with a standard deviation of 9.8 N. Compared the average predicted force of 25.9 N, this is accurate enough for our purposes.

Fig. 2.6 shows seven states out of a sequence proposed by the ergonomic (top row)

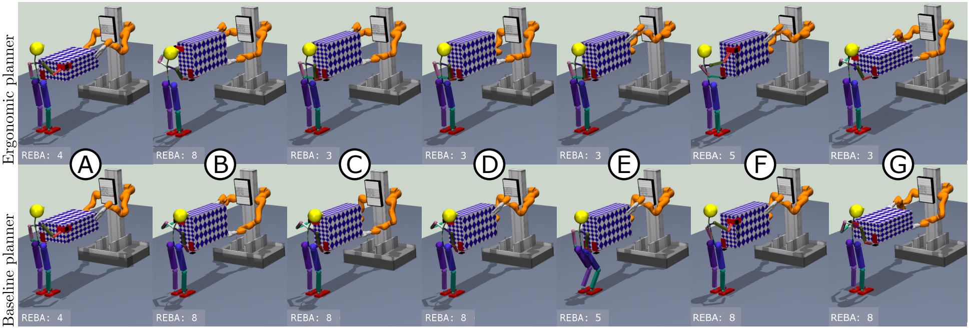


Figure 2.6: Ergonomic planner (top) versus baseline planner (bottom) showing seven states of the planned sequence for rotating the box +180°. At the bottom of each subfigure the REBA score is printed. The states correspond to the dashed lines in Fig. 2.7.

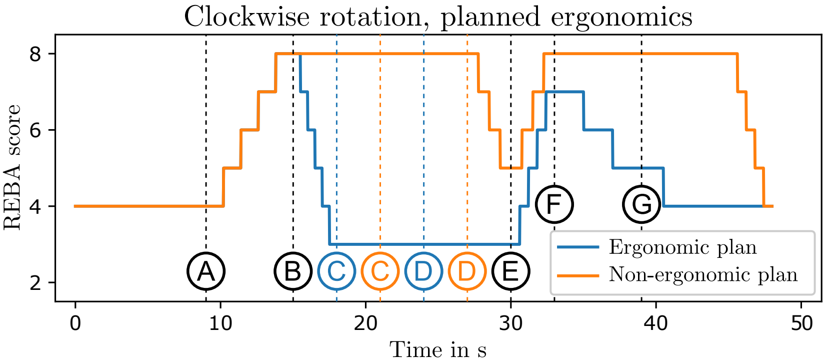


Figure 2.7: Predicted ergonomic scores for the ergonomic and the baseline planner for rotating the box +180°. The dashed lines correspond to the states depicted in Fig. 2.6 is reached. In case of states C and D the two planners differ in when the state is reached (see lines in corresponding colors).

and baseline (bottom row) planners for the goal of rotating the box 180°. The original planner just tries to minimize the time to task completion. As a result, the height of the object is held constant and, by default, the robot always regrasps first. For a fair comparison between the two planners, the height of the object in the initial state is the ergonomically optimal height of the starting pose according to the human pose model.

The ergonomic planner adjusts the height to optimize ergonomics. Who regrasps first also depends on what is most ergonomic for the human. This can be observed in states C and D in Fig. 2.6. The baseline planner requires the human to stay in a very unergonomic pose until the robot has regrasped twice, while the ergonomic planner allows the human to regrasp to a more ergonomic pose as quickly as possible and to stay in the more ergonomic pose as long as possible.

Figures 2.7 and 2.8 show the predicted, respectively the measured, REBA scores from the ergonomic and baseline planners. The dashed lines indicate the seven states corresponding to the snapshots in Fig. 2.6. Due to safety reasons, the robot takes 6.0 s for regrasping or rotating the object. Height adjustments can be performed much faster and

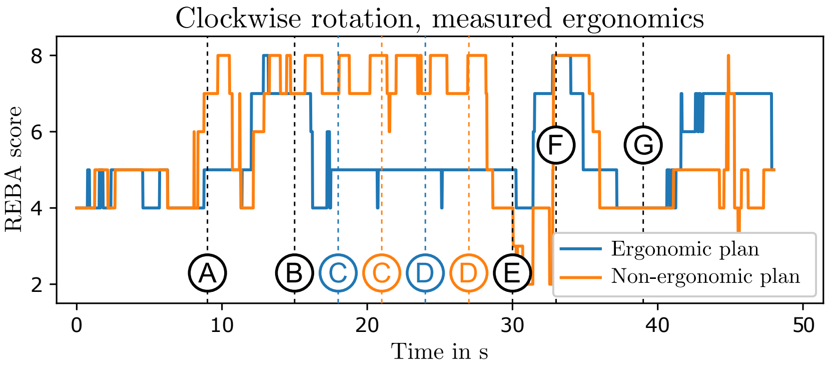


Figure 2.8: Measured ergonomic scores for the ergonomic and the baseline planner for rotating the box +180°. As in Fig. 2.7, the dashed lines correspond to the states depicted in Fig. 2.6.

Table 2.3: Predicted and measured REBA scores of the ergonomic and baseline planner for rotating a 10 kg object by 180°, mean±std for four different people and two rotation tasks each (clockwise and counterclockwise rotation).

		Average REBA	Maximum REBA	% of time with REBA ≥ 8
Planned	ergonomic planner	4.6±0.2	6.5±1.1	0.9±1.5
	baseline planner	6.0±0.7	7.8±0.4	25.2±24.2
Measured	ergonomic planner	5.1±1.2	7.9±0.9	5.9±6.3
	baseline planner	5.5±1.6	8.0±1.3	17.7±18.0

when it is the human's turn to regrasp, a 3.0 s transition time is much more comfortable for the human. With a longer transition time, the humans would need to take active care not to be too fast. This results in some state changes in the ergonomic planner occurring at a different time than in the baseline planner.

Fig. 2.7 shows lower REBA scores for the ergonomic planner whenever a more ergonomic alternative can be found. The results in Fig. 2.8 also generally show a REBA score below the score associated with the baseline planner. Up to about 35 s, the predicted REBA scores are reflected in the measurements, except for the ergonomic plan not actually getting the REBA score as low as 3. Around 30 s and in the last 7 s, the baseline plan was executed surprisingly ergonomically, resulting locally in a lower REBA score for the baseline planner.

The ergonomic plan differs from person to person, which indicates the planner accounts for subject-specific differences. Table 2.3 lists the combined results of the planners for all participants of the user study, each of whom rotates the box clockwise and counterclockwise, once in simulation and once together with the physical robot. Averaged over all generated plans, the predicted average REBA score is clearly lower for the ergonomic planner compared to the baseline planner, and the standard deviation is smaller. In a single case, the predicted maximum REBA score was as high for the ergonomic planner as for the baseline planner. This is the case when no more ergonomic alternative is known by the pose estimator. Except for one single case (state B in Fig. 2.6

and Fig. 2.7), the ergonomic planner could avoid plans including poses in the "High Risk" category, i.e., REBA ≥ 8.

The REBA scores observed during the experiments show smaller differences between the two planners. Though large differences were observed between participants, in general the ergonomic planner still yields a lower ergonomic cost, and the amount of time spent in poses with a 'high' ergonomic risk is reduced considerably. In all cases the standard deviation is lower for the ergonomic planner.

2.8. Conclusion and Discussion

This chapter presents a novel concept for computing optimal ergonomics-enhanced plans in cooperative physical human-robot interaction tasks. The first contribution is a novel human model which allows for prediction of an ergonomic assessment corresponding to a state within a task. It consists of a learned pose model and a computational load model. The pose model is trained with human motion capture data in order to predict the human pose as realistically as possible. The load model assumes some prior knowledge of the task, such as the mass and geometry of the manipulated object. Given a state within the task, the pose and load models provide the human pose and corresponding interaction forces for calculating a corresponding ergonomics score. Our prediction model gives a subject-specific estimate of the ergonomics of the states within a task.

The second contribution is the integration of this prediction model with a planning algorithm. The presented planner incorporates states and actions for both robot and human. This allows the computation of sequential plans optimized for human ergonomics. It is also possible to compute plans with a guaranteed upper bound on the permissible ergonomics score.

We have shown in simulation and robot experiments of a collaborative human-robot box-rotating task that the proposed concepts lead to improved human ergonomics, within the presented experiment with 4 subjects. Although the results look promising, further testing with a larger group of users is necessary. Also, evaluation of the proposed ergonomics-enhanced planner in more complex collaborative tasks remains for future work.

For the conducted study, the simple pose predictor sufficed. However, the errors corrected by the IK are currently around 14 cm and 15°, which definitely leaves room for improvement.

In this chapter, we selected the REBA score for ergonomic assessment. However, the system is flexible enough to allow for incorporating other ergonomics indicators. For future use of the proposed method, exploration into the effects of incorporating alternate ergonomics indicators is recommended, as the discrete nature of REBA sometimes causes a large difference in ergonomic cost for very small posture changes. This, together with the small number of study participants, contributes to the large standard deviations observed in the results.

This chapter demonstrates our approach to be capable of finding a plan which affords improved ergonomics for people working with a robot. After additional verification, future work will focus on concepts to encourage the human to follow the ergonomic plan, and react appropriately when the human does not.

3

An Incremental Inverse Reinforcement Learning Approach for Motion Planning with Separated Path and Velocity Preferences

Humans often demonstrate diverse behaviors due to their personal preferences, for instance, related to their individual execution style or personal margin for safety. In this chapter, we consider the problem of integrating both path and velocity preferences into trajectory planning for robotic manipulators. We first learn reward functions that represent the user path and velocity preferences from kinesthetic demonstration. We then optimize the trajectory in two steps: first the path and then the velocity, to produce trajectories that adhere to both task requirements and user preferences. We design a set of parameterized features that capture the fundamental preferences in a pick-and-place type of object-transportation task, both in shape and timing of the motion. We demonstrate that our method is capable of generalizing such preferences to new scenarios. We implement our algorithm on a Franka Emika 7-DoF robot arm, and validate the functionality and flexibility of our approach in a user study. The results show that non-expert users are able to teach the robot their preferences with just a few iterations of feedback.

The contents of this chapter have been slightly modified from the paper of the same name, by Armin Avaei*, Linda van der Spaa*, Luka Peternel, and Jens Kober, published in *MDPI Robotics, Vol. 12, No. 2, 2023.*

*These authors contributed equally to this work.

The work in this chapter received partial additional support by the European Research Council Starting Grant TERI "Teaching Robots Interactively" (project reference 804907) and the European Space Agency through the project "Rhizome: Off-Earth Manufacturing and Construction".

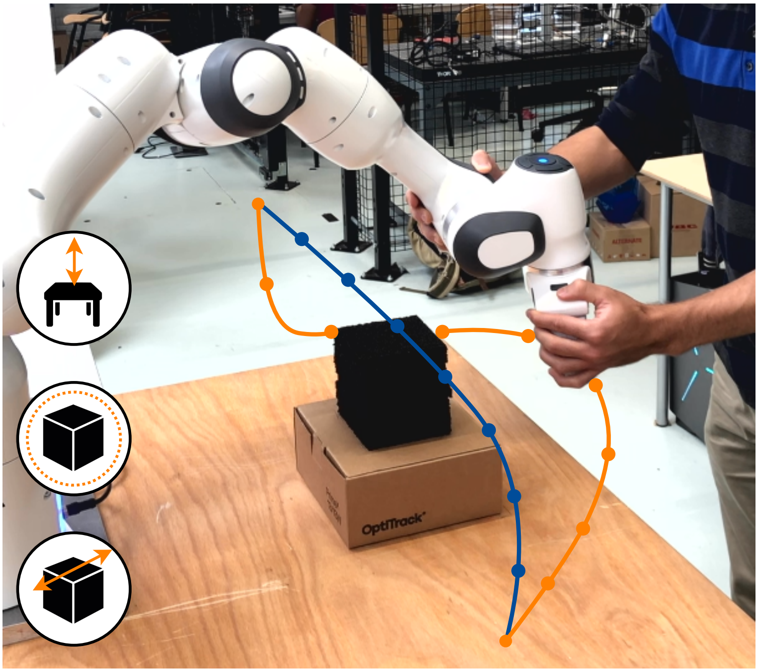


Figure 3.1: Leveraging demonstrations as means of understanding the human's preferences in an object-carrying task: The robot originally plans the blue trajectory without knowledge of human preferences. The user demonstrates the orange trajectory which in this instance contains the following preferences: "Stay close to the table surface", "Keep larger distance from the obstacle", and "Pass on the far side of the obstacle". We develop a method for learning and generalizing such preferences to new scenarios (i.e., new start, goal or obstacle positions).

3.1. Introduction

Autonomy is increasingly being discussed under the aspect of cooperation. A gentler breed of robots, "cobots", have started to appear in factories, workshops and construction sites, working together with humans. A challenge in the deployment of such robots is producing desirable trajectories for object-carrying tasks. A desirable trajectory not only meets the task constraints (e.g. collision-free movement from start to goal), but also adheres to user preferences. Such preferences may vary between users, environments and tasks. It is infeasible to manually encode them without exact knowledge of how, with whom, and where the robot is being deployed (Jain et al., 2015). Manual programming is even more detrimental in cooperative environments, where robots are required to be easily and rapidly reprogrammed. In this context, learning preferences directly from humans emerges as an attractive solution.

We address the challenge of learning personalized human preferences, starting from a robot plan that may not match the execution style or safety standards of a specific human user (e.g. robot carries the object closer to the obstacle than the user prefers). Fig. 3.1 illustrates how a user may demonstrate a trajectory encoding multiple implicit preferences to correct the original robot plan.

One way to adhere to human preferences is by means of variable impedance control (Duchaine and Gosselin, 2007; Peternel et al., 2014). While such strategies can ensure safe and responsive adaptation, they suffer from being purely reactive (i.e., they do not remember the corrections). The robot should not only conform to a new trajectory, but it has to update its internal model in order to understand the improvement of the corrected trajectory (Bajcsy et al., 2017; Losey et al., 2022; Losey and O'Malley, 2019). Thus, ideally, we should encode knowledge of humans' desired trajectories as a set of parameters that are incrementally updated based on the corrected trajectory.

To this end, Learning from Demonstration (LfD) approach enables robots to encode human-demonstrated trajectories. LfD frameworks have the advantage of enabling non-experts to naturally teach trajectories to robots. A widespread trajectory learning method in LfD is Dynamic Movement Primitives (DMPs) (Ijspeert et al., 2002). In addition to encoding trajectories, DMPs are able to adapt the learned path by updating an interactive term in the model (Gams et al., 2016; Kulvicius et al., 2013). Additionally, they can adapt the velocity of the motion by estimating the frequency and the phase of a periodic task (Peternel et al., 2014), or learning a speed scaling factor (Nemec et al., 2018). As a result, DMPs can capture human path and velocity preferences on a trajectory level. Losey and O'Malley (2019) demonstrated that such velocity preferences can also be learned online, from interactive feedback, although with some effort. However, these methods lack any knowledge about the task context or why the trajectory was adjusted in the first place. Hence, such an approach fails to generalize user preferences to new scenarios of the same task due to the lack of a higher-level understanding of human actions. We show this in the supplementary comparison study presented in Sec. 3.4.

A better approach is to pair parameters with features that capture contextual information (e.g. distance to obstacle), and utilizes this information to find an optimal solution in new scenarios. Such generalization can be achieved by learning a model of what makes a trajectory desirable. Modeling assumptions can be made to form a conditional probability distribution over trajectories and contextual information, e.g. as in Ewerton et al. (2016). While proven effective in simple reaching tasks, whether such models can directly capture complex human preferences in a contextually rich environment remains an open question. However, Inverse Reinforcement Learning (IRL) approaches have already proven to be capable of this (Wirth et al., 2017).

Unlike traditional IRL methods requiring expert demonstrations (Ratliff et al., 2006; Ziebart et al., 2008), more recently derived algorithms allow preference learning from user comparisons of sub-optimal trajectories (Wirth et al., 2017). Potentially, a much wider range of human behavior can be interpreted as feedback for preference learning in general (Jeon et al., 2020). In this chapter, however, we focus on reward learning for robot trajectories. A model-free approach can be used to learn complex nonlinear reward functions (Ibarz et al., 2018), but such an approach requires many queries to learn from, which is time-intensive. Therefore, we keep a simple linear reward structure. To shape this reward, we identified four fundamental preference features of the pick-and-place type of object-transportation tasks in the literature: height from table/ground (Bajcsy et al., 2017; Jain et al., 2015; Losey et al., 2022), distance to obstacle (Bıyık et al., 2022; Jain et al., 2015), obstacle side (Kirby et al., 2009; Kretzschmar et al., 2016), and velocity (Nemec et al., 2018; Peternel et al., 2014). These features are relatively scenariounspecific, and therefore suitable for generalization in object-transportation tasks of the kind we consider in this chapter: pick-and-place tasks in the presence of obstacles. To the best of our knowledge, there is no method to account for all these features together in a unified framework.

Given such a set of features, coactive learning (Shivaswamy and Joachims, 2015) can be used to learn a reward function. In coactive learning, the learner and the teacher both play an active role in the learning process: the learner proposes one or multiple solutions and learns from relative feedback provided by the teacher in response. Coactive learning has an upper boundary on regret, leaving room for noisy and imperfect user feedback. Furthermore, it is an online algorithm, i.e., the system can learn incrementally from sequential feedback. An adapted version of coactive learning was applied in Jain et al. (2015) to learn trajectory preferences in object-carrying tasks. To this end, users iteratively ranked trajectories proposed by the system. Although selected based on the learned reward, the trajectories were generated using randomized sampling, which increases the number of feedback iterations necessary for convergence. Methods in Bajcsy et al. (2017) and Losey et al. (2022) adapt the robot trajectory to a user's preferences based on force feedback and optimize the remaining trajectory with online correction in a specific scenario. However, these methods cannot capture velocity preferences on top of path preferences.

To address this gap in the state-of-the-art, we propose a novel framework for optimizing trajectories in object-transportation tasks that meet the user's path and velocity preferences, where we first optimize the path and then the velocity on the path. The objective function for the optimization comprises a human preferences reward function and a robot objective function that ensures the safety and efficiency of the trajectories. This explicit separation of the agents' objectives allows for negotiation, where the robot is recognized as an intelligent agent which may give valuable input of its own.

The approach takes a full demonstrated trajectory as the feedback for the learning model, comparing it to the robot's previous plan at each iteration. A minimum acceleration trajectory model significantly reduces the size of the task space, hence increasing the optimization efficiency. To capture the preferences, we design a set of features that correspond to the four preferences, covering both the motion shape and timing, which we identified from the literature to be fundamental for the considered pick-and-place tasks.

Unlike Bajcsy et al. (2017) and Losey et al. (2022), we request iterative feedback and employ an optimization scheme that samples from the global trajectory space. While this is less efficient in terms of human effort for teaching preferences in a specific scenario (i.e., the user has to provide at least one full task demonstration), it allows us to additionally capture velocity preferences on top of the path preferences. Furthermore, our method enables the separation of velocity and path preferences both during the learning and in the trajectory optimization stage. With our combination of a trajectory optimization scheme and carefully selected preference features, we can generalize to new contexts without needing (many) additional corrective demonstrations. We show this in our user study. In contrast to Jain et al. (2015), we learn from a few informative feedback demonstrations, and give special attention to the trajectory sampling by employing model-based trajectory optimization. This facilitates fast learning and generalization of preferences to entirely new contexts.

We evaluate the proposed method in a user study on a 7-DoF Franka Emika robot arm. In the key previous user studies of learning human preferences (Bajcsy et al., 2017; Losey et al., 2022; Palan et al., 2019), the experimenter instructed the human participants what preference to demonstrate to the robot. Differently, in our user study, we let the participants freely select their own preferences while demonstrating the task execution to the robot. Additionally, our study examines whether the users can actually distinguish the learned trajectory capturing their preference from the trajectories capturing only part of their preference. In a supplementary study, we qualitatively compare our method to two relevant methods from the literature. We discuss the structural differences between the methods, and show in simulation how these differences affect the learning of preferences from human (corrective) demonstrations.

In summary, this chapter's main contribution is a methodology that is able to capture velocity preferences on top of path preferences by separating the velocity optimization from the path optimization. Learning the path and velocity separately provides users with the option to avoid the challenge of providing a temporally consistent demonstration at each iteration. This offers users the flexibility to demonstrate their path and velocity preferences either simultaneously or in separate demonstrations. Secondly, the learned preferences are transferred to new scenarios by exploiting a trajectory model. Importantly, we perform a user study to validate whether the proposed method can learn and generalize freely chosen preferences, in contrast to the many user studies in the literature which prescribe user preferences. Additionally, we perform a supplementary study to compare pros and cons of the proposed approach to two common methods from the literature.

The rest of the chapter is organized as follows: In Sec. 3.2, we explain the algorithm and methodology in detail. The user study is described in Sec. 3.3, and the experimental results are shown and discussed. A supplementary study is presented and discussed in Sec. 3.4. Finally, we present our conclusion and view on future work in Sec. 3.5.

3.2. METHOD

THE problem is defined in the following manner: given a context $\mathscr{C}$ describing start, goal, and obstacle positions, the robot has to determine the trajectory $\boldsymbol{\xi} = [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_N] \in \Xi$ (set of state sequences) that conforms to the human preferences and meets the task goals. The states are defined as $\boldsymbol{s}_k = [\boldsymbol{x}_k; \dot{\boldsymbol{x}}_k]$ (position and velocity), with k indicating trajectory samples.

In our setting, the true reward functions are known by the user but not directly observable by the robot. Hence the problem can be seen as a Partially Observable Markov Decision Process (POMDP) (Bajcsy et al., 2017). Our reward functions have parameters that are part of the hidden state, and the trajectories provided by the user are observations about these parameters. Solving such problems, where the control space is very complex and high-dimensional, is challenging. Therefore, we simplify the problem through approximation of the policy by separating planning and control, and treating it as an optimization problem. Furthermore, we make the problem tractable by reducing our state space to one of viable smooth trajectories.

The resulting framework, depicted in Fig. 3.2, first learns the appropriate reward

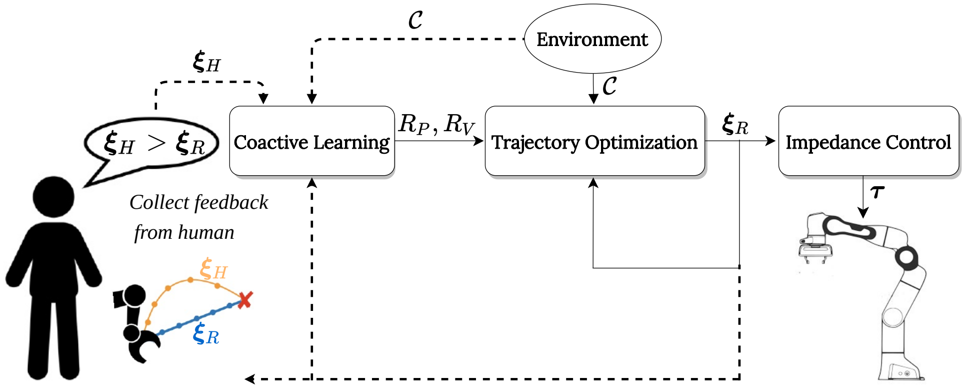


Figure 3.2: The human user provides demonstrations $\boldsymbol{\xi}_H$, which are used to learn a distribution over reward functions via coactive learning given a context $\mathscr{C}$. We use the learned rewards, R_P for the position and R_V for the velocity, to optimize the robot's trajectory according to the human preferences. The resulting trajectory $\boldsymbol{\xi}_R$ is executed using an impedance controller, which sets the robot joint torques $\boldsymbol{\tau}$. We repeat this process, querying the human for preferred trajectories until convergence. The human can then be taken out of the loop.

functions, then plans a trajectory maximizing the rewards via optimization. Once the trajectory is defined, we use impedance control to track it in a safe manner. Notably, we separate the problem of path and velocity planning in the learning and optimization steps. Updating the path and velocity weights separately provides users with the option to avoid the challenge of providing a temporally consistent demonstration at each iteration. As a result, users have the flexibility to demonstrate their path and velocity preferences either simultaneously or in separate demonstrations.

3.2.1. Learning Human Reward Functions from Demonstration

We follow previous IRL work (Jain et al., 2015; Ratliff et al., 2006) in assuming that the reward functions are a linear combination of features ϕ with weights θ. Accordingly, we define path and velocity reward functions R_P and R_V as

$$R_P(\boldsymbol{x};\mathscr{C},\boldsymbol{\theta}_{HP}) = \boldsymbol{\theta}_{HP}^T \boldsymbol{\Phi}_P(\boldsymbol{x};\mathscr{C}), \tag{3.1a}$$

$$R_V(\bar{\dot{x}},\bar{\boldsymbol{x}};\mathscr{C},\boldsymbol{\theta}_{HV}) = \boldsymbol{\theta}_{HV}^T \boldsymbol{\Phi}_V(\bar{\dot{x}},\bar{\boldsymbol{x}};\mathscr{C}), \tag{3.1b}$$

where $\boldsymbol{\theta}_{HP}$ and $\boldsymbol{\theta}_{HV}$ denote the unknown weights that respectively capture the human path and velocity preferences. In case of the velocity reward, we divide the trajectory into equal segments (i.e., range of samples) indicated by r. Then, $\bar{\boldsymbol{x}}_r$ and $\bar{\dot{x}}_r$ are the average of the position vectors and the velocity norms in a segment. $\boldsymbol{\Phi}_P$ and $\boldsymbol{\Phi}_V$ are the total path and velocity feature counts along the trajectory:

$$\boldsymbol{\Phi}_P(\boldsymbol{x};\mathscr{C}) = \sum_{k=1}^{N} \boldsymbol{\phi}_P(\boldsymbol{x}_k;\mathscr{C}),\ \boldsymbol{\Phi}_V(\bar{\dot{x}},\bar{\boldsymbol{x}};\mathscr{C}) = \sum_{r=1}^{M} \boldsymbol{\phi}_V\left(\bar{\dot{x}}_r,\bar{\boldsymbol{x}}_r;\mathscr{C}\right). \tag{3.2}$$

Note that the velocity features are a function of both the segment's velocity and position, allowing us to capture position-dependent velocity preferences.

To have comparable rewards, all trajectories are re-sampled to contain a fixed number of N states. The velocity inherently affects the number of samples within a trajectory, which is why we divide the trajectory into M segments and consider the average velocity within each segment ($M < N$). Features are directly computed from the robot state and context of the task. We describe them in the next subsection.

During kinesthetic demonstration, the robot is in gravity compensation mode. That gives the human full control over the demonstrated trajectories, which we assume to correlate exponentially to the human's internal reward, following the Maximum Entropy assumptions (Ziebart et al., 2008) and Boltzmann's principle of rationality, as in Bajcsy et al. (2017):

$$P(\boldsymbol{\xi}_H | \mathscr{C}, \boldsymbol{\theta}_{HP}, \boldsymbol{\theta}_{HV}) \propto e^{\boldsymbol{\theta}_{HP}^T \boldsymbol{\Phi}_P(\boldsymbol{\xi}_H;\mathscr{C}) + \boldsymbol{\theta}_{HV}^T \boldsymbol{\Phi}_V(\boldsymbol{\xi}_H;\mathscr{C})}, \tag{3.3}$$

which, for brevity, we can write as $P(\boldsymbol{\xi}_H | \mathscr{C}, \boldsymbol{\theta}_H) \propto e^{\boldsymbol{\theta}_H^T \boldsymbol{\Phi}(\boldsymbol{\xi}_H;\mathscr{C})}$.

Assuming that the human behavior is approximately optimal with respect to the true reward (i.e., their preferences), we use a variant of coactive learning introduced in Bajcsy et al. (2017) to learn the weights $\boldsymbol{\theta}_{HP}, \boldsymbol{\theta}_{HV}$. However, we can only compute our $\boldsymbol{\Phi}_P, \boldsymbol{\Phi}_V$ (3.2) over full trajectories. Therefore, instead of updating the weights based on an estimate of the human's intended trajectory from physical interaction, we use a full kinesthetic trajectory demonstration by the human after each task execution to update the sum of the features over the trajectory (3.2). This results in the following incremental update rule:

$$\boldsymbol{\theta}_H^{i+1} = \boldsymbol{\theta}_H^i + \alpha \left(\boldsymbol{\Phi}\left(\boldsymbol{\xi}_H^i; \mathscr{C}\right) - \boldsymbol{\Phi}\left(\boldsymbol{\xi}_R^i; \mathscr{C}\right) \right), \tag{3.4}$$

at iteration i, with learning rate $\alpha \in (0, 1]$. Intuitively, the update rule is a gradient that shifts the weights in the direction of the human's observed feature count. It should be noted that we update the path preferences only using the position part of the state, and the velocity preferences are updated depending on where in space the velocities were observed.

3.2.2. Features and Rewards

We define the objective function for trajectory optimization as a combination of human rewards and robot objectives. The human rewards consist of features that capture human preferences (3.1), whereas the robot objectives define a basic behavior for the robot. Moreover, the robot objectives counterbalance the effect of the human rewards in the optimization. While we learn the weights in the human rewards (Sec. 3.2.1). The weights in the robot objectives are hand-tuned. In this section, we first describe the features associated with the human rewards, and then the robot objectives.

The human preferences are captured via the four features listed below (see Fig. 3.1 for an example of the listed path preferences). We chose these features as they characterize dominant behaviors in manipulation applications that depend on user preferences. Additionally, the features cover the different dimensions of the workspace (in space and time), creating a complete definition of motion behavior.

Height from the Table: The preferred height from the table, on a range of 'low' to 'high' is captured by the sigmoid function $\phi_h = \frac{1}{1+e^{-\lambda(h+p)}}$ with h indicating the vertical distance from the table, p the center of the function (an arbitrary 'medium' height above the table), and λ the parameter defining the shape of the function. The choice of a sigmoid function is to hinder the effect of this preference when close to upper and lower boundaries during the weight update (e.g., a demonstration at 75 cm above the table should not impact the weight update very differently from a demonstration at 70 cm). The decreasing slope at the boundaries additionally allows other objectives to have a higher impact on the trajectory in such regions during the optimization.

Distance to the Obstacle: We encode the user's preferred distance to the obstacle, on a range of 'close' to 'far' using the exponential feature $\phi_d = e^{-\beta d^2}$, where d is the Euclidean distance to the center of the obstacle, and β is the shape parameter. This exponential function gradually drops to 0 at a certain distance from the obstacle. This distance is a threshold outside which the local behavior of the optimization is no longer affected by the distance to the obstacle. Importantly, if a negative weight is learned associated with this feature, the trajectory is still attracted towards the obstacle even if the initial trajectory lies outside of this threshold. This is because our optimization strategy globally explores different regions of the workspace, and in this case it would detect that there is a reward associated with being closer to the obstacle.

Obstacle Side: We define this feature on a range of 'close' (the side of the obstacle closer to the robot) to 'far' (the side of the obstacle far from the robot) via the tangent hyperbolic function $\phi_s = \frac{2}{1+e^{\gamma S}} - 1$. Here, S is the lateral distance between a trajectory sample and the vertical plane at the center of the obstacle, and γ is a shape parameter. This symmetric function is designed to have a large span in order to be active in all regions of the workspace. However, as the gradient of this function decreases at larger lateral distances, so does the influence of this function in the local trajectory optimization.

Velocity: To capture the user's velocity preferences, we adopt a different approach using a discretized linear combination of uniformly distributed Radial Basis Functions (RBFs) in a range $[\dot{x}_{\min}, \dot{x}_{\max}]$. For each segment r, we map the average velocity norm $\bar{\dot{x}}_r$ onto these RBFs, given by:

$$\psi_j\left(\bar{\dot{x}}_r\right) = e^{-\left(\varepsilon \bar{\dot{x}}_r - c_j\right)^2}, \tag{3.5}$$

where shape variable ε defines the width, and c_j defines the center of the j^{th} RBF, with $j = 1, 2, \ldots, n$ (we use n=9).

Inspired by Fahad et al. (2018), we discretize the above feature to two bins, based on the distance d_r of each segment center to the obstacle. Hence, we have two cumulative feature vectors: $\mathbf{\Phi}_{V1}$ for $d_r \in [0, d_c)$, and $\mathbf{\Phi}_{V2}$ for $d_r \in [d_c, \infty)$. This allows us to approximate the speed of motion separately in areas considered to be respectively 'close' to or 'far' from the obstacle based on the distance threshold d_c (obtained from demonstration data). This way, we capture velocity preferences relative to the obstacle position. Similarly, features can be defined relative to other context parameters, to capture velocity preferences that depend on other parameterized positions.

However, the issue might arise that the two trajectories do not have the same number of segments in each distance bin. In such a case, we employ feature imputation using the mean of the available values.

The robot's objectives are composed of the following:

Path Efficiency Reward: We calculate the total length of a trajectory, which we use as a negative reward. Penalizing the trajectory length is essential in counterbalancing the human preference features in the optimization process. Essentially, it pulls the trajectories towards the straight line path from start to goal and rewards keeping them short.

Collision Avoidance Reward: We use the obstacle cost as formulated by Zucker et al. (2013), which increases exponentially once the distance to the obstacle drops below a threshold. The negative cost is our reward.

Robot Velocity Reward: This reward achieves a low and safe velocity in absence of human velocity preferences and is defined based on (3.5). In IRL, it is beneficial to learn how people balance other features against a default reward (Vasquez et al., 2014).

3.2.3. MOTION PLANNING VIA TRAJECTORY OPTIMIZATION

We discuss the problem of motion planning in two parts. First, we address the optimization of the path of the trajectory in the workspace. We then address the optimization of the velocity along this path, defining the timing of the motion.

Solving the path optimization problem over the Cartesian task-space would be complex and inefficient. Instead, we employ a trajectory planning algorithm (MathWorks, 2018) that interpolates between waypoints with piecewise clothoid curves. This algorithm minimizes the acceleration which results in a smooth and realistic motion. We exploit this algorithm to significantly reduce the search space for the path optimization, and sample trajectories using a vector of waypoint coordinates $\mathbf{p}$ and its corresponding time vector $\mathbf{t}_P$, $\boldsymbol{\xi} = f(\mathbf{p}, \mathbf{t}_P)$.

We consider three waypoints $\mathbf{p} = [\mathbf{p}^s; \mathbf{p}^m; \mathbf{p}^g]$, corresponding respectively to the start position, an arbitrary position within the path, and the goal position. We further simplify the problem by fixing the time vector to $\mathbf{t}_P = t^g [0; \frac{D(\mathbf{p}^m)}{D(\mathbf{p}^g)}; 1]^T$, where $D(\cdot)$ indicates the Euclidean distance of a waypoint to $\mathbf{p}^s$, and t^g is the time, just for the path optimization, we assume all trajectories take to finish[1]. An uneven distribution of waypoints would bias the reward value. Setting up the time vector in this manner ensures a constant velocity throughout the trajectory, which results in an even distribution of samples over the path. Trajectories can then be sampled only as a function of waypoint positions $\boldsymbol{\xi} = f(\mathbf{p})$.

We then solve for the optimal waypoint vector $\mathbf{p}^*$ using the following nonlinear program formulation:

$$\begin{aligned} \mathbf{p}^* &= \arg\max_{\mathbf{p}} \Big(R_P(\mathbf{p}; \mathscr{C}, \boldsymbol{\theta}_{HP}) + \boldsymbol{\theta}_{RP}^T \boldsymbol{\Phi}_{RP}(\mathbf{p}; \mathscr{C}) \Big), \\ &\text{subject to: } \boldsymbol{h}(\mathbf{p}) = \mathbf{0},\ \mathbf{p}_{\text{low}} \leq \mathbf{p} \leq \mathbf{p}_{\text{upp}}. \end{aligned} \tag{3.6}$$

Here, the objective function consists of the human path reward R_P and the robot's path objective, which is a linear combination of predetermined weights $\boldsymbol{\theta}_{RP}$ and the aforementioned path reward functions $\boldsymbol{\Phi}_{RP}$. The equality constraint ensures the start and goal positions are met. As a result, we are effectively searching for the waypoint $\mathbf{p}^m$ that maximizes the objective function. The upper and lower boundaries $\mathbf{p}_{\text{low}}$ and $\mathbf{p}_{\text{upp}}$ limit the trajectory to stay within the robot's workspace. Once $\mathbf{p}^*$ is found, we construct the full trajectory using $\boldsymbol{\xi}_P^* = f(\mathbf{p}^*, \mathbf{t}_P)$. Fig. 3.3 shows an example of the convergence of the optimizer towards a path that adheres to 'low height', 'close side', and 'close to obstacle' preferences.

[1] The shape of the paths is not affected by t^g in the time ranges of our manipulations, therefore we assume the path to be independent of velocity.

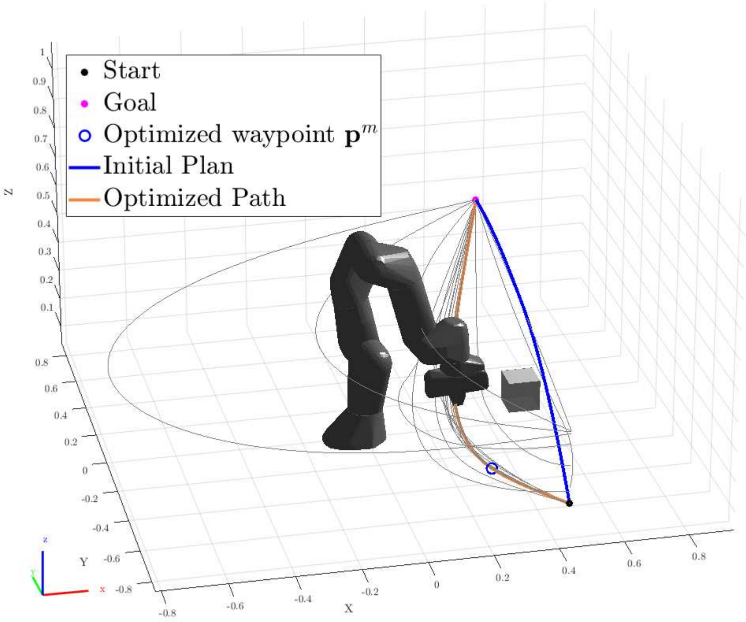


Figure 3.3: An example of convergence towards the optimal path. The optimizer places $\mathbf{p}^m$ in different locations in the workspace to generate different paths. The paths explored by the optimizer are indicated in gray. The orange path indicates the output of the path optimizer, resulting from placing the middle waypoint at the location indicated by the blue circle.

Having the optimal path $\boldsymbol{\xi}_P^*$, we divide the trajectory into M segments (as described in Sec. 3.2.1). Next, we store the positions of the waypoints at the end of the segments in $\mathbf{p}_V^* = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M]$. This vector is fixed to maintain the shape of the trajectory. The corresponding timestamps, stored in $\mathbf{t} = [t_1, t_2, \ldots, t_M]$, are the variables we optimize. Thus, trajectories sampled by the optimizer are only a function of the time vector $\boldsymbol{\xi} = f(\mathbf{t})$. By optimizing $\mathbf{t}$ we optimize the average velocity of each segment. The optimal time vector

$$\begin{aligned} \mathbf{t}^* &= \arg\max_{\mathbf{t}} \Big(R_V(\mathbf{t}; \mathscr{C}, \boldsymbol{\theta}_{HV}) + \theta_{RV}\phi_{RV}(\mathbf{t}; \mathscr{C}) \Big), \\ &\text{subject to: } \boldsymbol{g}(\mathbf{t}) \leq \mathbf{0},\ \mathbf{t} \leq \mathbf{t}_{\text{upp}}, \end{aligned} \tag{3.7}$$

where the objective function is composed of R_V and the robot's velocity objective ϕ_{RV}, which provides a reward for carrying objects at $\dot{x}_{\text{robot}}$ with a fixed weight θ_{RV}. The inequality constraint $\boldsymbol{g}(\mathbf{t})$ bounds the velocity over each segment to $\dot{x}_{\min}$ and $\dot{x}_{\max}$, not allowing the timestamps to get too close or far from each other. The upper boundary on $\mathbf{t}$ acts as a limit on the total duration of motion.

Finally, the trajectory that adheres to both the path and velocity preferences is constructed using $\boldsymbol{\xi}_R = f(\mathbf{p}_V^*, \mathbf{t}^*)$. The full method is summarized in Algorithm 1.

Algorithm 1: Learning human preferences from kinesthetic demonstration

```
Record $\boldsymbol{\xi}_H^0 = \{\boldsymbol{x}_k, t_k\}_{k=1}^N$, obtain context $\mathscr{C}$
$\dot{\boldsymbol{x}}_k \leftarrow \frac{d}{dt}\boldsymbol{x}_k$, compute $\bar{\dot{x}}_r$ and $\bar{\boldsymbol{x}}_r$
Initialize $\boldsymbol{\theta}_H^0, \boldsymbol{\theta}_R, \boldsymbol{\xi}_R^0$
Set $i = 0$
while executing task do
    if Received Human Feedback then
        $\boldsymbol{\theta}_H^{i+1} = \boldsymbol{\theta}_H^i + \alpha\left(\boldsymbol{\Phi}\left(\boldsymbol{\xi}_H^i; \mathscr{C}\right) - \boldsymbol{\Phi}\left(\boldsymbol{\xi}_R^i; \mathscr{C}\right)\right)$
    $\mathbf{p}^* \leftarrow \text{Optimize}(\boldsymbol{\theta}_{HP}^{i+1}, \boldsymbol{\theta}_{RP}, \mathscr{C})$
    $\mathbf{t}^* \leftarrow \text{Optimize}(\mathbf{p}^*, \boldsymbol{\theta}_{HV}^{i+1}, \theta_{RV}, \mathscr{C})$
    $\boldsymbol{\xi}_R = f(\mathbf{p}^*, \mathbf{t}^*)$
    $\boldsymbol{\tau} \leftarrow \text{Impedance}(\boldsymbol{\xi}_R)$
    $i = i + 1$
```

3.3. Method Validation with User Study

To validate our framework we conduct two user experiments on a Franka Emika 7-DoF robot arm. Thereby we show a proof-of-concept of our approach in a real-world scenario with non-expert users. In both experiments, we use a set of three pick-and-place tasks in an agricultural setting as shown in Fig. 3.4. The primary goal of each task was moving the tomatoes from the initial position to the goal without any collisions with the obstacle. The experiments were approved by the Human Research Ethics Committee at the Delft University of Technology on 06/09/2021. Informed consent was obtained from all subjects involved in the study.

We recruited 14 participants (4 women, 10 men) between 23 and 36 years old (mean = 26.8, SD = 3.6), six of whom had prior experience with robotic manipulators, but none of whom had any exposure to our framework.

Each user first took approximately 10 minutes to get familiar with physically manipulating the robot in the workspace. In this period, we also instructed users about the goal of the task and the preferences the robot could capture, just to help them define and generalize their own preference relative to the table and obstacle. Users then proceeded with the two experiments. To subjectively assess whether the framework can capture a range of different behaviors, in the first experiment we let the users freely choose their path and velocity preferences. Once users were more familiar with the framework, in the second experiment we assessed how effectively they could teach a set of pre-defined preferences to the robot. The overview of the user study is provided in Fig. 3.5. We discuss each experiment in the following subsections. A video of the experiments can be found here: https://youtu.be/hhL5-Lpzj4M.

3.3.1. User-Defined Preferences

In the first experiment, we investigate how our framework performs when users openly choose their set of preferences. We are specifically interested in assessing how well the robot plans motions in new task instances with a context it has not seen before (i.e.,

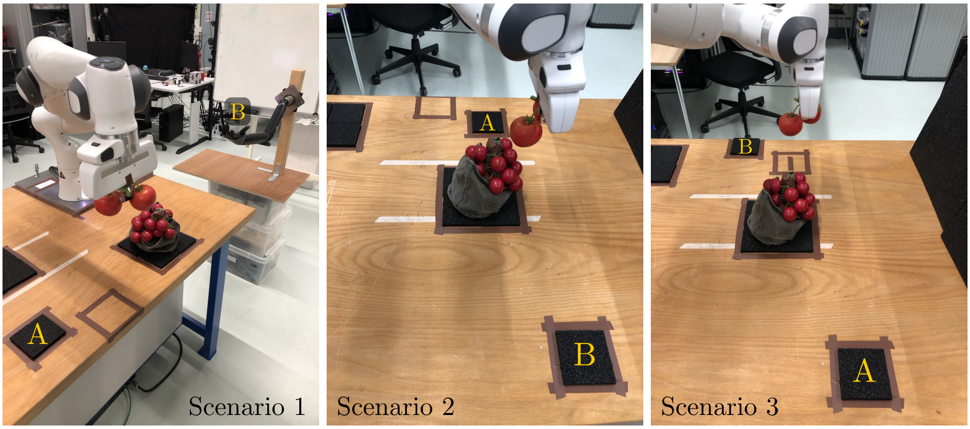


Figure 3.4: From left to right Scenarios 1–3. ‘A’ and ‘B’ indicate the start and goal positions respectively. The obstacle to be avoided is the bag of tomatoes. Scenario 1 and 2 shared the same starting positions, and Scenario 2 and 3 shared the same obstacle positions. Notice the difference in height of the goal position in Scenario 1 compared to Scenarios 2 and 3.

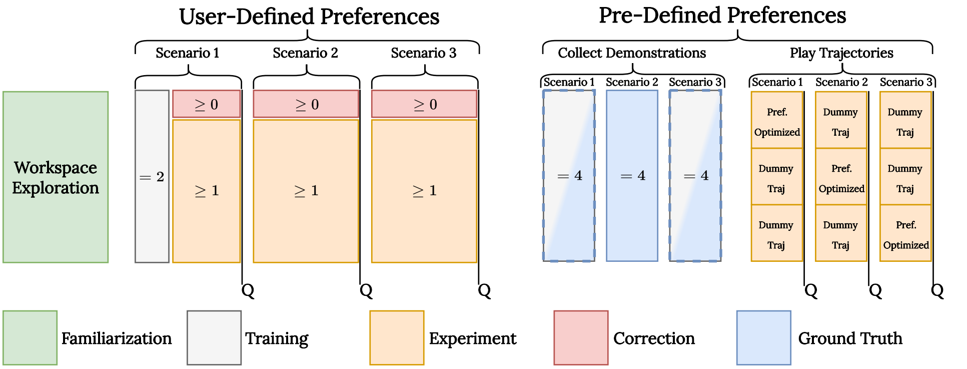


Figure 3.5: The experimental protocol. Users started with workspace familiarization, then went through the first experiment assessing the performance of the framework in understanding their preferences. Finally, in the last experiment, they provided ground truth demonstrations and evaluated the demonstrated trajectories in adhering to the set of predefined preferences. The numbers indicate the number of demonstrations given, either by the human (training/correction/ground truth) or the robot (experiment). The order in which the dummy trajectories were shown to the users was different in every scenario, to not induce a bias by making it too easy for the users to guess the “right” answer. The ‘Q’ symbols indicate when participants were provided with questionnaires.

generalization of preferences to new scenarios). We also evaluate the user experience in terms of acceptability and effort required from the user’s perspective. Accordingly, we test the following hypotheses:

H1. The proposed framework can capture and generalize user preferences to new task instances.

H2. Users feel a low level of interaction effort.

Procedure and Measures

Users first performed a demonstration in Scenario 1 (Fig. 3.4) for path preferences with the robot in gravity compensation mode. Notably, we did not limit users to a discrete set of preferences. For instance, instead of asking users to pass on either the close or far side of the obstacle, we asked them to intuitively demonstrate how far to either side of the obstacle they would prefer to pass. They could, for example, decide to pass right above the obstacle which would correspond to a "stay to the middle of the obstacle" for the "obstacle side" path preference.

We then collected a second separate demonstration for the velocity preferences. During velocity demonstrations, the robot was only compliant along a straight line path covering the full range of distances to the obstacle. This allowed the users to demonstrate their preferred speed without having to care about the path. The velocity optimization step can take up to 3 minutes[2], therefore we simplified the method for learning and planning velocity preferences to find the velocity c_j with the highest feature count in this part of the study.

Users were instructed to provide corrections via additional kinesthetic demonstrations (max 10 min per scenario) until they were satisfied with the resulting trajectory. However, the users were informed that the trajectory speed was only trained once and would not be updated further. When a user was satisfied with the robot's trajectory in a scenario, the robot would proceed to the next, generalizing what it had learned to the new context. Users were instructed to provide additional demonstrations in each new scenario if they felt the robot's trajectory did not sufficiently match their preferences.

After observing each trajectory, the users filled out a subjective questionnaire for qualitative evaluation, rating the following statements on a 7-point Likert scale:

1. The robot accomplished the task well.
2. The robot understood my **path** preferences.
3. The robot understood my **motion** preferences.

To evaluate the effort, we counted the number of times a user-provided feedback, and let the participants fill out the NASA Task Load Index (TLX) at the end of this experiment. The independent variables of this experiment are the contexts which are varied for each scenario for assessing workload.

While we do not provide a baseline here to which to compare results, NASA-TLX is still appropriate since it can capture absolute results (Hart and Staveland, 1988). The statements we asked the users to rate for the qualitative evaluation are phrased in a similar non-relative way. This way, the users evaluated our method not relative to a condition we provided, but rather to their own internal ground truth of what they perceived as sufficient understanding of their preferences and acceptable task load.

Results

Users demonstrated a multitude of path preferences, including "Keep low distance to the obstacle" and "Stay at medium height above the table". Similarly, for velocity preferences, while the majority opted for a constant "medium" speed, both the preference of

[2] The constrained nonlinear optimization is solved using the Matlab function `fmincon`.

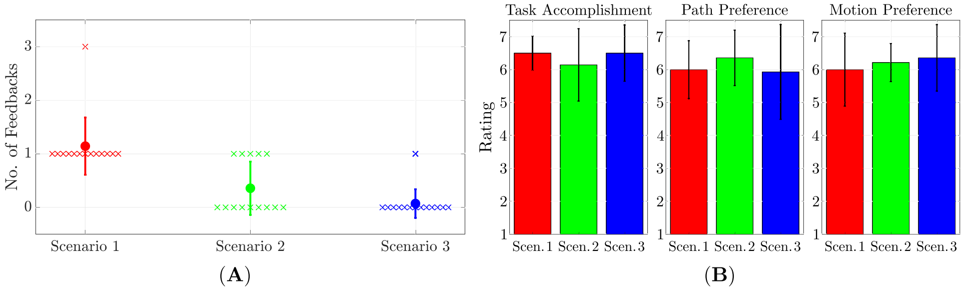


Figure 3.6: Results of the first experiment. (**A**) An average number of feedback provided to the system for each task. The dot represents the mean score, the error bars represent the standard deviation, and the crosses indicate individual data points. (**B**) Results of the Likert questionnaire for the first resulting trajectory in every task (i.e., prior to any additional demonstrations) - the error bars correspond to standard deviation.

going “slower when close to the obstacle” and “faster when close to the obstacle” were demonstrated at least once.

Fig. 3.6(A) shows that the average amount of feedback given to the system after the first task drops[3], with the majority of the users satisfied with the results of generalization after the initial demonstration (we count the training step in Task 1 as feedback). This result is also reflected in Fig. 3.6(B), showing that the users scored the first trajectory produced in every scenario consistently high for all three statements, supporting the claim that the framework can generalize both path and velocity preferences to new instances of the given task. This provides strong evidence in favor of both **H1** and **H2**.

The NASA-TLX results in Fig. 3.7 show that the users experienced low mental and physical workload. Although kinesthetic teaching is normally associated with high effort, our framework's effort scores remain mostly on the lower side of the scale. Just looking at the scores in Fig. 3.7, it may seem that the perceived “Effort” is correlated with the “Temporal Demand”. However, looking at the individual results, users who felt a large temporal demand did not perceive a large effort (to achieve desired performance) and the other way around. One participant was particularly strict on a height preference the algorithm failed to capture, resulting in 3 iterations of feedback in Scenario 1. Overall, the results in Fig. 3.7 support **H2**.

3.3.2. Pre-Defined Preferences

To objectively evaluate the accuracy, and the user's ability to discern preferences, we conduct an experiment where users are asked to adhere to the following path preferences (we did not consider velocity preferences in this experiment):

- Pass on the side of the obstacle that is closer to the robot.
- Stay far from the obstacle.
- Keep a high elevation from the table.

[3] A two-tailed Wilcoxon signed rank test for paired samples, as the data is not normally distributed, shows that fewer corrections were made in Scenario 2 compared to Scenario 1 with $p = 0.0023$ and in Scenario 3 compared to Scenario 2 with $p = 0.046$

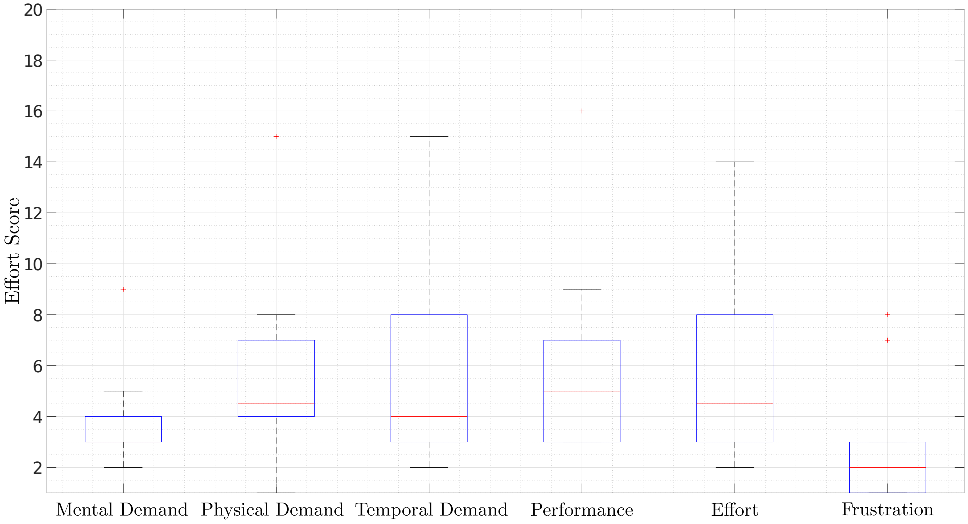


Figure 3.7: Results of the NASA-TLX questionnaire after the first experiment.

Exactly how to express these preferences, and how to trade off between them if necessary, is left to the users. We test the following hypotheses:

H3. The method remains consistently accurate in all scenarios.

H4. Users can clearly distinguish that the output of the framework is following the specified preferences.

Procedure and Measures

We collected four demonstrations per scenario. For half of the participants, we trained the model on the mean of the four demonstrations from the first scenario, and for the other half, we used the mean of data from the third scenario. This is to establish that our method generalizes over the tested contexts, even when changing the set used as the training data.

After that, the users were shown 3 trajectories per scenario: the output of our framework, and two dummy trajectories (Fig. 3.8). The dummy trajectories were designed to adhere to 2 out of 3 path preferences. This allowed us to observe if users could distinguish our method's results compared to sub-optimal trajectories.

As an objective measure of the accuracy of our method, we computed, per scenario, the total Euclidean distance of samples within each trajectory with respect to the mean of the demonstrations (using N=80). Furthermore, we compare the total feature counts along each trajectory and measure the error with respect to the ground truth in the feature space.

Subjectively, users rated a 7-point Likert scale per trajectory: “The robot adhered to the demonstrated preferences”.

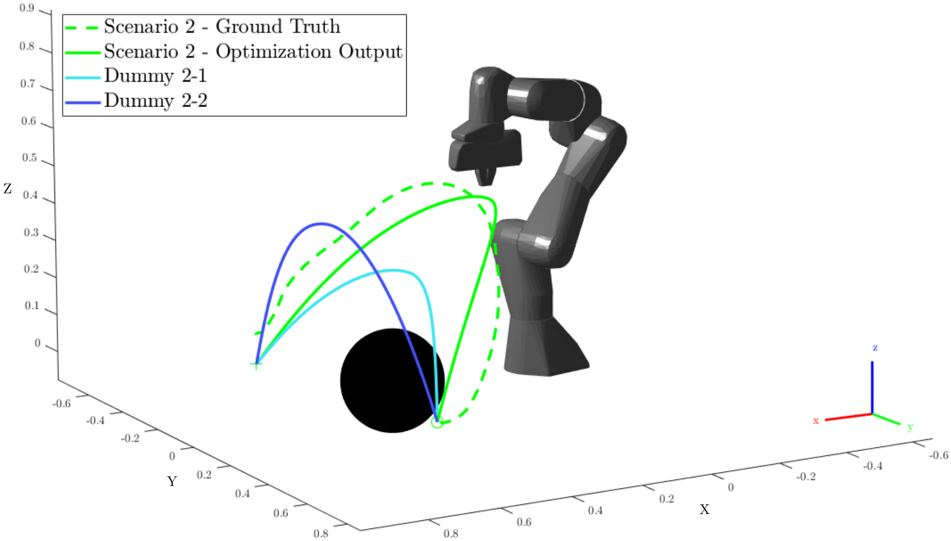


Figure 3.8: Scenario 2 results (second experiment) for a single user. The dummy trajectories, in light and dark blue, are designed not to meet the 'height from table' and 'obstacle side' preferences respectively. The green dashed and solid lines are the mean of human ground truth demonstrations and the robot trajectory respectively. The black sphere represents the obstacle. The framework was trained on data from Scenario 3 and had no access to the ground truth shown.

Table 3.1: Average distance error of trajectory samples w.r.t. the ground truth, normalized w.r.t. distance of start to goal, in meters: mean [min, max].

	Scenario 1	Scenario 2	Scenario 3
Optimized	0.14 [0.09, 0.18]	0.20 [0.12, 0.27]	0.17 [0.13, 0.24]
Dummy 1	0.24 [0.13, 0.34]	0.26 [0.16, 0.38]	0.30 [0.21, 0.41]
Dummy 2	0.27 [0.18, 0.33]	0.39 [0.28, 0.47]	0.23 [0.18, 0.28]

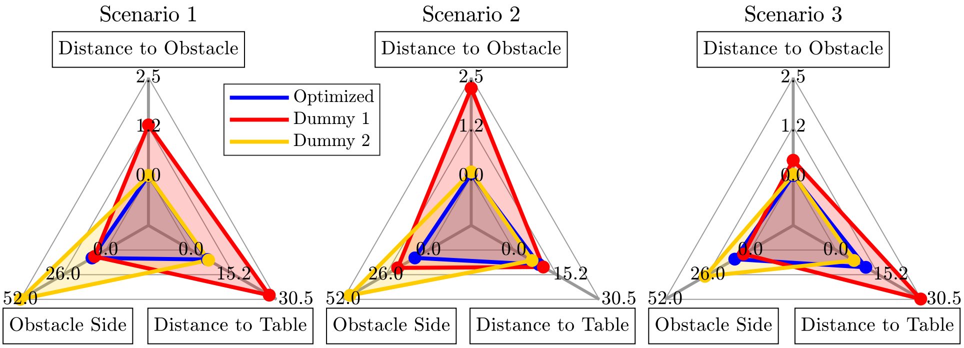


Figure 3.9: Total feature count errors of each path preference (all participants), w.r.t. the ground truth (i.e., smaller values for each axis are favored).

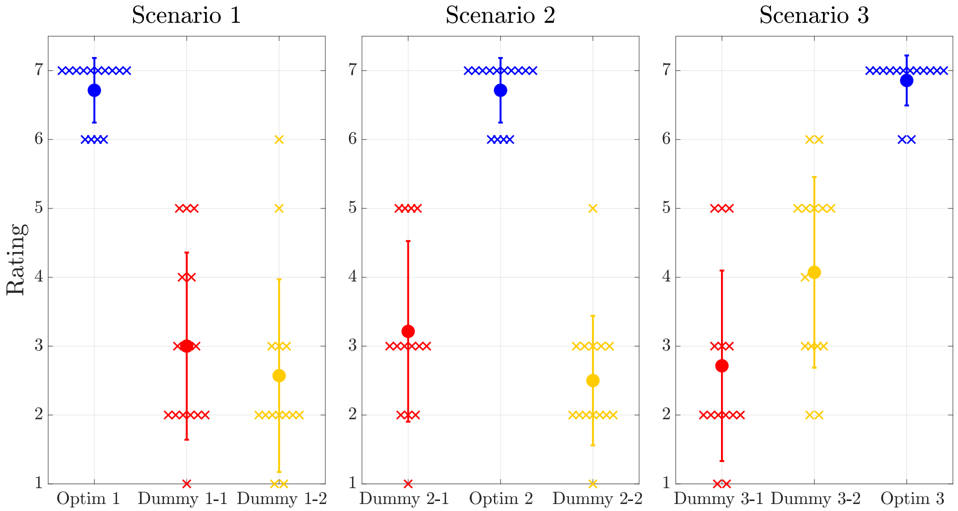


Figure 3.10: Result of Likert questionnaire for experiment 2. Crosses indicate individual ratings, while the dot and error bar, respectively, represent mean and standard deviation. Users clearly recognize and highly rate the output of the framework in terms of adhering to path preferences.

RESULTS

Fig. 3.8 shows a generalization result of our method under the aforementioned path preferences. The robot attempts to capture and optimize for each user's personal interpretation of the preferences (e.g. one user's definition of 'high' is different from another). We show the combined results of all users in Tab. 3.1, listing the trajectories' mean, min and max Euclidean distance to the ground truth, normalized relative to the start-to-goal distance in each scenario (respectively 1.08, 0.74, 0.88 m). The optimized trajectories have the smallest error, but the results only partially support **H3**, as the error in Scenario 2 and 3 is slightly larger than in Scenario 1. This scenario has the longest distance from start to goal, for which the framework seems to perform better.

Fig. 3.9 shows the errors of the trajectories in feature space. In all scenarios, our optimization result occupies the smallest area. However, in Scenario 2 and 3 the optimized trajectories occupy a slightly larger area than in Scenario 1, showing the same trend of performance loss in scenarios with the shorter length. Furthermore, in Scenario 2 and 3, dummy trajectories occasionally perform slightly better for one of the preferences. Nevertheless, we see in Fig. 3.10 that users clearly score the output of our framework higher, which strongly supports **H4**. This indicates that users prefer all preferences to be satisfied simultaneously. The best performing dummy (S3-D2), with the smallest area in Fig. 3.9 and lowest values in Tab. 3.1, correlates to a high rating in Fig. 3.10. This also supports **H4**, suggesting that non-expert users intuitively recognize such preferences in trajectories.

3.3.3. Discussion

As the state-of-the-art methods do not have the same functionalities (e.g., path-velocity separation) as the proposed method, we conducted a user study only on the proposed method itself. To account for that, we employed absolute types of metrics (i.e., Likert and NASA-TLX), which can be interpreted independently, rather than tied to a specific external baseline. For example, the Likert scale is tied to an agreement with the given statements and the natural point on the agreement scale serves as a general baseline. Therefore, the result is not tied to a specific relative baseline. If methods that enable the same functionalities are developed in the future, the same Likert scale/questionnaire can be employed to compare the subjective results independently of a specific baseline. When such methods are developed, a direct comparison is recommended, as it will provide stronger results.

An advantage of the proposed method is that it learns fast. During the first part of the user study, participants spent on average 16.5 s interacting with the robot before expressing satisfaction with the results. This is partially due to having access to kinesthetic demonstrations. This method of demonstration has been criticized as challenging in applications involving high DoF manipulators (Akgun et al., 2012; Jain et al., 2015). However, the separation of learning and control in our framework means that users do not have to provide the correct configuration of the arm in their demonstrations. This feature made it significantly easier for the users to provide demonstrations, which is reflected in the reported low mental and physical loads (Fig. 3.7).

The separation of path and velocity planning has additional benefits. Formulating the optimization as a multi-objective problem with both position and velocity features results in undesirable interaction of objectives. For instance, when velocity features reward high speeds, the trajectory would converge to a longer path. Conversely, path features with high rewards in specific regions of space would result in slow motion in those regions to increase the density of samples and consequently the overall reward. On the other hand, the separated trajectory optimization has the limitation that it cannot account for dynamical quantities such as joint velocity and acceleration, and the efficiency of movements in robot's joint space can not be considered.

A challenge with our definition of robot and user objectives is that the trajectory optimization outcome does not always align with task requirements. For instance, a strong "stay close to the object" preference can result in a minimum cost for a trajectory that is briefly in a collision. Tuning the collision weight can only partially solve this issue, as at a certain point this cost can interfere with the path preferences.

Our user study results showed that non-expert users can intuitively use our method to quickly teach a wide range of preferences to the robot. While the generalization results to different task instances show that we do not always reproduce trajectories with the exact desired shapes in the workspace (see Fig. 3.8), the subjective performance evaluation shows that users still deem these trajectories highly suitable in terms of task accomplishment and the preferences achieved. State-of-art LfD methods are very capable of producing accurate and complex dynamic movements (Mülling et al., 2013). However, in tasks where there are multiple ways of achieving the same goal, we prefer to trade off motion accuracy for achieving planning propensities on a higher level.

Unfortunately, our approach inherits the limitations of IRL approaches that require specifying reward features by hand. Both features and robot rewards depend on several parameters which require tuning. The problem becomes especially difficult as our features simultaneously govern the behavior of reward learning and trajectory optimization. For instance, high gradients in the feature function lead to erratic behavior of the optimizer, leading to poor solutions and convergence to local optima. Yet, for certain features, a sufficiently high gradient is required to facilitate the learning of preference weights that are large enough to counterbalance each other. As a result, we had to resort to further tuning of parameters, such as the learning rate in (3.4). An interesting direction for future work would be to test if and how well these issues can be alleviated by feature learning from additional demonstrations, as was done in Bobu et al. (2022). Furthermore, an approach in (Katz et al., 2021) could be employed to learn the relative weighting among features and add additional features through nonlinear functions using neural networks.

In feature engineering or learning, the definition of the context determines how expressive the features are. We considered a limited set of vectors as the context in this work (i.e., obstacle position, start and goal positions). It is possible to include additional information, such as object properties (e.g. sharp, fragile or liquid) (Jain et al., 2015), human position (Bajcsy et al., 2017; Losey et al., 2022), and number of objects. The more rich the context, the more preferences the model can capture in complex environments. However, training diversity can become an issue with contextually rich features, as the model would require more demonstrations to cover a wider range of situations. This will increase training time. An evaluation of the trade-off between improved generalization and higher training time is left for future work.

3.4. Supplementary Comparison Study

The purpose of this supplementary study is to highlight different aspects and properties of our method in comparison to two common methods from the literature: Dynamic Movement Primitives (DMPs) (Calinon and Lee, 2019) modified with potential fields for obstacle avoidance (Gams et al., 2016), and the method used in Bajcsy et al. (2017) (referred to as PHI). Since these methods are different conceptually and by design (i.e., optimize for different properties), we examine their aspects in a practical transportation task qualitatively. These aspects are: adherence to preferences, robot objectives, trajectory feasibility, and online learning. In the next subsections, we first discuss the different aspects in more detail, before showing the effects in the transportation task and discussing the pros and cons of the different methods.

3.4.1. Conceptual Differences per Aspect

Adherence to Preferences

The methods capture preferences in a different way. Even though we added obstacle awareness to the DMPs we compare to, they lack an explicit notion of preferences. A forcing function is learned to match the shape and velocity distribution of the demonstration, but without any parameterization over features that may capture behavior relative to the context. The potential fields for obstacle avoidance add a basic level of context-awareness, but a predefined one.

Both our method and PHI learn an explicit preference model that is structured as a linear combination of context-parameterized features. Like our model, PHI considers the "*Height from the Table*" and "*Distance to the Obstacle*". We additionally consider the "*Obstacle Side*", such that our features cover the different dimensions in space and allow us to capture the preferences in every direction. PHI instead considers other features, such as "*Distance to Human*" and "*Efficiency*".

Our features are counterbalanced by explicit robot objectives (Sec. 3.4.1). In PHI, it is possible to replace the features with the features we use, including the ones for the robot which will not be updated during learning. This way, we can test the effects of the change of features and the change of method.

Robot Objectives

In contrast to PHI, we chose to explicitly separate objectives such as "*Path Efficiency*" and "*Collision Avoidance*", from the preferences we try to capture. Instead, we let the robot have a reward function of its own. The same effect can be achieved in PHI by fixing the weights of selected features.

The effects of the trade-off between the learned human rewards and the given robot objectives visible in the iterative updates can be viewed as a negotiation between the preferences of two independent agents. We believe that this separation and negotiation will be beneficial especially as tasks become more complex and the artificial agent has knowledge complementary to the human. The benefits may be less visible in the simple task considered in this chapter.

As DMPs do not explicitly model an objective function to be optimized, this attribute does not apply.

Trajectory Feasibility

Our method does not automatically check if the planned trajectory is feasible to execute by the robot. A motion feasibility objective can be added to the robot objective function to take this into account in the path optimization. Alternatively, it is possible to rely on a lower-level controller to take care of the feasibility. However, the lower-level controller will be agnostic to the optimization that provided the target trajectory. Therefore, it is better to take the feasibility into account already during the optimization.

Rather than weighing the learned preferences against robot objectives, PHI ensures motion feasibility by optimizing the trajectory in the robot configuration space. This requires an additional simulation step, incorporating the kinematic model of the robot, during trajectory optimization. But then, no corrections need to be applied in hindsight to ensure trajectory feasibility.

Online Learning

Our method requires a full trajectory to learn from, whereas PHI updates the internal model at each time step. This potentially makes our method less efficient. On the other hand, it allows us to capture velocity preferences in addition to path preferences. Also, because we separate the demonstration from the execution, we obtain a more 'clean' observation of the preferences, as we do not have to deduce from the interaction forces what the human demonstration would have looked like without the robot interference.

This is likely to benefit generalization. In case of large user corrections, it may even reduce the user's effort to demonstrate their preferences without the robot interfering. Especially velocity preferences have been found cumbersome to correct in an online manner (Losey and O'Malley, 2019).

When it comes to online updates, DMPs are the fastest, because there is no complex underlying model that needs to be updated. But the trade-off of having a much simpler model is that it lacks the ability to capture preferences in a way that might generalize to changes in the scenario.

All three methods update their model to reduce the error with respect to the latest observation from demonstration. A learning rate trades off learning and overfitting on the corrective demonstration. DMPs updates correct the behavior on the trajectory level, whereas PHI and our method update at a higher level where the observed trajectories are considered a consequence of a human reward model. Nevertheless, future observations that appear contradictory to earlier ones will cause (partial) unlearning of the earlier updates. This results in erroneous behavior learned from imperfect corrections to be corrected, but may in some cases also lead to undesired unlearning.

3.4.2. COMPARISON

We will now present a qualitative comparison between the three methods, PHI with two different feature sets: _ϕ_{orig}, _ϕ_{our}. Our aim is to show the effects of the conceptual differences discussed in Sec. 3.4.1. To make the comparison as fair as possible, we let all the models learn from the same demonstration data. All methods have access to and consider the obstacle position for planning.

We modify PHI to bypass the estimation of the human desired trajectory from forces, as we have direct access to the desired trajectory from demonstration. We compute the "human correction" every time step from the mismatch between the planned trajectory and the demonstration. The trajectory optimization in PHI requires a robot model for the optimization. As our trajectory optimization does not take the robot dynamics into account, we use a fully actuated point mass for the trajectory optimization. In order to achieve comparable smooth optimal paths, we interpolated the trajectory with a spline instead of linearly as was done originally. For both sets of features, the feature weight ranges and update rates were hand-tuned to achieve as close a trajectory match in the initial scenario as we could manage. This initial scenario is illustrated in Fig. 3.11.

We consider a situation where the user has a preference for "passing on the close side of the obstacle" due to the existence of a wall on the other side that the robot is not aware of. Furthermore, we want to "remain close to the obstacle", and to "slow down when passing close to the obstacle". We use a single kinesthetic demonstration containing these three preferences as the input to all methods. For PHI_ϕ_{orig}, we obtained the correct choice of obstacle side in Fig. 3.11 by assuming a person standing on the other side of the obstacle and making use of their "human feature", learning not to come too close to the human.

Fig. 3.11 shows the demonstration we use for training, as well as the trajectories obtained from the three methods. As the results are generated for the same context as in the demonstration, these results reflect the performance prior to any generalization of preferences. As PHI updates its internal model at every time step, we observe partial un-

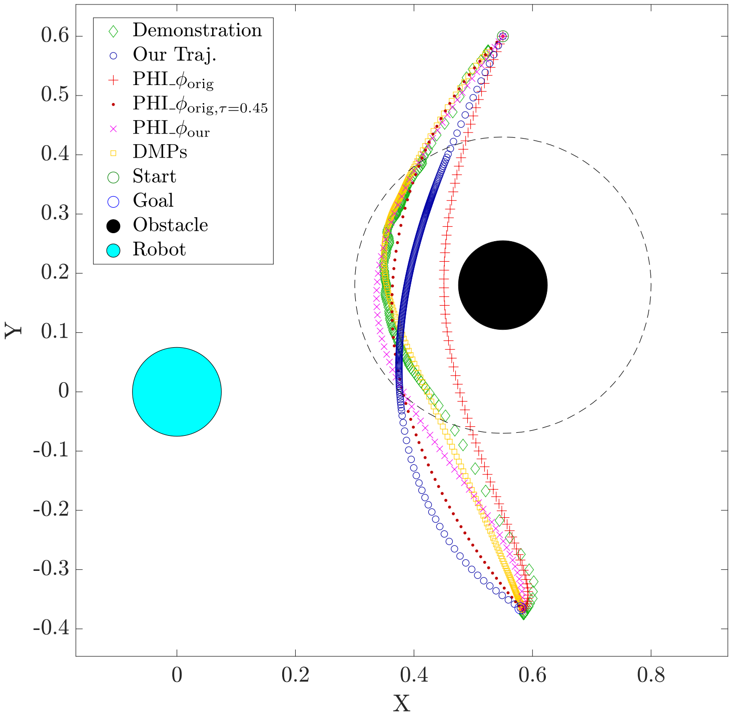


Figure 3.11: Training scenario with the human demonstrated trajectory (green diamonds) and the learned reproductions: ours in dark blue circles, PHI_ϕ_{orig} in red plus signs, with an intermediate learning result in dots, PHI_ϕ_{ours} in purple crosses, and DMP in yellow squares. By placing the markers at equal time intervals, we display the velocity on the trajectories (i.e., the closer the markers, the slower the motion). As PHI does not support differences in velocity, all red and purple markers are spaced equally along the trajectory. The black, cyan, blue, and green circles respectively represent the obstacle, robot, goal (bottom), and start (top) positions. For this study, we set d_c= 22.5 cm (indicated by the dashed circle). We consider points within this region as “close” to the obstacle.

learning of some features towards the end of the trajectory. This is particularly visible for the “Obstacle Distance” in PHI_ϕ_{orig}. In Fig. 3.11, we show an additional trajectory PHI_$\phi_{\text{orig},\tau=0.45}$, which is generated by PHI with the original features and the weights learned at 45% of the trajectory. We see that PHI_$\phi_{\text{orig},\tau=0.45}$ is considerably closer to the demonstrated trajectory. The demonstrated trajectory has many waypoints close together, quite close to the obstacle, as it slows down when passing it. PHI, on the other hand, has its waypoints equally spaced. As a result, towards the end of the trajectory, a considerable batch of PHI waypoints is further away from the obstacle by default. When the weights continue to update on the difference, we obtain the trajectory PHI_ϕ_{orig}, which lies closer to the obstacle. With our features, in PHI_ϕ_{our}, the effect is less pronounced as the features trade off differently, yet the learned path is still different from Our Trajectory, as PHI uses a different trajectory optimization method.

Especially considering PHI_ϕ_{our}, all three methods perform reasonably well in terms of adhering to the aforementioned path preferences, with a slight variation in how close

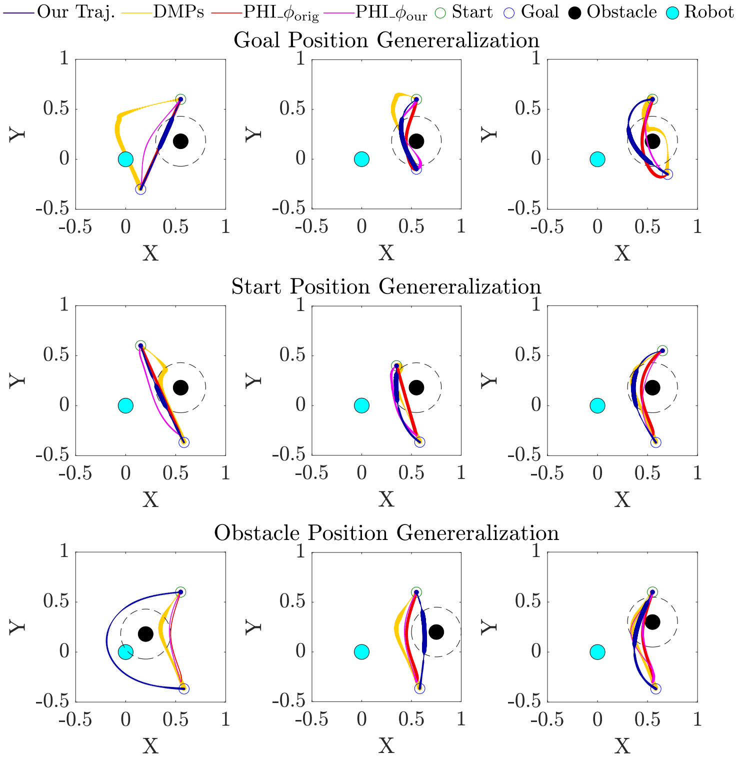


Figure 3.12: We demonstrate generalization by modifying the goal (top), start (middle), and obstacle (bottom) positions. The yellow, blue, and red and purple trajectories correspond respectively to the output of the DMPs, our framework, and the two versions of PHI. The thickness of the line indicates the inverse of normalized velocity (i.e., the thicker the line, the slower the trajectory).

the robot passes by the obstacle. As discussed in Sec. 3.4.1, PHI is not able to capture any velocity preferences. Notably, DMP performs well in this aspect as it is able to replicate the demonstrated behavior in terms of both path and velocity.

Next, we modify the scenario nine times, in three different ways: changing respectively the goal, start, and obstacle positions. We compare how each method is able to generalize the initial observation to the different contexts. Fig. 3.12 displays trajectories produced by the three methods in the nine new scenarios, PHI with the two different feature sets.

We observe that the trajectory by our method (shown in dark blue) passes on the left side of the obstacle, close to the robot, in every case. This does not necessarily mean that person providing the original demonstration would generalize the exact same way. In the bottom left case, this is even quite unlikely. If we look just at the path, PHI performs reasonably well, with both feature sets, proposing a more likely path in the bottom left case, but passing through the obstacle most clearly in the top right case. However, PHI is not able to capture any velocity preferences. The velocity preference, of slowing down when passing close to the obstacle, is only achieved by our framework.

Table 3.2: Qualitative evaluation of the different aspects of the three methods: DMPs, PHI, and ours. The marker 'o' indicates a value between '-' and '+'.

	Adherence to preferences[a]	Robot objectives[b]	Trajectory feasibility[c]	Online learning[d]
DMPs	-	-	-	+
PHI	o	o	+	o
Ours	+	+	o	-

[a] The criteria, based on Fig. 3.12, where '-' is given for adherence to only a few, or inconsistently many, preferences, and '+' for adherence to most preferences in most of the cases.
[b] The criteria, based on the model structure, where '-' is given when no robot objectives can be added, and '+' when arbitrary robot objectives can be added.
[c] The criteria where '-' indicates no guarantees for trajectory feasibility, and '+' indicates trajectory feasibility can be guaranteed at all times.
[d] The criteria where '-' indicates the inability to learn in real-time, and '+' indicates the ability to learn and re-plan while the task is being executed.

3.4.3. Discussion

The comparison with DMPs illustrates how a lack of higher-level knowledge about why a trajectory was demonstrated in a specific manner leads to failure in generalization to new contexts. These results emphasize the need for consideration of human models, such as our reward in (3.1), in LfD methods. PHI, with its model, does considerably better. However, we observe that the internal trajectory optimization reacts differently to the different sets of features, resulting in slight differences in generalized trajectories. The main point, regardless of the applied features, remains that PHI is not able to capture velocity preferences. Tab. 3.2 summarizes the strengths and weaknesses of the three methods with respect to the aforementioned aspects.

PHI optimizes the trajectory in the joint space, which can be done fast since inverse kinematics is only required at waypoints. It ensures the planned trajectories are feasible for the robot, which can be interpreted as implicit robot objectives being satisfied. On the other hand, our method optimizes the trajectory in task space, thus additional inverse kinematics computations are necessary together with an explicit description of corresponding robot objectives. The use of inverse kinematics can also be problematic when there are redundant DoF or when there are potential self-collisions. Nevertheless, planning in the task space is closer to where the human preferences typically are (i.e., more intuitive) and can handle obstacle avoidance in a manner that is more predictable for a non-expert human.

It should be noted that our framework does take up to two minutes of optimization due to the constrained nonlinear optimization steps for both path and velocity. We did not optimize our path and velocity optimizations as they were fast enough for our current purpose. Considerable computation speed can potentially be gained by more careful choices of optimization algorithm. Nevertheless, DMPs will still be faster to compute, as they do not require any further optimization after training. However, there is no guarantee that the DMPs will encode and generalize the desired preferences.

3.5. Conclusion and Future Work

We presented a novel approach for learning and executing human preferences in robot object-carrying tasks. Our user study showed fast convergence of the algorithm, in terms of number of human corrections, and a proof-of-concept for generalizing path and velocity preferences between contexts within a given task. The efficiency and accuracy of our approach were validated in a real-world scenario. Our supplementary study compares the performance of our framework to two common methods from the literature, providing additional insights into the benefits and drawbacks caused by the structural differences between the methods. Both in the user study and in the supplementary study, a single informative feedback sufficed (in all cases except one) to capture the human preferences. In the user study, this was tested without prescribing a preference to the users. Our framework was in most cases successful in generalizing these preferences to previously unseen scenarios. Our results support that our model contributes to personalized planning of object-carrying tasks with low interaction effort.

Future studies comparing our method (with just path preferences) to PHI (Bajcsy et al., 2017; Losey et al., 2022) in a user study could lead to useful insights into people's preferences on iterative versus online learning. Further research could consider a combination of our method and PHI that would benefit from the advantages of both, namely achieving generalization both in-task and over new task instances through learning from online interaction. Next to that, the trajectory model we used to make the problem tractable is quite simplistic and does not describe human motion behavior very well. Future research can aim to replace this model with a library of motion primitives generated from demonstrations to better capture the shape of the trajectories. More accurate trajectory models can enable the extension of the framework to settings where the human and robot come into contact with each other through a shared object (physical human-robot collaboration). Furthermore, it should be studied whether more complex nonlinear formulations of the reward function using Gaussian Processes (Bıyık et al., 2020) or Neural Networks (Ibarz et al., 2018), and/or learning them from user input (Bobu et al., 2022; Katz et al., 2021), can effectively capture context-aware preferences without the need for rigorous feature engineering. We believe the presented framework is especially effective in collaborative settings where knowledge of the preferences of a partner is essential to the execution of the task.

4

Disagreement-Aware Variable Impedance Control for Online Learning of Physical Human-Robot Cooperation Tasks

In order to make the coexistence between humans and robots a reality, we must understand how they may cooperate more effectively. Modern robots, empowered with reliable controls and advanced machine learning reasoning can face this challenge. In this article, we presented a Disagreement-Aware Variable Impedance (DAVI) Controller, where the robot stiffness is regulated as a function of the perceived disagreement with the human cooperator. We tested the algorithm on a 7 DoF Franka Emika Panda robot performing the learning of a pick&place task with continuous adaptation of the goal location and the via-points with human interactive corrections, triggered by our proposed approach. A pilot study was conducted with 5 users in order to understand the reliability of the method.

The contents of this chapter are based on the paper of the same name, by Linda van der Spaa, Giovanni Franzese, Jens Kober, and Michael Gienger, published in *IEEE International Conference on Robotics and Automation full day workshop – Shared Autonomy in Physical Human-Robot Interaction: Adaptability and Trust, 2022.*

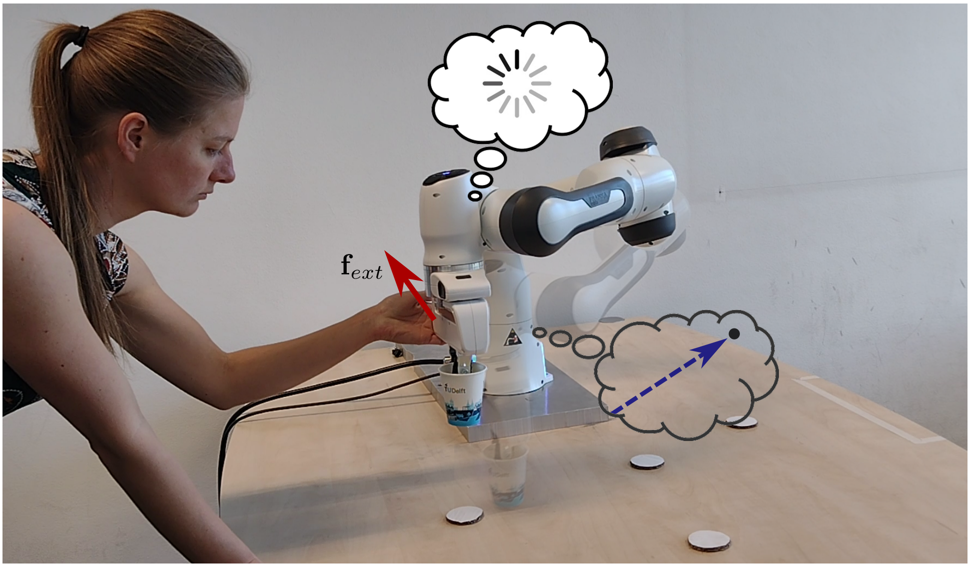


Figure 4.1: Scenario for (cooperatively) moving a cup to one of the available coasters. Initially, the robot does not know where or how to move the cup. The robot can be guided even when the human moves the cup without touching the robot.

4.1. Introduction

The strength of a team depends very much on the ability of its members to cooperate. Humans and robots have different strengths and weaknesses. Potentially, teaming up a robot with a human would allow the partners in the team to complement each other to the benefit of both. However, the actual benefit depends on the cooperation skills of both the human and the robot. In case these are lacking, the attempted cooperation may instead inadvertently lead to reduced performance.

Successful cooperation requires partner-awareness, as well as the ability to communicate and negotiate on personal preferences (how to do something), intentions (what to achieve) and constraints. Factors like preferences depend, at least partially, on the partners in the team and the cooperation between them. Therefore, we argue that most effective cooperation can be achieved when learned online; i.e. while trying to cooperate, the agents update their behavior to improve on the overall result. As cooperation skills grow over time, preferences may change as well as what cooperative behavior may be optimal. Life-long learning is required in order to keep adapting accordingly.

Our specific interest is in physical human-robot cooperation (pHRC) tasks with prolonged physical interaction continuing over a sequence of dependent actions. In such tasks, haptic communication has been shown effective in integrating intentions in shared decision making (Groten et al., 2012), and is actually able to lead to faster optimal decisions than explicit communication (Pezzulo et al., 2021). We focus on tasks where human and robot move an object to a new location in space, only communicating intuitively through the interaction forces (see Fig. 4.1).

Ultimately, the robot has to learn how the human prefers to do the task, in order to provide appropriate support. Starting from an initial solution which allows the human to finish the task together with the robot, we want the robot to learn to adhere to the human partner's personal preferences, learning from feedback the human implicitly provides in the interaction. Before we can focus on this rather complex problem in the next chapter, we need a control framework to handle the physical interaction. Specifically, we need a controller that can detect the human partner's disagreement from the physical interaction and in response smoothly transition to a learning mode that allows the robot to update or extend any existing internal model.

In this chapter, we set up a toy learning scenario for illustrative purposes (Algorithm 2), for the sake of testing our proposed control framework (Algorithm 3). We learn some high level target policy $\pi(\boldsymbol{x})$, but more importantly allow a modulation of the robot stiffness as a function of the disagreement with the human. The control of the task is negotiated between the human and the robot and when the robot is passive, the task model and desired policy are updated interactively.

The variable admittance/impedance of the robot for a safer human robot interaction was already proposed in the literature. For example, in Khoramshahi and Billard (2020), the robot admittance is increased when the human applies sufficient positive work to the system, effectively and smoothly changing the robot behavior to that of a passive follower but without interactively improving or modifying the desired execution of the task. Alternatively, Franzese et al. (2021) proposes to decay the stiffness as a function of the epistemic uncertainty of the policy encoded with a Gaussian Process.

We show equally smooth transitioning for impedance controlled trajectory tracking, ramping down the impedance upon detection of significant interaction force to the point that the robot can become fully passive, handing over all control to the human to learn from the new demonstration. Our method is not limited to using the interaction force as the metric of disagreement.

In this chapter, we present a control framework for pHRC which 1) has disagreement-awareness in physical interaction, 2) responds smoothly to detected disagreement including a negotiation phase in which the control is transferred to the human, and 3) allows a higher-level learning algorithm to learn from both the disagreement and subsequent corrections. We evaluate our framework in a cooperative pick-and-place scenario with a 7 DoF Franka Emika Panda robot and a small number of users.

4.2. METHOD

OUR DAVI controller is designed to allow and recognize user disagreement on the trajectory level and handle it in a user friendly way, with the objective to allow a cooperative robot to try out supportive policies and learn from its mistakes. Before we give a detailed description of the controller, we will first present a toy learning example to provide a context for the controller (Sec. 4.2.1), also for later testing. After that, we will explain the impedance control basis (Sec. 4.2.2) on top of which we built the DAVI controller (Sec. 4.2.3).

Algorithm 2: Illustrative toy policy learner

```
1  Initialize: states set $\mathcal{S} = \{\mathbf{x}_0\}$, goal states set $\mathcal{S}_g = \emptyset$, policy transitions $\mathcal{T} = \emptyset$
2  while Learning do
3      Start episode in $\mathbf{x} = \mathbf{x}_0$
4      while Episode do
5          if $\mathbf{x} \in \mathcal{S}_g$ then
6              deactivate DAVI (as if disagreement detected)
7              quit Episode
8          if not isActive(DAVI) $\wedge\ \dot{\mathbf{x}} = 0$ then
9              if $\mathbf{x} \neq \mathbf{x}_k$ then
10                 $\mathcal{S} \leftarrow \mathcal{S} \cup \mathbf{x}$
11                 $\pi(\mathbf{x}_k) = f(\mathbf{x}_k, \mathbf{x}, \mathcal{C})$, $\mathcal{T} \leftarrow \mathcal{T} \cup \pi(\mathbf{x}_k)$
12                 $k \leftarrow k+1$, $\mathbf{x}_k \leftarrow \mathbf{x}$
13             else if timeout then
14                 $\mathcal{S}_g \leftarrow \mathcal{S}_g \cup \mathbf{x}$
15         if ((isActive(DAVI) $\wedge\ \mathbf{x} = \mathbf{x}_{k+1}$) $\vee$ (not isActive(DAVI) $\wedge\ \mathbf{x} \in \mathcal{S}$)) $\wedge\ \pi(\mathbf{x}) \in \mathcal{T}$
            then
16             $k \leftarrow k+1$, $\mathbf{x}_k \leftarrow \mathbf{x}$
17             $\mathbf{x}_{k+1} = \text{finalState}(\pi(\mathbf{x}_k))$
18             activate DAVI$(\mathbf{x}, \pi(\mathbf{x}_k))$
```

4.2.1. State and Action Learning from Interactions

We define a discrete set of states $\mathcal{S}$ as the points (in continuous space) through which the human may want the robot to pass when doing the task. Some of these states may be terminal states in which an episode is considered finished. These are stored in $\mathcal{S}_g$. We consider a 3D end-effector workspace with a fixed end-effector orientation, not caring about the robot's joint configuration. Initially, the state space only contains the starting position $\mathbf{x}_0$, and no actions $\pi(\mathbf{x})$ are known, resulting in an empty transition set $\mathcal{T}$ (L. 1 in Algorithm 2).

We let all episodes start with the robot end-effector position $\mathbf{x}$ in start state $\mathbf{x}_0$ (L. 3). Episodes then run for as long as the robot is not in a goal state $\mathbf{x} \in \mathcal{S}_g$ (L. 5-7). Since state $\mathbf{x}$ measured at the robot end-effector is a continuous state, all conditions checking whether $\mathbf{x}$ is in a set with sampled values (e.g., $\mathcal{S}$ and $\mathcal{S}_g$), as well as equality and inequality conditions, are implemented by checking whether $\mathbf{x}$ is closer to one of the stored values than a predefined small distance ε, which we set at 2 cm. To help the human feel where these states are, we let the robot transition to active attraction to states in $\mathcal{S}$ within 10 cm distance.

When the robot is stopped in an new position (L. 8-9) for 0.25 seconds, it is considered a new state and added to $\mathcal{S}$ if it is not close to any value already in the set (L. 10). Whether or not $\mathcal{S}$ already contained $\mathbf{x}$, the state is stored as the desired next state, possibly with an action trajectory description which may also be a function of the previous state $\mathbf{x}_k$ and context parameters $\mathcal{C}$, the desired policy $\pi(\mathbf{x}_k)$ to be followed, the next time $\mathbf{x}_k$ is visited. This is stored or updated in the transition set $\mathcal{T}$ (L. 11). The data aggregation is communicated to the user by a haptic vibration of the end-effector and the index keeping track of the visited (waypoint) states in $\mathscr{S}$ is incremented (L. 12).

If the robot is not moved from a state for at least 5 seconds (L. 13), the state is flagged as a final goal state (L. 14). Before the state is added, the robot hand signals with a countdown of three vibrations to alert the human. This would allow them to continue the demonstration if the state was not their intended final goal.

Whenever the robot reaches a state which is either the next state to which the robot was actively moving (L. 15, first condition) or a state recognized while passively being moved past it (L. 15, second condition), it is checked if $\mathscr{T}$ contains a policy for that state (L. 15, third condition). If so, the next desired state is extracted from the stored policy (L. 17) and the DAVI controller is activated to follow the stored policy (L. 18).

The DAVI controller (Algorithm 3) runs on a separate thread and has direct access to the position, orientation, forces, and torques measured at the end-effector. Upon activation (L. 1), the reference trajectory for the impedance controller (Sec. 4.2.2) is initialized to start at the actual current position of the robot (L. 2).

For the demonstration of the DAVI controller, it is of little importance what specific target trajectories we provide for tracking. Therefore, we keep it simple and connect the current state and the next desired one with a straight-line trajectory, assuming the absence of obstacles. The methods we use are not limited to linear trajectories.

4.2.2. Cartesian Impedance Control

As a base control layer, we use a Cartesian impedance controller (Algorithm 3). Briefly, in Cartesian impedance control (Hogan, 1984), the end-effector dynamics are modeled in the form of a mass-spring-damper system

$$\Lambda(\mathbf{q})\ddot{\mathbf{x}} = \mathbf{K}\Delta\mathbf{x} - \mathbf{D}\dot{\mathbf{x}} + \mathbf{f}_{\text{ext}}, \tag{4.1}$$

where $\Lambda(\mathbf{q})$ is the physical system's Cartesian inertia matrix, $\mathbf{K}$ is a diagonal matrix with the desired stiffness in the principal directions, $\mathbf{D}$ is the corresponding critical damping matrix, and $\mathbf{f}_{\text{ext}}$ are the external forces. The external forces are estimated using the provided model of the robot (mass matrix, Coriolis, gravity) and the estimated joint friction provided in Gaz et al. (2019).

We distinguish between active and passive mode. In active mode, the end-effector is controlled to follow a trajectory. We employ a relatively low impedance (of ≤ 600 N/m) for safe physical interaction. To keep $\Delta\mathbf{x}$ from growing too large, we apply online attractor distance modulation (Gams et al., 2009) to allow a reactive following with a limit on the force exerted by the robot according to

$$\hat{\mathbf{x}} = \mathbf{x}_0 + \alpha(\mathbf{x}_1 - \mathbf{x}_0) \qquad 0 \leq \alpha \leq 1 \tag{4.2}$$

$$\dot{\alpha} = \frac{v_{\text{ref}}}{\|\mathbf{x}_1 - \mathbf{x}_0\|} \frac{1}{1 + \|\mathbf{x} - \hat{\mathbf{x}}\|/l} \tag{4.3}$$

where α determines the progress of the (in this case linear) trajectory, l is the equivalent tracking error that makes the progress rate to drop to half, and v_{ref} (of 0.3 m/s) is the desired velocity along the trajectory.

In passive mode, the end-effector stiffness and damping are set to zero. In this circumstance the robot is still gravity compensated. At any time during active mode, a

Algorithm 3: DAVI controller

```
Function activate(x, ζ):
    Initialize: reference trajectory $\zeta$, target position $\hat{\mathbf{x}} = \mathbf{x}$
    isActive = True
    run()
Function run():
    while isActive do
        $\hat{\mathbf{x}} = g(\mathbf{x}, \hat{\mathbf{x}}, \zeta)$                                   Eq. (4.2)
        $\gamma = \text{Disagreement}(\mathbf{f}_{\text{ext}})$
        $\dot{K} = \text{ImpedanceModulation}(\gamma),\ K \in [0, K_{\max}]$      Eq. (4.6)
        $\text{ImpedanceControl}(\hat{\mathbf{x}}, K)$
        isActive = $(K > 0)$
```

detected disagreement triggers a transition to passive mode; this allows the human to kinesthetically demonstrate the new desired behavior. At all times, the robot records the states and actions it observes during interactive task execution so it can learn from them.

4.2.3. Disagreement Detection and DAVI Control

Intuitively, disagreement can be detected based on interaction force/torque, or deviations from the expected trajectory. The two are coupled by the set robot impedance(s). Alternatively, we can detect disagreement based on the human *virtual* work, the work they would do if the robot would not exert a force. In contrast to reacting to the *actual* work the human does (Khoramshahi and Billard, 2020), this also detects disagreement when the human keeps the robot from moving.

The energy the human is injecting into the system is the virtual work done by the external forces. Substituting Eq. (4.1), we can write the linear approximation:

$$E_{\text{ext}} = -\mathbf{f}_{\text{ext}}\Delta\mathbf{x} = \mathbf{f}_{\text{ext}}^T\mathbf{K}^{-1}\left(\mathbf{f}_{\text{ext}} - \Lambda(\mathbf{q})\ddot{\mathbf{x}} - \mathbf{D}\dot{\mathbf{x}}\right). \tag{4.4}$$

We will now take a closer look at the terms in this equation to see how we may simplify. The equation is meaningful as long as the stiffness $\mathbf{K}$ is positive. The inertia $\Lambda(\mathbf{q})$ and damping $\mathbf{D}$ are positive by default. Only if the human partially gives in to the robot, the velocity and external force may have directions such that the product $\mathbf{f}_{\text{ext}}^T\mathbf{D}\dot{\mathbf{x}}$ results in a negative number. If the human is decelerating the robot, $\mathbf{f}_{\text{ext}}^T\Lambda(\mathbf{q})\ddot{\mathbf{x}}$ is still positive. When the robot is starting an action, it may be that the human only partially counteracts its acceleration towards the (low) target velocity and $\mathbf{f}_{\text{ext}}^T\Lambda(\mathbf{q})\ddot{\mathbf{x}}$ can briefly be negative as well. At the ending of a robot action, the human could similarly counteract the robot decelerating, e.g., when the human does not want the robot to stop at the position it is aiming for. In all other cases, approximating Eq. (4.4) by the simple square

$$E_{\text{ext}} = \mathbf{f}_{\text{ext}}^T\mathbf{K}^{-1}\mathbf{f}_{\text{ext}} \tag{4.5}$$

will result in an overestimate of the actual injected energy. As the moments in which the human partially counteracts the robot as described above are brief transient phases, occurring at low velocity and acceleration (with a lightweight robot), we feel justified to further just consider the simplification of Eq. (4.5).

Thus considering that the injected external energy can be estimated as a function of the norm of the external force (and controlled stiffness), we assign a negative value to our disagreement constant γ every time the external force is beyond a safety threshold $\mathbf{f}_{\text{ext}}^{\text{th}}$ and positive otherwise (Algorithm 3 L. 8). Because of the simplification from Eq. (4.4) to Eq. (4.5), it may be that described transient phases take slightly longer than they would if we would respond instead to a full estimate of E_{ext}.

The stiffness changes according to

$$\dot{K} = \text{sign}(\gamma) K_{\text{max}} / \Delta t_{\text{transition}}. \tag{4.6}$$

The stiffness value will saturate when it goes beyond the set max limit. The hyperparameter $\Delta t_{\text{transition}}$ regulates the desired stiffness rate during the negotiation phase on whom has fully control of the task. If an external force was applied unintentionally, as long as the interaction was not longer than $\Delta t_{\text{transition}}$, then the impedance has not dropped entirely to zero and hence the passive mode is not activated. When the force drops again below the safety threshold, positive γ of Eq. (4.6) will ramp the stiffness back up to the maximum. This hysteresis time band helps to prevent unintentional switching from robot to human control (Hoque et al., 2021). Once the impedance on the trajectory the robot was following has become zero, the robot changes to passive mode. From now on, it keeps track of its proximity to the states it has stored in its model. The robot transitions back to active mode, i.e., γ becomes positive, when it detects itself in a state (other than the one it just came from) where it knows what action to take. The full algorithm is summarized in Algorithm 3.

4.3. Experimental Evaluation

We test our general framework on the pick&place task shown in Fig. 4.1. The cup can be moved to one of the other coasters, but our robot has no information on them or any prior on how it might move. For a parameterized behavior, knowledge of the environment, such as where the coasters are, would improve generalizability. But just for showing the use of interactive learning with disagreement-awareness, we test in a fixed environment, only using the end-effector position and external force data.

We asked five people to teach the same task of pick&place of Fig. 4.1[1]. Their expertise in robotics ranged from beginner to expert. The goal was to challenge the algorithm robustness with all possible interactions, from under to over-confident. The participants first showed the robot to place the cup on one of the other coasters, with an arbitrary number of intermediate states. Next, they altered the trajectory to pass through at least one additional or alternative state. At least once, they were asked to steer the robot to another coaster, a new goal state. Each participant was asked to disagree with the robot at least once, moving the robot to a different point in space, unknown and known, in each of the following ways:

[1] The study was performed as a pilot for the larger study conducted in Ch. 5, which was already approved by the Human Research Ethics Committee at the Delft University of Technology. A separate, retrospective ethics approval for this specific pilot study was granted by the same committee on 10/01/2024.

- moving the robot in a different direction w.r.t. the trajectory followed initially,
- stopping the robot on the trajectory it is executing,
- making the robot move over a state without stopping there, teaching it to move to a further lying state instead.

Figure 4.2 shows the position and force result of a disagreement case. A new state is learned on the executed trajectory. It is a typical force profile for all cases of disagreement. Cases in which a state is added in a different region in space generally only show a higher force peak. After the disagreement phase, the human is free to teach the robot a new state, which it registers when its motion is stopped. This is observed at $t = 96.7$s. After that, we see the the force on the end-effector increase again. Since the robot has arrived in a state it had not seen before, the user is performing a kinesthetic demonstration to show the robot what it should do the next time it arrives in the state that was just observed.

Of the few people we tested, the less experienced users struggled considerably more with deciding through which points in space to move and remembering them. While they could teach the robot the same things, they experienced increased difficulty. They tended to be more surprised when the robot would activate to start moving towards a state it had recognized as close. We let that be the robot's way of asking the human: "is this where you want to go?" But less experienced users reflexively let go, or at least did not immediately resist the robot, which the robot would interpret as confirmation, until the human would actively disagree again. This led to some confusion, stiffness going up and down and some additional interaction forces. However, when the users understood they were basically negotiating with the robot, they could successfully push their point and make the robot understand.

Figure 4.3 shows a state set that is learned with an inexperienced user. At the start, the robot only knows the state marked "t=0.0". Each state is marked with the time it was added to the robot's state set. The recorded end-effector trajectories are also shown in the figure. Both states and trajectories are color coded a lighter shade for each newly observed one. The figure shows the new states learned on the demonstrated trajectories. Changes in preferred state sequence could be demonstrated with smooth trajectories made possible by the smooth mode transitions, and once it was recognized that new behavior was being demonstrated, the robot lets the human demonstrate without interference. A video of the experiment, as well as our code, can be found in our GitHub repository[2].

4.4. Discussion

By responding to interaction forces by changing the impedance and sending haptic cues on model updates, there is two-way communication between the human and the robot in the physical interaction. This communication allows gradual mode switching between human and robot control without taking the human attention off the physical task, the way pressing a button would do. We expected this to make the interaction

[2] https://github.com/LindavdSpaa/DAVI_controller

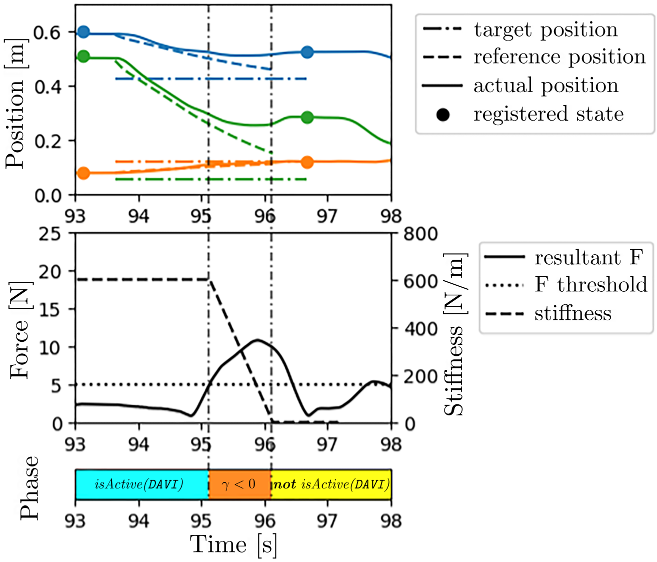


Figure 4.2: Position, resultant force and stiffness of the end-effector during an action that is corrected to a new position in space on the trajectory the robot was executing. The colors in the position plots indicate x (blue), y (orange) and z (green) respectively. The phase bar shows first the robot is in control. Upon detecting disagreement, the control transferred to the human.

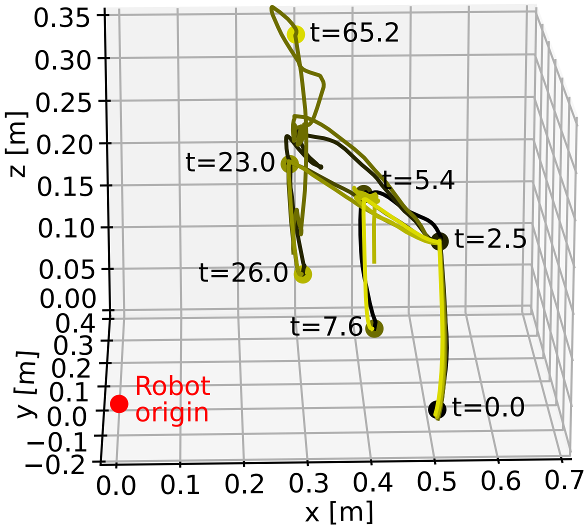


Figure 4.3: States and trajectories from which they were observed colored in order of learning from dark to light. Time stamps show when a state was added to the robot state space.

intuitive for users. Some of the users who tested our framework agreed. On the other hand, we also received the remark that switching at a button-press better disambiguates for the human when the robot is accepting the demonstration. A future study is necessary to compare and evaluate the intuitiveness of our implicit mode switching w.r.t. explicit switching, with a more representative group of subjects.

The comfort experienced with the mode switching and reactivation on model recognition varies with people's expectations and preferences. When and how fast (or slow) the robot responds currently depends on a number of preset variables. Ideally, these variables or a more general model defining the interaction and learning dynamics should be learned to match people's individual preferences.

In the current setting, the robot remembers every state it has seen from the moment we set it to start learning. At every point in its state space, it has stored a corresponding subsequent state it was shown to go to. For some participants, it was harder to remember the states and sequences they had taught the robot. Indeed, it may not always be desirable for the robot to remember all it has seen in the past. How and what to selectively forget is out of the scope of this study.

The disagreement detection in the presented implementation is based on a force threshold. Hence is would also trigger if a user is pushing in the direction the robot is going, e.g., to speed up the movement. This issue can be resolved by additionally considering the direction of the force. Similarly, for tasks requiring applying a force to an object or the environment, the disagreement detection will need to be modified to consider the difference to the expected force.

4.5. Conclusion and Future Work

With the presented framework, we showed how to smoothly transition between letting the robot execute the task and demonstrating alternative behaviors. At least, we concluded smoothness of the transition based on the observed trajectories such as the one shown in Fig. 4.3. Testing the subjective user perception remains future work and should be tested with a larger group of users.

By actively recognizing when the human is demonstrating, in contrast to only letting the human take the lead during execution (Khoramshahi and Billard, 2020), the task execution can be interactively corrected using the given human feedback.

For generalization, states can be parameterized with respect to objects' reference frames. We do this in the next chapter with a pre-defined states set. When learning the states set online, it introduces the additional complexity of solving the possible ambiguity in the selection of the right frame for the given goal (Franzese et al., 2020). Furthermore, the linear trajectory assumption defined in Eq. (4.2) can be relaxed and a nonlinear trajectory or a dynamical system can be learned during the kinesthetic teaching interaction (Franzese et al., 2021). For this reason, we believe that the presented framework opens up many further directions of future work, as it allows online learning of (parameterized) high-level pHRC policies on real hardware with real (non-expert) users in the loop, as we will show in the next chapter.

5

Simultaneously Learning Intentions and Preferences during Physical Human-Robot Cooperation

The advent of collaborative robots allows humans and robots to cooperate in a direct and physical way. While this leads to amazing new opportunities to create novel robotics applications, it is challenging to make the collaboration intuitive for the human. From a system's perspective, understanding the human intentions seems to be one promising way to get there. However, human behavior exhibits large variations between individuals, such as for instance preferences or physical abilities. This chapter presents a novel concept for simultaneously learning a model of the human intentions and preferences incrementally during collaboration with a robot. Starting out with a nominal model, the system acquires collaborative skills step-by-step within only very few trials. The concept is based on a combination of model-based reinforcement learning and inverse reinforcement learning, adapted to fit collaborations in which human and robot think and act independently. We test the method and compare it to a baseline that imitates the human, both in simulation and in a user study with a Franka Emika Panda robot arm.

The contents of this chapter have been slightly modified from the paper of the same name, by Linda van der Spaa, Jens Kober, and Michael Gienger, currently under review in *Autonomous Robots*.

5.1. Introduction

Physical human-robot collaboration (pHRC) is becoming increasingly popular, as it has the potential to increase flexibility and efficiency in industrial automation (Hanna et al., 2022) as well as support people in home environments (Fitter et al., 2020). To realize a fluent and intuitive collaboration, such novel robot systems should ideally be capable of understanding the intentions of the human partner, and to adapt their behavior accordingly. From a system's perspective, autonomously learning to interpret human intentions will make it easier and more intuitive for humans to engage in joint tasks, an important step towards cooperative intelligence (Sendhoff and Wersing, 2020). This requires a learning algorithm that is fast, needs little data, and learns in a way that is safe for both the robot and its environment.

In cooperation, the success of a task depends on the combination of what all actors do. Moreover, *how* a task is best completed depends additionally on the individual actors' preferences and the interaction dynamics. A team learning to work together needs to learn how the 'system' (including their colleagues) is responding. They need to learn how to follow/express their preferences within the bounds imposed by both the task and their teammates' preferences and capabilities.

This chapter addresses the challenge of enabling a robot to learn to cooperate with a human. In the setting we consider, the robot does not know the exact intention of the human and simultaneously attempts to act according to the human's preferences. We make an explicit distinction between *intentions*: *what* (sub)goal someone currently has, and *preferences*: *how* the person likes to approach the (sub)goal. Figure 5.1 shows two example scenarios: The robot needs to learn how to help the human move the object, a wheel or a clothes hanger, to the goal intended by the human.

We consider the problem on the abstract level where the two agents (human and robot) have pre-learned/programmed skills, e.g., *grasp the object*, or, *pull towards x, y, z in space*. Compliant control lets the success of actions depend on the physical interaction. This allows us to focus on the learning problem in this chapter, without simultaneously having to consider the specific mechanics of physical human-robot interaction.

This chapter's first contribution is a novel method for learning a human preference model for intention-aware cooperation, from collaborative episodes. The method 1) learns a personalized model of a human partner from physically cooperating with this partner, from scratch or improving a nominal model; 2) models human preferences as an explicit function of intention, enforcing inherent intention awareness; 3) applies second order Theory of Mind (ToM) reasoning to model the human's preferences separate from the robot's, resulting in explicit partner awareness. This allows the robot to optimize an objective different from the human for improved cooperative behavior. The process is iterative: after each collaborative episode, the robot updates its internal models based on the observed partner response and the intention observed in hindsight at the end of the episode. As its internal models improve, so does the robot's response. Since most optimization is done internally in the modeled environments, the robot requires very few experimental episodes for learning. We achieve this by combining existing Reinforcement Learning (RL) and Inverse Reinforcement Learning (IRL) methods in a novel way.

Secondly, we contribute by testing our method in a user study with a diverse group of

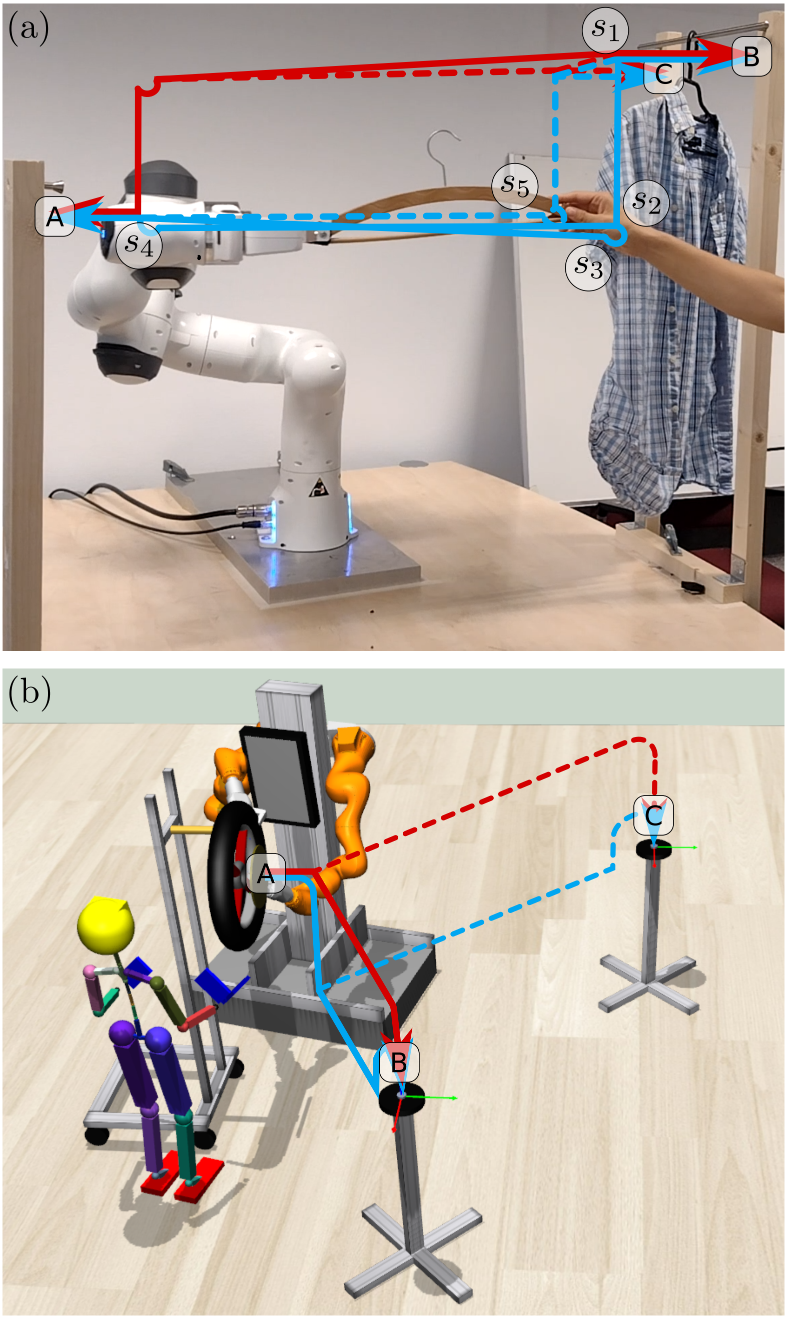


Figure 5.1: Two cooperative scenarios: The robot needs to learn how best to assist the human to move the object between support points A, B, and C. Two colors of arrows indicate different paths along which the human may prefer to move the object (the wheel on the left (a), the clothes hanger on the right (b)). The dashed lines indicate how the preferences may generalize when the goal is different.

mostly novice users. We compare our "Learner" to an "Imitator" baseline which lets the robot merely imitate its partner. Users were free to choose their preferences (within the limits of the setup). We evaluate the user experience and the performance both quantitatively and qualitatively. In simulation, we additionally try our method in a scenario with increased complexity for further evaluations and insights for directions of future work.

Sec. 5.2 discusses the related literature. Then, the method is presented in Sec. 5.3. Implementation considerations are discussed in Sec. 5.4. We describe the scenarios shown in Fig. 5.1 in Sec. 5.5, on which we evaluate our method's performance in a user study in Sec. 5.6, and in additional simulations in Sec. 5.7, before we conclude in Sec. 5.8.

5.2. Related Work

Before we present our method to learn a behavioral model of a human partner for improved intention-aware planning, we will first discuss relevant literature in the three main directions related to our work: intention-aware planning, behavioral modeling, and model learning.

5.2.1. Intention-aware planning

Literature on intention estimation for human-robot cooperation (HRC) tends to fall into one of the following three categories: (sub)goal estimation – predicting which (sub)goal out of a set of possibilities the human is trying for (Karami et al., 2009; Malik et al., 2018); action prediction – predicting which (primitive) action the human will take next (Belardinelli et al., 2022; Gienger et al., 2018; Hawkins et al., 2014); motion extrapolation – predicting how fast the human will continue in which direction (Bai et al., 2015; Duchaine and Gosselin, 2007), or along which trajectory (Park et al., 2019; Ranatunga et al., 2015).

The last category is useful for collision avoidance (e.g., to independently navigate the same environment (Bai et al., 2015; Park et al., 2019)), and for motion following (e.g., steering a single tool (Duchaine and Gosselin, 2007; Ranatunga et al., 2015)). More abstract level planning needs higher-level action predictions. On the top level, an estimate of the goal the human wants to reach will allow a robot to plan further ahead. Somewhere in between are reaching and placement tasks, where the intention encodes both the motion and the goal (Koert et al., 2019), and, in the pHRI case, the interaction forces (Haninger et al., 2022; Lai et al., 2022).

Instead, we consider tasks consisting of a chain of actions, and different possible goals can each be reached in multiple ways. We seek to learn human preferences in (physical) cooperation while we have no direct access to the human partner's intention. Similar to Koppula et al. (2016) and Park et al. (2019), we define the problem as a Markov Decision Process (MDP). Intentions can be incorporated as a 'hidden state', resulting in a Partially Observable MDP (POMDP) (Bai et al., 2015; Karami et al., 2009), or a Mixed Observability MDP (MOMDP) (Ong et al., 2009). This definition allows us to use standard techniques for learning a fitting robot policy, determining when it will take which action given the observations.

For robot-robot cooperation in the MDP domain, Multi-Agent Reinforcement Learning (MARL) techniques have been derived from single-agent techniques (Buşoniu et al., 2010). Some of those methods could be applied to human-robot cooperation problems, but have the disadvantage of requiring a large number of trials which is impracticable for learning in interaction.

5.2.2. HUMAN BEHAVIOR MODELING

For directed cooperation, a robot needs a model of the agents with whom it should cooperate (Choudhury et al., 2019). Agents can be modeled by a black box model, such as a neural network (Schmerling et al., 2018; Zyner et al., 2019). Although such a model can give accurate predictions, collecting sufficient representative data in a pHRC scenario is expensive from a human perspective. More recently, Shih et al. (2022) and Parekh et al. (2022), Wang et al. (2022), and Xie et al. (2021) solved this by learning a low-rank latent space in different ways from few demonstrations which allows for interpolation to predict previously unseen partner policies or strategies respectively. Shih et al. (2022) and Parekh et al. (2022) show the effectiveness of the approach with human subjects. Nevertheless, these models lack a structure providing more detailed insight into how the prediction is obtained, which makes it non-straightforward to allow for goal uncertainty induced by the partner such as the intentions we consider. As the methods do show great promise, it is an interesting direction for future work to research how this approach could incorporate hidden but leading partner intentions.

Alternatively, gray box models have a structure which offers insight into the prediction process and increases data efficiency, if a proper structure is provided. A simple single parameter can already improve a robot's cooperative skills (Nikolaidis, Hsu, et al., 2017). More complex structures may be derived from dynamics (Stouraitis et al., 2020) or from Theory of Mind (Choudhury et al., 2019). ToM originates from the fields of psychology and philosophy (Baker and Tenenbaum, 2014) and reasons about the reasoning of others. For example, a robot may model a human as an agent with its own internal model of the task and the world. When such a model includes the human's reasoning about the robot's reasoning about them, etc. (infinite regress), it is no longer practical. Successful implementations limit the regress to one or two levels (Buehler and Weisswange, 2018; Malik et al., 2018; Sadigh et al., 2016). ToM can be considered as an IRL problem (Jara-Ettinger, 2019). We follow this example, using IRL to learn a mental model of the human partner, which we can then use to optimize our robot's collaborative actions.

5.2.3. INVERSE REINFORCEMENT LEARNING

Inverse Reinforcement Learning focuses on inferring the underlying reward function from demonstrated samples. However, the problem of reward reconstruction is ill-posed: more than one reward function could describe the same demonstrated policy. Maximum Entropy IRL (ME-IRL) offers a solution to this problem which is the least biased on the demonstrations (Zhifei and Joo, 2012; Ziebart et al., 2008). Derived methods have been applied successfully to learn from non-expert data (Boularias et al., 2011) or incrementally update the model as data comes in (Jin et al., 2011; Rhinehart and Kitani, 2018), or both and from physical interaction (Losey et al., 2022).

In Cooperative IRL (CIRL) as described in Hadfield-Menell et al. (2016), the human and the robot optimize the same Q-function. This is also the case in Malik et al. (2018), where the human and the robot are modeled as different actors. Instead, we explicitly treat our robot and human as independent agents by learning/keeping separate reward functions for each, without giving up on the overall cooperative objective. This has been more common in autonomous driving (Mehr et al., 2023; Peters et al., 2023; Schwarting et al., 2019) following dynamic game theory, but not yet in pHRC.

5.3. Method

In order to optimize the robot response in cooperation, we learn a model of the human partner's behavior, including their response to the robot. We break this loop into three interconnected learning processes, indicated by the ellipses in Fig. 5.2. We start off with a nominal (safe) robot policy π^{R} and an initial estimate of the human reward function R^{H}. After every episode we try the collaborative task, we update our human reward estimate on the observed human actions in ζ and intention ι. To the human reward estimate, we apply RL to compute the most likely human response $\hat{\pi}^{\mathrm{H}}$, which we then use to compute an improved robot response π^{R}. Thus, we iterate.

A key element of the presented method is the explicit modeling of the human's intention, a variable which is not directly observable but assumed to uniquely define a person's response. The intention ι is a discrete variable. We assume the set of possible intentions $\mathscr{I}$ is known. The human preference model, captured in R^{H}, is a function of this intention, which allows it to be inferred by comparing the model to the observed actions. A real-valued parameter vector is updated in R^{H} after every episode to improve the feature match to the observed paths from the start to the intended goal.

To summarize, the *preferences* are captured by a human reward function R^{H} learned through IRL, while the *intentions* are captured by a variable ι that the robot cannot access directly but needs to infer from observed human actions.

We consider a discrete state-action space, where the state $\boldsymbol{s}$ is the combined state (for the robot, human, objects, environment, etc.) and we have separate actions for the robot a^{R} and the human a^{H}. The states and actions we employ in our experiments are detailed in Sections 5.4 and 5.5. As the model improves, so does the robot's response, decreasing cooperation effort.

First, Sec. 5.3.1 briefly recaps the necessary background on MDPs, Q-iteration, and soft-max policy optimization. Sec. 5.3.2 explains how these concepts have been modified to fit our collaborative case with hidden intention. Sec. 5.3.3 briefly discusses ME-IRL and its application to our multi-agent learner before Sec. 5.3.4 summarizes our algorithm.

5.3.1. MDPs and their model-based solution

An MDP is defined by the tuple $\{\mathscr{S}, \mathscr{A}, T, R, \gamma\}$, consisting of a state space $\mathscr{S}$ containing states s, action space $\mathscr{A}$ containing actions a, transition model $T(s' \mid s, a)$, reward function $R(s, a, s')$ and discount factor $\gamma \in [0, 1)$. In the model-based case, where the entire tuple is available, the value indicating the desirability of each state-action pair can be computed via Q-iteration:

$$Q(s,a) \leftarrow \sum_{s' \in S} T(s' \mid s, a) R(s, a, s') + \gamma V(s'), \tag{5.1}$$

with value function $V(s) = \max_{a \in \mathscr{A}} Q(s, a)$.

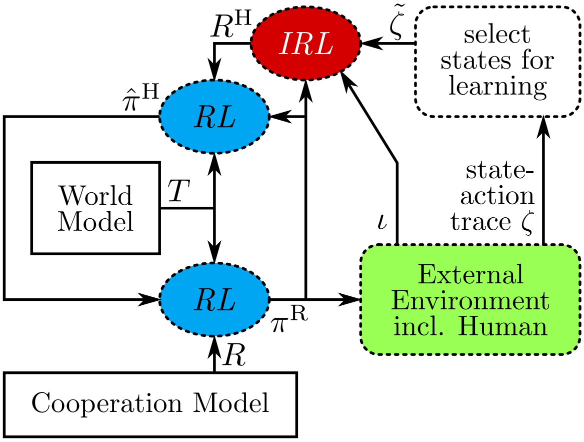


Figure 5.2: Method overview, showing the learning processes in ellipses, other processes in dotted-lined rounded rectangles, models in solid-lined rectangles, and functions and data on the arrows. The human preference model R^{H} is updated by the IRL process based on observed state sequence ζ and intention ι. The two RL processes compute human policy estimate $\hat{\pi}^{\mathrm{H}}$ and robot policy π^{R}.

The Q-function is a sound basis for extracting a policy $\pi(a \mid s)$ an agent can use to decide which action to take in a state. We select our policies by taking the weighted softmax as described by Tijsma et al. (2016):

$$\pi(a_i \mid s) = \frac{e^{\tau Q(s,a_i)}}{\sum_a e^{\tau Q(s,a)}}. \tag{5.2}$$

The exponential relationship between an action's Q-value and its probability to be selected results in directed exploration around the optimal policy. Exploration can be decreased by weighting the Q-values by a temperature parameter $\tau \geq 1$. A small amount of directed exploration tends to speed up learning. It will mitigate modeling errors in cases multiple actions come up with similar values and which one shows up best depends heavily on an inaccurate model. This may very well happen in our case, since all internal MDPs depend on the human preference model, which is being learned.

For the robot, we do restrict exploration with a lower bound on the acceptable action Q-value: $\eta \max_{a^{\mathrm{R}}} Q^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}})$. This way, we prune potentially bad actions. Additionally, the bound can be set to only allow actions with a value at least as high as a baseline deterministic policy.

5.3.2. MULTI-AGENT POLICY OPTIMIZATION WITH HIDDEN INTENTION

In our collaborative case, there are two necessary adaptations if we are to use the MDP principles of the previous subsection. First, we need to account for a collaborative partner whose actions we assume we cannot control. Second, one state variable—this partner's intention—is hidden for the robot. We assume the human knows their own intention, so we treat it as a regular state variable within the human model. However, the robot does not know this true intention, and we hence need to maintain the uncertainty over it in the robot model.

To solve the first problem, we extract the single-agent transition function for the agent we are interested in from the combined transition function $T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}})$, where a^{R} is the robot's action and a^{H} the human's. This is done by substituting the partner policy. For the robot transition function T^{R} we replace a^{H} by an estimate of the human policy $\hat{\pi}^{\mathrm{H}}$, resulting in $T^{\mathrm{R}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{R}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota))$. For the human transition function T^{H} we replace a^{R} by the robot policy π^{R}, resulting in $T^{\mathrm{H}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{H}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, \pi^{\mathrm{R}}(\boldsymbol{s}, \iota), a^{\mathrm{H}})$. Note that the single-agent transition function is a function of intention ι because the partner policy depends on the intention.

For predicting the human policy π^{H}, we need an estimate of how the human perceived the robot policy π^{R}, hence a second order ToM. We cut the regression by using the most recent robot policy. It is an overestimate of what the human can know, but it is the best we have. If we would assume instead that the human models the robot as a random agent, the human would be modeled without any trust in the robot policy, which will make it much harder to learn policies that actually rely on the robot taking a certain action. Since our robot is learning, we cannot model human learning of the robot reward as in Nikolaidis, Nath, et al. (2017), nor can we follow Tian et al. (2023) and disregard the large effect of our robot's actions on the discrete state transitions within the human model. Modeling how the human partner would learn to trust the robot is out of scope of the current chapter, although interesting to explore in future work.

5

The resulting human policy estimate $\hat{\pi}^{\mathrm{H}}(a^{\mathrm{H}} \mid \boldsymbol{s}, \iota)$ is used to obtain the robot transition function. The robot reward function $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}}, \boldsymbol{s}')$, additionally depends on the human actions to explicitly encode cooperation objectives. Here, $\hat{\pi}^{\mathrm{H}}$ is substituted in the same way as in the robot transition function T^{R}, resulting in $R^{\mathrm{R}}(\boldsymbol{s}, \iota, a^{\mathrm{R}}, \boldsymbol{s}') = R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota), \boldsymbol{s}')$.

The human policy estimate is a function of the intention, which the robot cannot observe directly. We assume that the human acts consistently under a given intention, which enables the robot to infer the intention from observations of the taken human actions. Since it is only one-dimensional and very small-sized, it is computationally feasible to resolve this second problem by computing the MDP and its solution for each possible intention $\iota \in \mathscr{I}$. For larger problems, we advise to adapt a MOMDP solver (Ong et al., 2009).

At runtime, a belief distribution is estimated over the possible intentions, using a Bayes filter:

$$b(\iota') = C\hat{\pi}^{\mathrm{H}}(a^{\mathrm{H}} \mid \boldsymbol{s}, \iota') \sum_{\iota \in \mathscr{I}} P(\iota' \mid \iota) b(\iota), \tag{5.3}$$

with normalizing constant C. The intention transition probability is the likelihood of the observed human action combined with the chance of keeping or changing the intention:

$$P(\iota' \mid \iota) = \begin{cases} \beta, & \iota' = \iota \\ \frac{1-\beta}{n-1}, & \iota' \neq \iota \end{cases} \tag{5.4}$$

with 'intention bias' $\beta \in [\frac{1}{n}, 1]$ and n possible intentions. The closer β is chosen to 1, the harder it is for the robot to understand, and thus adapt to, the situation when the estimated intention does not match the human's. This may happen because the human changed intention, or the estimate may have been wrong because of errors in the learned model. Smaller β results in faster robot adaptation (at runtime), but too small a β makes it impossible for the robot to effectively exploit its intention parameterized internal model.

Having the belief estimate, the final robot Q-function is obtained by superposition (Schweitzer and Seidmann, 1985):

$$Q^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}) = \sum_{\iota \in \mathscr{I}} b(\iota) Q^{\mathrm{R}}_{\iota}(\boldsymbol{s}, a^{\mathrm{R}}). \tag{5.5}$$

5.3.3. IRL HUMAN MODEL UPDATES

The IRL objective is to maximize the total reward of the optimal trajectory ζ^*. The reward

$$\sum_{s_j \in \zeta^*} R(s_j) = \boldsymbol{\theta}^T \boldsymbol{\phi}_{\zeta^*} = \boldsymbol{\theta}^T \sum_{s_j \in \zeta^*} \boldsymbol{\phi}(s_j) \tag{5.6}$$

is a linear combination of the features $\boldsymbol{\phi}$ observed in the trajectory, weighed by $\boldsymbol{\theta}$. Expert demonstrations $\tilde{\zeta}_i$ are assumed representative for the optimal trajectory. ME-IRL maximizes the log-likelihood of the observed trajectories (Ziebart et al., 2008):

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\arg\max}\, L(\boldsymbol{\theta}) = \underset{\boldsymbol{\theta}}{\arg\max} \sum_i \log P(\tilde{\zeta}_i \,|\, \boldsymbol{\theta}, T), \tag{5.7}$$

This can be solved by gradient descent $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \lambda \nabla L$, where ∇L equals the difference in feature counts between the observed trajectories and the expected feature counts according to the model. The expected feature counts are computed by internal soft-max Q-iteration, using the human transition model T^{H}. Here, we only consider the intention observed during the episode. Like other incremental IRL methods (Jin et al., 2011; Rhinehart and Kitani, 2018), we perform a single gradient descent update after each episode.

In our interactive case, we must be somewhat selective in providing demonstration data to the learning algorithm. States are now visited because of both the human and the robot action. If the robot made a wrong choice and the human had to wait or correct, this should not be interpreted as optimal, just because it was observed. To resolve this, any loops between states that are visited multiple times are assumed to be caused undesirably by inexperience, and are therefore removed before updating the human model.

5.3.4. THE COMBINED ALGORITHM

Algorithm 4 shows the full method. After initialization of the human model (L. 2), the learning loop starts. The robot models are extracted (L. 4–5) and the Q-functions are optimized per intention (L. 6). During a cooperative episode (initialized in L. 7), the robot Q-values are computed for the current state (L. 9). The policy in the current state is obtained (L. 10) using the bounded soft-max discussed in Sec. 5.3.2. When the robot and the human have performed their actions and the state is updated (L. 11), so are the robot belief (L. 12) and the state-action trace ζ (L. 13). The episode continues until a goal is reached (L. 14), then the state-action trace and the human intention (the reached goal) are returned to the model environment (L. 15). States are selected for learning (L. 16) and the human transition model is extracted (L. 17). The IRL step updates the feature weights of the human preference model (L. 18) using the state-action sequence observed during the latest episode and the human transition model given the observed intention. The human reward model (L. 19), Q-function (L. 20), and policy estimate (L. 21) are updated, and the cycle repeats.

Algorithm 4: Learning human-aware cooperation

```
Given: transition model $T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}})$, robot reward $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}}, \boldsymbol{s}')$, features $\boldsymbol{\phi}(\boldsymbol{s}, \iota)$
Initialize: human reward $\boldsymbol{\theta} = \boldsymbol{\theta}^0$, $R^{\mathrm{H}}(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$, policy estimate $\hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota) \leftarrow \mathcal{U}(\boldsymbol{s})$
while *True* do
    $T^{\mathrm{R}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{R}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota))$
    $R^{\mathrm{R}}(\boldsymbol{s}, \iota, a^{\mathrm{R}}, \boldsymbol{s}') = R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota), \boldsymbol{s}')$
    $Q^{\mathrm{R}}_{\iota}(\boldsymbol{s}, a^{\mathrm{R}}) \leftarrow \textsc{QIter}(T^{\mathrm{R}}, R^{\mathrm{R}})\ \forall \iota \in \mathcal{I}$
    Initialize: initial state $\boldsymbol{s}_0$, belief $b_0(\iota) \leftarrow \mathcal{U}(\iota)$, state-action trace $\zeta \leftarrow \emptyset$
    while *Collaborative Episode* do
        $Q^{\mathrm{R}}(\boldsymbol{s}_t, a^{\mathrm{R}}) = \sum_{\iota} b(\iota) Q^{\mathrm{R}}_{\iota}(\boldsymbol{s}_t, a^{\mathrm{R}})$
        $\pi^{\mathrm{R}} \leftarrow \textsc{BoundedSoftMax}(Q^{\mathrm{R}})$
        $\boldsymbol{s}_{t+1}, a^{\mathrm{H}}_t, a^{\mathrm{R}}_t \leftarrow \textsc{DoAction}(\pi^{\mathrm{R}})$
        $b_{t+1} \leftarrow \textsc{UpdateBelief}(b_t, \boldsymbol{s}_t, a^{\mathrm{H}}_t)$
        $\zeta \leftarrow \textsc{UpdateStateActionTrace}$
        if $\textsc{isGoalState}(\boldsymbol{s})$ then
            return $\zeta, \iota^{\mathrm{H}}$
    $\tilde{\zeta} \leftarrow \textsc{SelectStatesForLearning}(\zeta)$
    $T^{\mathrm{H}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{H}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, \pi^{\mathrm{R}}(\boldsymbol{s}, \iota), a^{\mathrm{H}})$
    $\boldsymbol{\theta} \leftarrow \mathrm{IRL}(\tilde{\zeta}, T^{\mathrm{H}}(\iota^{\mathrm{H}}), \boldsymbol{\theta})$
    $R^{\mathrm{H}}(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$
    $Q^{\mathrm{H}}(\boldsymbol{s}, \iota, a^{\mathrm{H}}) \leftarrow \textsc{QIter}(T^{\mathrm{H}}, R^{\mathrm{H}})$
    $\hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota) \leftarrow \textsc{SoftMax}(Q^{\mathrm{H}})$
```

5.4. Implementation

We test our method in two different scenarios in which a human and a robot cooperatively need to move an object from one support to another. The robot knows where the supports are, but not which one the human intends to move to. This section describes how we model such scenarios as an (MO)MDP for learning. In the final subsection (Sec. 5.4.6), we describe the baselines we compare our learning method to. The code to this research can be found in our GitHub repository[1].

5.4.1. States

The physical states $\boldsymbol{s}$ are defined from the perspective of the manipulated object, defining its position $\boldsymbol{p}$, orientation $\boldsymbol{q}$, and affordance (Koppula et al., 2016)—in our case its manipulability μ: $\boldsymbol{s} = \begin{bmatrix} \boldsymbol{p}^T & \boldsymbol{q}^T & \mu \end{bmatrix}^T$. We consider positions in 3D (x, y, z). Depending on the scenario, $\boldsymbol{q}$ can be a single angle or a quaternion. The manipulability defines how the object may be moved depending on by whom it is held (e.g., the object may only be moved if it is held by both human and robot). Concretely, μ is an integer encoding who is holding the object.

The object must be held by both human and robot anytime it is not resting on a

[1] https://github.com/LindavdSpaa/learning_collaborative_preferences

Table 5.1: Actions and their necessary state conditions.

Action	State pre-conditions
wait/passive	
grasp	object resting, not held by actor
let go	object resting, held by actor
take off (support)	object resting, held by both actors
put on (support)	object at *mounting point*[2]
rotate[3]	object held in free space
move over[4]	object held in free space
move up/down[5]	object held in free space

support, which may be anything that will keep the object in a stable position without the help of an actor. Each support provides a possible start or goal state, with a specific object position and orientation. The human may intend to put the object on any of these supports.

In between these supports, a small number of strategically chosen waypoints define key locations in space. Examples are the position from which to mount the object onto a support – which is assumed to be the same as the position to which the object can be unmounted – or a position below such a "mounting point" at a height which is comfortable for the human to carry the object. The space in between waypoints is assumed to be free of obstacles.

Next to the physical state $\boldsymbol{s}$, there is the human intention ι encoding the desired goal to put down the object. This may be any of the available supports. The robot has no direct access to this variable. The initial intention estimate is set to zero at the initial support and distributed uniformly for the others.

5.4.2. ACTIONS

The general set of high-level actions and their pre-conditions are listed in Table 5.1. It depends on the state which actions are allowed. In free space, where the object is only supported by the robot and the human, neither is allowed to let go. Rotation is allowed around a single axis at a time. Movement between waypoints is allowed either horizontally *or* vertically, along straight-line trajectories. We made this choice purely for demonstration purposes, to define easily distinguishable possible preferences while keeping the state space small. From and to a support, the motion is defined based on the geometry of the object and the support.

In simulation, we only consider the discrete states connected by the abstract actions. On the hardware, the way the robot grasps and lets go of the object is pre-programmed.

[2] next to support, oriented correctly
[3] around a single axis
[4] to waypoint at same height
[5] to waypoint at same (x, y)-coordinate

The other actions are defined along straight-line trajectories, either in linear or in rotational space, and tracked applying disagreement-aware variable impedance (DAVI) control (Ch. 4). In order to track the straight-line trajectories between our states in a robust way with our 7 DoF robot arm, we extend the DAVI controller with the following null-space component: We train a Gaussian Process (GP) (Williams and Rasmussen, 2006) on a small set of feasible arm configurations (one per gripper pose) to obtain a consistent reference for the redundant degree of freedom. During actions, we control both the gripper position and orientation (6 DoF) and the joint configuration (7 DoF), one set of DoF with lower impedance than the other to resolve the redundant control. Close to state positions and orientations, the Cartesian impedance on the gripper dominates the joint impedance, to make sure the robot reaches the state. As the distance to the known states increases, so does the impedance on the joints, while the Cartesian impedance on the gripper is reduced. This way we smoothly bend our straight-line trajectories a little bit to avoid joint limits and allow the elbow of the robot arm to change side when necessary.

The robot always has the option not to act. In this "passive mode", the robot just compensates the gravity with zero stiffness and the human is free to drag the robot around with the object.

5.4.3. Transitions

The abstract physics of the problem, considered by the internal model, are simple: an object can be moved if both actors have a grasp on it. A robot action may either have the desired effect or no effect at all, if the human counteracts the action. The DAVI controller ensures smooth transitioning from active to passive in case of counteractive action, so the human is always in control of where the object is moved to.

If the robot is in passive mode, the human fully determines the transition. The human can also choose to passively follow the robot. Human partners are instructed only to do actions which end up in a valid next state.

5.4.4. Observations

In simulation, the abstract states and actions are observed directly. In the robot experiment, the closest abstract state is considered to be the state arrived at. This state is used to infer the action the human took to realize the transition, and to estimate the next abstract human action and choose a matching abstract robot action. The corresponding motion trajectory is always planned from the actual robot position and orientation, not from the abstract one the robot is expected to be at.

At the supports, there is the additional bound that the actual position should be close to the expected one. Furthermore, the velocity and interaction forces must be close to zero before letting go of the object is allowed, assuming the human will stop trying to move the object when it is stably supported, and using that as a sign that it is safe for the robot to let go.

In the experiment, the robot does not directly observe the human grasp on the object. Instead, we generally assume the human is holding the object, until the object has been non-moving for a number of seconds. We instruct our users to keep a hold on their end of the object when the robot is active on the task, and make sure not to activate the robot when they have not, so that the assumption holds.

5.4.5. REWARDS

As described in Sec. 5.3, we have separate reward functions for the human and the robot. The human reward, $R^{\mathrm{H}}(\boldsymbol{s},\iota)$, is defined by the learned preference model (Sec. 5.3.3), which is a function of features describing each state's relation to the start, the intended goal, and unintended alternative goal candidates. The features in the human preference model are the product of two Gaussian Radial Basis Functions (RBFs) and a binary component. The first set of RBFs are a measure of linear distance and are centered at points defined relative to the intended goal support, another support, or the world (e.g., comfortable carrying height). These points cover the waypoints, but multiple waypoints relative to different supports map to the same feature point if the supports are not the intended goal. The second set of RBFs are a measure of angular distance and are centered at the allowed absolute orientations and at the final intended orientation. The standard deviations of the RBFs are chosen at 2 cm and 10° respectively. The binary component indicates the manipulability. With our choice of waypoints, this results in a total of 26 features. The feature vectors are normalized per state, $\boldsymbol{\theta}$ is scaled to $-1 \leq \boldsymbol{\theta}_i \leq 1$, and initialized at 0.1 at the intended support, -1 at the other supports, and 0 elsewhere.

The reward the robot receives for state transitions, $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}}, \boldsymbol{s}')$, punishes actions which are counteracted by the human, or do not change the state, by $r^{-} = -1$. Passive behavior, when a supportive alternative exists, is punished less severe, by $r^{0} \in (-1,0)$. The reward factor r^{0} is deciding for the robot behavior. A smaller magnitude makes passive robot behavior more desirable as, relatively, the punishment for choosing a wrong action increases: the robot is "more afraid" of taking a wrong action. If $r^{0} = r^{-}$, the robot does not care about taking wrong actions and the benefit of learning an internal model disappears. The effect of different values of r^{0} is evaluated in Sec. 5.7.

5.4.6. BASELINE AGENTS

Both in simulation and in the real-world experiment, we compare our *Learner* agent to 1) an *Imitator* and 2) a *Passive* agent. In the user study, we added an additional ablation study, comparing to 3) a *plain ME-IRL* agent.

The Passive agent is hard coded to grasp at the start and let go when the object rests at a support. In between it just compensates the gravity, i.e., is in "passive mode". This passive policy is also the internal baseline the robot compares its actions to when computing its policy.

The Imitator agent follows the passive policy in states where it has yet to observe a human action. Otherwise, it takes the action it has observed most recently. This can capture most of the preference, but because it has no notion of intention, it will always have a chance of $\frac{n_\iota - 1}{n_\iota}$, with n_ι the number of possible intentions, of choosing wrongly in the deciding state. If the human decides to return the object to the start support, the Imitator will not understand and keep trying to move elsewhere (if it observed an action in that state before, coming from the same start support). The way we defined the Imitator, allowing the start support to be also a goal support would mean that the robot will not hold on to the object to let the human move it away from the start, as it does not have an internal model to consider that option.

The plain ME-IRL agent learns its policy applying plain ME-IRL on the observed trajectories to learn its reward function, without having an explicit internal model of the human. This is similar to Losey et al. (2022), except for that we update the model only at the end of each episode, as we cannot observe the human intention (where the human wanted to go) before observing the final state of the episode. We use the same intention-parameterized features as for our Learner, nor did we change any of the other parameters. We initialize the feature weight as in the human model of our Learner. A maximum likelihood estimate of the intention is obtained based on how often each intention occurred in the current state given the start state. A more intelligent estimate would improve the agent's behavior. However, designing such an intention estimate is not the topic of this chapter. It could be interesting for future work.

5.5. Scenarios

We test our learning algorithm in two different cooperative scenarios. The scenario of moving a clothes hanger (Fig. 5.1a) has a state space that allows human users a number of different preferences while moving a clothes hanger between three possible supports (intentions). We designed this scenario such that we could run it on a Franka Emika Panda robot arm, to test our algorithm in a user study.

The scenario of moving a wheel between stands (Fig. 5.1b) has a larger state and action space, that allows human preferences to include seemingly inefficient detours. In this scenario, we test our algorithm only in simulation. In simulation, we can also easily test the generalization to cases where the stands change position and height.

In both scenarios, we use the following learning parameters: For the iterative IRL, a learning rate $\lambda = 0.1$ is used. The robot action exploration is restricted by a soft-max temperature $\tau^{\mathrm{R}} = 5$, and, for the robot, with an additional bound $\eta = 0.9$. The human is assumed to explore even less, $\tau^{\mathrm{H}} = 25$. The intention bias is chosen at $\beta = 0.95$.

5.5.1. Clothes Hanger Scenario

In the "Hanger Scenario", we use a quaternion to define the object(=hanger) orientation. We consider just a single rotation, around the vertical axis. There is no reason to not hold the hanger with the hook on top, but the peg (A) (Fig. 5.1a) we can hang it on is oriented differently than the rail on which we have our support points B and C.

Supports B and C are at the same height, support A is considerably lower. To each of the supports, there is a mounting point, a bit over a hanger 'radius' away in 'unhooking' direction, so that the hanger is sufficiently clear to be rotated. In between these mounting points, we define additional waypoints in space by recombining their (x, y) and z positions. With only the two distinct heights, there are 24 states and between 2 and 6 actions per state, including not acting.

We set $r^0 = -0.33$, which gives us balanced behavior: reasonably careful not to take wrong actions, yet not too afraid to act. For learning, we use discount factor $\gamma = 0.9$.

5.5.2. Wheel Scenario

In the "Wheel Scenario", we describe the object(=wheel) orientation by a single angle, around the axis pointing from the robot to the human. The wheel can hang 'vertically' on the rack, or be placed 'horizontally' on one of two stands (Fig. 5.1b). The affordance μ, whether the robot and the human have a grasp on the wheel, is considered to be directly observable.

The rack and the two stands are all at different heights, respectively at 1.7, 1.0 and 1.2 m. Every episode, we initialize the positions of the stands at random, at a distance between 1.6 and 3.6 m from the rack and between 1.0 and 2.2 m from each other. Intermediate waypoints are defined as in the *hanger scenario*. Additionally, we define a “comfortable carrying height”, at 0.95 m, below each mounting position. When working with real people, this height should be adjusted according to how tall the user is. If a point in space would collide with a stand, the point is projected in negative x-direction by a bit over a wheel radius distance.

Moving up or down to different heights, are all separate actions. At each height, there is the possibility to move over towards each of the other supports. All actions, of both robot and human, are assumed to be directly observable by the robot. In total, there are 36 states and up to 8 possible actions per state.

We test different r^0. To better allow our human model to capture detour preferences, and the robot model to support it, we lower our discount factor to $\gamma = 0.6$. As the human model is learned from demonstrations that reach a goal, and the robot receives punishment for not supporting the human, the learned policies still terminate releasing the wheel at a support, despite the low discount factor.

5.6. User Study: Clothes Hanger Scenario

5.6.1. Experiment

We did a user study with a Franka Emika Panda robot arm and 24 users (16 male, 8 female) of an age between 19 and 77 years old, with the median at 28 and the interquartile range between 25 and 35. Five of the participants had participated before in a user study involving a similar robot arm; one participant had multiple years experience with collaborative robot arms including the Franka Emika Panda, although not in a setting that involved physical interaction; one other participant had experience programming industrial robot arms; and there was one participant with experience with physical human-robot collaboration in terms of lane-keeping assistance.

The clothes hanger scenario was explained to the participants, including that the robot would never be given the information of where the users were asked to hang the hanger next (i.e., the intention). The users were informed that the robot could perform only a few distinct actions between the supports and six distinct points in the intermediate space.

All participants went through the same familiarization phase, in which they first moved the hanger around with the robot in passive/gravity compensation mode. Next, the robot would play a pre-programmed sequence of actions, letting the human follow and feel how it feels when the robot is maximally assistive. Then, the users were asked to follow the same sequence of motions they had observed the robot to lead previously, but this time with the robot trying to move elsewhere in each of the decision points in space. This way, the users would get comfortable disagreeing with the robot in case it would not follow their preference or intention. The participants could try each of the ‘modes’ until they felt comfortable with whatever the robot would do during the actual experiment.

Now that the users felt somewhat familiar with the task and the robot, they were

Table 5.2: Experimental episodes

Nr.	from, to	remarks
1	B, A	initial behavior
2	A, B	
3	B, A	for the second time
4	A, C	starting from A with different intention
5	C, B	new region in the state space
6	B, C	starting from B with different intention
7	C, A	starting from C with different intention
8	A,(B)A	changing intention: turning back halfway
9	A, B	starting from A like in Episode 2

asked to specify their preference, segmenting the movement to the lower and rotated support in {moving over horizontally, moving down, rotating} in the order of their choice, as well as the way back. During the remainder of the experiment, they were instructed to stick as closely to this preference as they could manage, no matter what the robot would do.

The actual experiment then consisted of moving the hanger nine times to a next hanging point (as listed in Tab. 5.2), while the robot would update its internal model in between. The whole sequence took between 2.5 and 4 minutes. This was done once with the robot applying the proposed IRL method, and once running an imitator baseline. Half of the users experienced the Learner first, half of them the baseline, approximately alternating between participants, to average out the learning effect of the users. After each set of learning episodes, the users filled out a questionnaire on what they felt about the robot learning (on a 7-point Likert scale), and the NASA TLX questionnaire to assess their personal experience. A demonstration of the experiment can be found here: https://youtu.be/k-JYV4hyTs8.

The experiments were carried out in accordance with the guidelines and regulations of the lab and the equipment. The experimental protocols were approved by the Human Research Ethics Committee at the Delft University of Technology on 19/11/2021. All participants signed their written informed consent before participating. All collected data was anonymized before storage.

5.6.2. Hypotheses

The Learner tries right from the start to be of assistance. When in doubt of the user preference or intention, it does not act, as the punishment is less for letting the human lead than for choosing a wrong action, such that the waiting “action” has largest expected reward. However, once close to a support with little choice of actions left, it acts without needing to observe the human first. Once it has observed previous roll-outs of the task, the parameterized internal model tries to generalize the learned preferences across the possible intentions. So once a side of the rack appears to be chosen, the robot may provide assistance without before having observed the human move in that direction. However, the awareness of multiple possible intentions, with our choice of r^0, leads to there always being one state in which the robot leaves the initiative to the human, necessary to observe/predict the human's intention.

The Imitator does not do anything until the task starting from a support has been observed at least once. Then it copies what it observed the previous time. When entering a new region in the state space (coming from a specific support), it stays again passive. Without a notion of intentions, once starting the task from a support observed previously, the robot never hesitates to act. It provides maximum support if the human wants to go the same way. If not, the human has to 'fight' the robot, make it understand the desired action goes elsewhere. This will be the case in the state where the human chooses to take the turn to another support, and also in the case the human moves back to the start support. The Imitator is by design not capable of understanding the start support as goal support (Sec. 5.4.6).

Based on these differences, we expect the Learner to be overall more supportive in the sense of taking the right action at the right time and being less passive in tasks and states that were not observed before. We formulate the following hypotheses (w.r.t. the Imitator baseline):

H1. The Learner will be better able to support the human preferences and intention.

As a result, we expect:

H2. The Learner makes the task easier for the human, in terms of reducing both physical and perceived effort.

Furthermore, we test if:

H3. The user feels more comfortable when cooperating with Learner.

We test ***H1*** objectively by comparing the relative number of actions the robot initiated both correctly and wrongly. A large percentage of correct actions indicates a match of preference, while a mismatch of intention increases the number of wrong actions. Subjectively, we compare the questionnaire results on perceived understanding of preferences and intentions, learning speed, and trust.

To test ***H2*** objectively, we compare the force and torque exerted on the robot integrated over the duration of the task. For subjective evaluation, users graded how easy they felt the robot made the task, next to filling out the NASA TLX questionnaire. Additionally, the questionnaires allow us to evaluate ***H3***.

Next to these hypotheses, we will qualitatively check the convergence of the learned policy.

5.6.3. Results

Fig. 5.3 shows the percentage of 'correct' and 'wrong' abstract actions taken by the robot, lightly colored for the Imitator and darker colored for our Learner. For the plain ME-IRL agent, we use the state sequences observed with the Learner, which are most clean of the influence of wrongly initiated robot actions, and compare the actions the plain ME-IRL agent would have taken. The results are shown in gray. The uncolored space in between the bars indicates the number of state transitions in which the robot did not initiate an action.

Actions are considered 'correct' if the next recognized proximal state corresponds to the state the robot started acting towards in the previous state. This means an action is registered as correct even when in between disagreement was detected and the robot aborted its action. Considerable 'false disagreements' were detected when users found

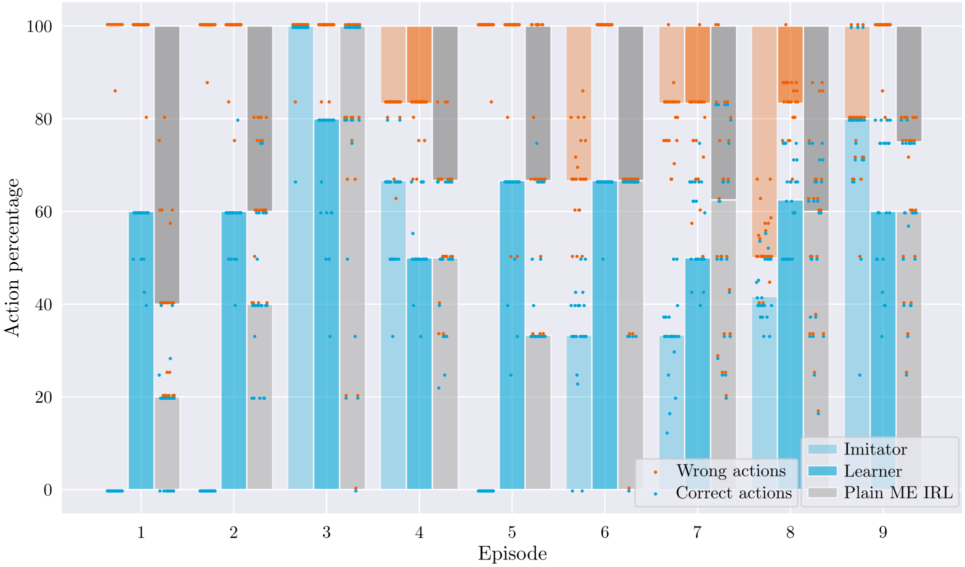


Figure 5.3: Percentage of active actions taken by the Imitator, Learner, and plain ME-IRL agent, for the nine episodes of moving the hanger as tabulated in Tab. 5.2. The correct actions (number of times when the next state recognized by the robot coincided with the state the robot decided to act towards in the previous state) are shown from the bottom up, the wrong actions (times when the next state did not match the initiated action) from above. To indicate the spread of the data, the dots represent the individual data points.

the robot too slow or pulled the robot with some force to the same next state but not via the straight line the robot tried to track. On the other hand, users occasionally disagreed close enough to the state the robot was acting towards to have the action registered as 'correct' before moving on to where they wanted to go. In those cases, the wrong action taken is registered as a 'correct' plus a passive action. Because of this effect, we expect the number of wrong actions for the plain ME-IRL agent to register slightly lower if they were recorded with the actual users.

Episode 3 is the episode in which pure imitation should give the optimal result (depending on the quality of the demonstration in Ep. 1). It is the only episode in which the intention matches the previous episode starting from the same start state. Indeed, we observe for this one (and only) episode that the Imitator outperforms the Learner. We see that the plain ME-IRL agent overfits considerably on the policy it thinks best. Like the imitator, it does very well in Ep. 3. However, as its learned model covers the full state space, already at initialization, it chooses more wrong actions than the Imitator in all other episodes. In many episodes, it also chooses more correct actions than the Imitator. In Ep. 9, the Imitator has full state information, but no recent observation of the specific intention. In Ep. 4, many users moved quite close to support B before moving over to C. This resulted in the Imitator's action going to B being registered as correct, while the Learner waited to observe the intention, and then taking one wrong action believing the user might intend to go to B. In all the other episodes, we see the Learner take at least as many, and often considerably more, correct actions. The plain ME-IRL agent is seen to

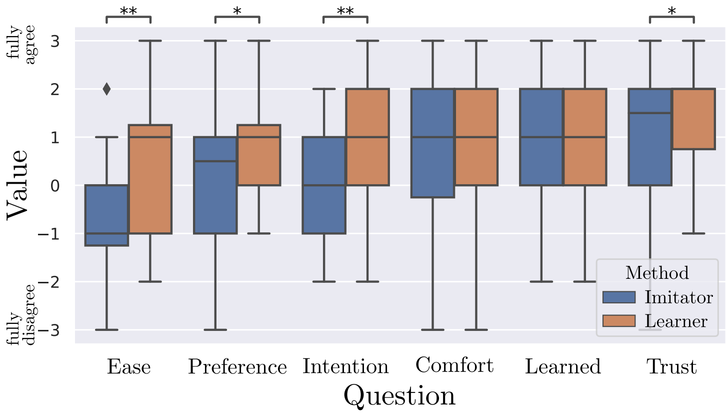


Figure 5.4: Questionnaire results to statements from left to right: The robot made it *easier* for me to perform the task. The robot understood my *preference*s, how I wanted to do the task. The robot was supporting me to go where I wanted to go (*intention*). I was *comfort*able with what the robot was doing. The robot *learned* fast. I *trust*ed the robot. Significant differences between the methods are indicated by * ($p < 0.1$) and ** ($p < 0.02$).

generalize its observations less well, as it chooses fewer correct actions in most episodes.

Over all the episodes, the Learner takes significantly more correct and fewer wrong actions compared to both the Imitator and the plain ME-IRL agent. We observe with similar significance that the plain ME-IRL agent is estimated to choose both more wrong and more correct actions than the Imitator. A two-tailed Wilcoxon signed rank test for paired samples (as the data is not normally distributed) shows the differences to be significant with $p < 10^{-6}$ for all compared action percentages. These results support ***H1***.

We see ***H1*** further supported by how the users graded different aspects of the robot performance (Fig. 5.4). The results for the two methods are compared using a two-tailed paired t-test, testing if the Learner was perceived as a significant improvement over the Imitator. The p-values are tabulated in the top half of Tab. 5.3. Significant differences are indicated by * and ** in the figure.

The most significant results are: the users 1) found the task easier to perform with the Learner compared to the Imitator, and 2) felt their intentions were better understood by the Learner. Additionally, with $0.02 < p < 0.1$, there is an indication that the users also felt their preferences were better understood, and they trusted the Learner more than the Imitator.

No difference is visible when it comes to how fast the users felt the robot was learning from their input. Interestingly, if we look at the individual results, more than half of the users felt the second method they observed learned faster. Since we alternated which method was tried first, this change in perception is canceled out in the results.

As an objective measure of effort, we consider the forces and torques integrated over the duration of the tasks. The duration is measured from the moment the robot starts grasping until the robot has let go at the intended goal state. Since the Imitator never let go between Episodes 8 and 9, these episodes are separated manually. Time in which the robot lost grasp on the hanger and was not moving is subtracted. As the trend in the resulting linear and angular impulse look very similar, we show only the linear impulse

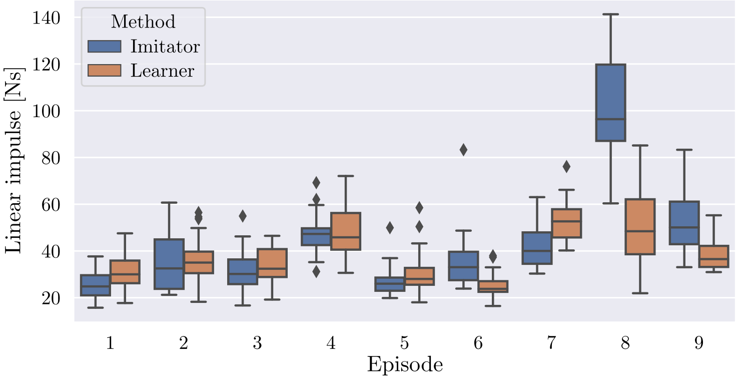


Figure 5.5: Linear impulse (interaction forces integrated over time) with the Imitator and Learner for the nine episodes of moving the hanger as tabulated in Tab. 5.2.

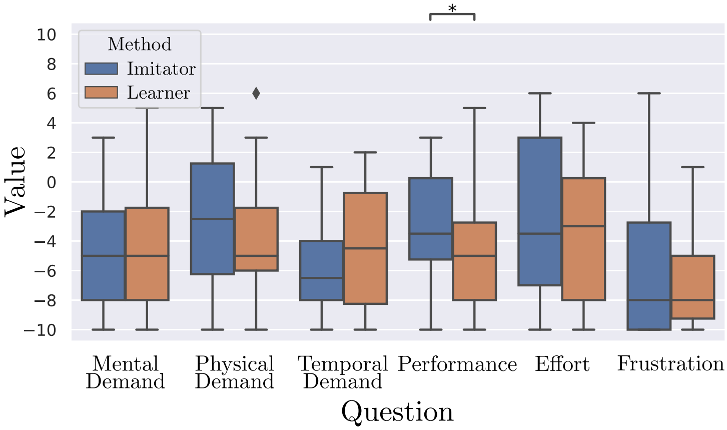


Figure 5.6: Results of the NASA TLX questionnaire. A lower score is better. Significant differences between the methods are indicated by * ($p < 0.1$).

in Fig. 5.5. The imitator performs very badly in Episode 9, where it could not understand that the human wanted to go back and had to be physically corrected until disagreement on the last action made it abort and not attempt any further actions.

In general, comparing Figs. 5.5 and 5.3, we see that the registered impulse increased when the robot was more active, regardless of the quality of the actions taken, with the exception for Episode 3. This lack of support for ***H2*** may be largely due to the preference mismatch on the action level. People generally found the straight-line trajectories unnatural, and several users seemed to prefer the robot to go faster. The presented method focuses on preferences on the level of discrete states, extending it to additionally learn preferences on how to transition between those states (Ch. 3), will likely lead to improvement on this result. In the individual results, we observe (unsurprisingly) that most users experienced lower mental demand the second time they did the task with the robot.

Table 5.3: Results of the questionnaires for the Learner and Imitator compared with a two-tailed paired t-test. The statements are phrased for the Learner w.r.t. the Imitator, as perceived by the users. P-values < 0.1 are printed bold, indicating a significant result. Values between parentheses indicate the answers to a question were not normally distributed (with $p < 0.1$) for one or both of the methods.

Statement	p-Value
Robot made task *easier*	**0.013**
Robot understood *preference*s	**0.070**
Robot supported the user *intention*	**0.013**
User was *comfort*able with robot	(0.137)
User thought robot *learned* fast	0.312
User *trust*ed robot	**0.062**
Lower *mental demand*	0.521
Lower *physical demand*	0.286
Lower *temporal demand*	(0.933)
Higher *performance*	**0.072**
Lower *effort*	(0.173)
Lower *frustration*	(0.175)

Subjectively, when explicitly questioned about the effort and demand of the task (Fig. 5.6, Tab. 5.3), the users did not grade the methods significantly different. However, they did feel they performed the task slightly better with the Learner compared to the Imitator. Furthermore, the users very significantly found the task easier to perform with the Learner compared to the Imitator (Fig. 5.4, Tab. 5.3). This does provide some weak support to ***H2***.

People did not report a significant increase in comfort with the Learner, but there is an indication that they trusted the robot more and, more significantly, that they found it easier to cooperate with. We can interpret this as a weak support to ***H3***.

To check if this trust is well placed, we have a look at the policy the robot learned. We need to look at this per preference, as the policy the robot learned is preference specific. Since our users were free to choose their preferences, we have more data on some preferences than on others. Our users chose 9 different preferences in total. To get the best impression of the variance between the users and how well the robot was able to learn, we look at the most frequently chosen preference, which is marked in Fig. 5.1a by the blue lines. Important states, in which different actions can be chosen, resulting in a different preference, are marked in circles for the intentions to go from support B to A, and from A to B.

Fig. 5.7 shows the learned action probabilities in those four critical states to those two intentions for the preference chosen by most users: From the rack to the peg (intention A): first move down (top left), then move over (top right), and finally rotate before hanging; and on the way back (intention B): first move over (bottom left), then rotate (bottom right), and finally move up before hanging the hanger on the rack. The actions shown in the figure are the actions defining the preference. The lines show the likelihood of the robot choosing the correct action compared to not taking an action (the dash-dotted line at 1.0). In blue, we show the expected relative action probability obtained from 100

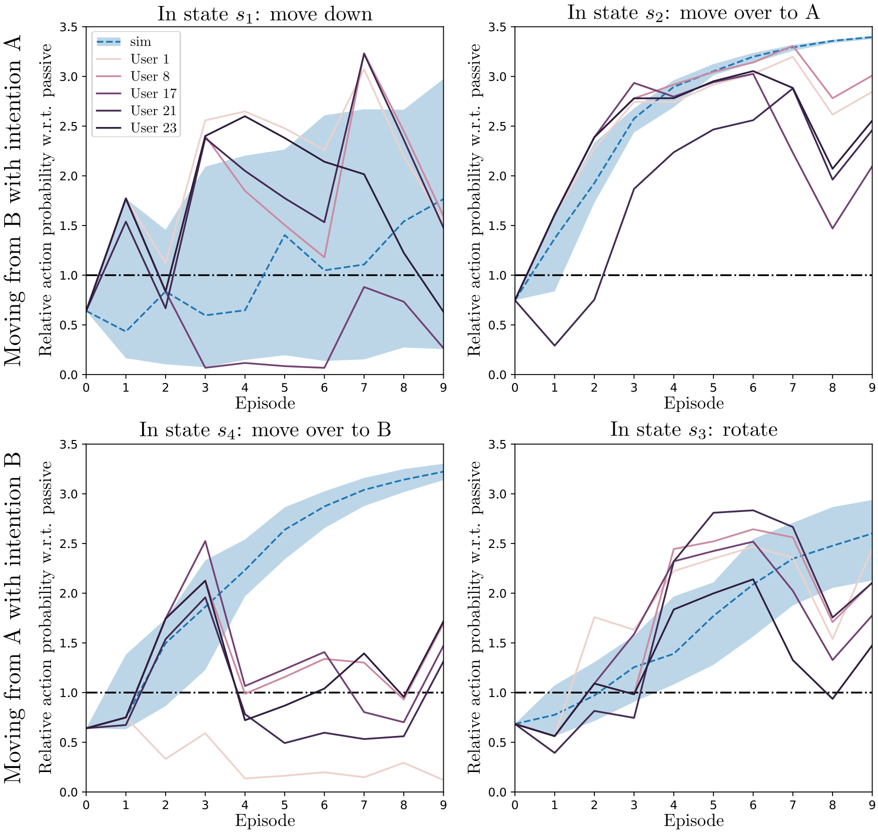


Figure 5.7: The relative probability of the preferred action being chosen for a specific intention, in the four states defining the preference that was chosen by the largest number of participants, as it was learned over the episodes. The states $s_1, \ldots, s_4$ are marked in Fig. 5.1a. The solid lines show the data from the five individual users who had this preference. The blue area is the interquartile region of a 100 simulations that were run with random start and goal supports, the dashed line shows the median.

simulations where the start and goal supports are chosen at random every episode, but the human follows the said preference perfectly.

From the simulation results, we see that our learner is able to capture some action preferences slower than others. This is due to the feature parameterization we chose. Nevertheless, in most of the critical states and with most of the users, the robot learns within a few episodes to recognize the correct action with a probability larger than the probability of staying passive.

We need to make a distinction here between the states on the left and on the right of Fig. 5.7. In states s_1 and s_4, in Fig. 5.1, we see a dashed line, an alternative path, going to intention C. The action the human will take in these states depends very much on the intention, while the states visited up to these states gave no information of the intention. In states s_1 and s_4, the Learner will not know where its partner wants to go. Not to accidentally choose a wrong action, we expect the Learner to learn to wait in these states. The small number of wrong actions shown in Fig. 5.3 indicates that this is indeed generally the case. It means that unless the Learner learns some really wrong behavior in these states, the final performance is not visibly affected by the preference model in these states. In these states we observe the largest effects of preference unlearning for certain intentions.

Specifically, we see the following four cases in Fig. 5.7 (from left to right, top to bottom): In s_1 (or the equivalent state when coming from C), the moving down action is only observed for intention A, in episodes 1, 3 and 7 (Tab. 5.2). In all other episodes, this preferred actions is slightly forgotten. This is also clearly illustrated by the wide blue band of the interquartile region of the simulated results. It suffices to move to A once in a while to keep the unlearning in check. Once moved down to the height of the goal, in s_2, moving over is quickly learned with little variance. This action is shared between all intentions as the next action to take. In s_4 when going to C (Ep. 4), it turned out to be physically very hard to follow a more or less straight line to s_5. In all of the recorded cases, the users first moved to s_3 before continuing to s_5, confusing the robot, and unlearning that the human wants to move over directly in the direction of the intended goal. Our features could not capture the preference of choosing in state s_3 whether or not to continue on to C. In s_3, learning to rotate before moving up to the final intended height was somewhat harder to learn than the moving over to A in s_2. Mostly the episodes going to A were confusing here. Nevertheless, clear learning of this preference is observed.

Additionally, there is a large dip at Episode 8. There, the human changed intention halfway to go back. This option was not included in the simulations shown in blue. At runtime, we saw that the policy the robot learned is robust to such a change of intention. However, it did confuse the model in the learning update, as our model takes the final observed intention as the baseline intention for the entire episode.

5.7. Additional Simulation Study: Wheel Scenario

In this section, we demonstrate the effect of different choices of r^0, which we choose in Sec. 5.6 to maximize the learning effect, as well as the effect on the learning performance of people acting less as deterministic agents. Because we can test with a larger state and action space in simulation, we can now also investigate how well our Learner is able to capture “inefficient” preferences, visiting more intermediate states than strictly necessary. Also, we can easily move our supports around in the simulation to demonstrate that our state space parameterization lets our agent generalize between contexts, similar to Avaei et al. (2023).

Both human and robot are simulated using the world model. The human can be controlled via a user interface, but for testing, we use pre-programmed human policies. Two human policies $\pi^{\mathrm{H}}(s,\iota)$ were provided to the simulator, characterizing the different preferences shown in Figure 5.8 (elaborating Fig. 5.1b). To these preferences, we can add a probability of the human being passive.

Of the different r^0 we tested, we present the results to the following values:

$r^0 = -0.25$ Low punishment – the robot will wait when unsure which action to take.

$r^0 = -0.50$ Medium punishment – the robot may try an action if it believes it could be better than waiting.

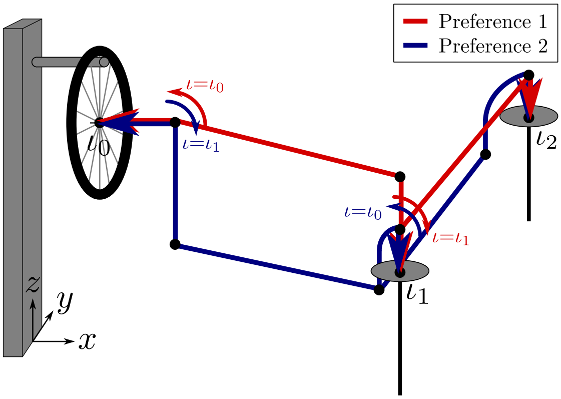


Figure 5.8: Two preferences for moving the wheel between supports: 1. take the shortest path with the least changes of direction, rotating at the last moment when necessary; 2. rotate the wheel to horizontal at the first opportunity and move over at comfortable carrying height.

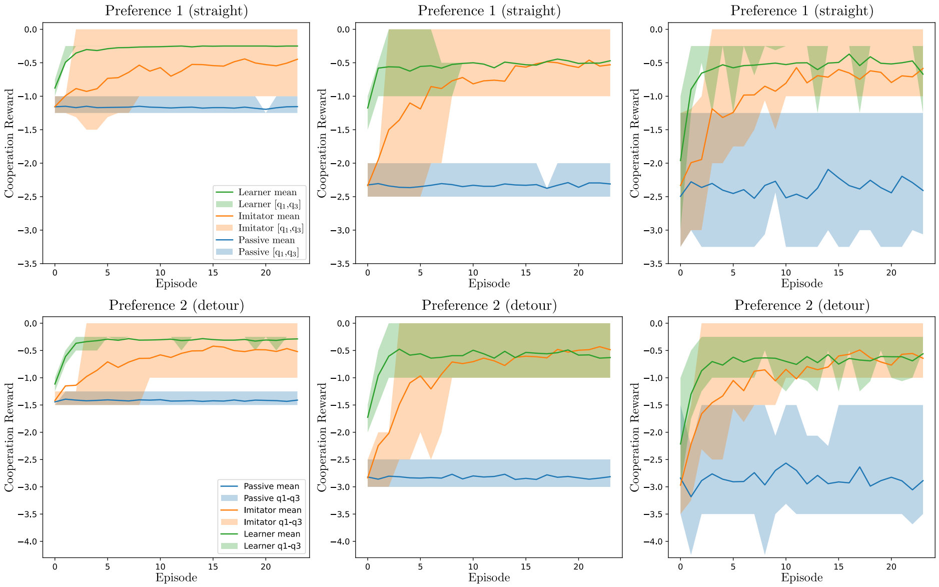
The start support and intention are chosen at random at the start of each episode. In each simulation, our Learner starts learning from an initial human model without initial preference (Sec. 5.4.5), and the Imitator starts with an empty list of actions to imitate.

Figures 5.9(a-b) show the mean and interquartile regions of the robot cooperation reward for a hundred simulations per preference, for $r^0 = -0.25$ and $r^0 = -0.5$ respectively. With a deterministic partner, we see our Learner converge within 4-8 episodes. The Imitator, after it has observed every combination of start and goal, settles down to take one wrong action with a 50% chance per episode: in the state where the human shows their intention. Here, we do not consider the possibility of going back to the start support. Since moving between supports B and C requires one action less (the wheel does not need to be rotated), the passive policy shows an interquartile range corresponding to the one passive action difference.

For low waiting punishment, the learned robot policy converges to waiting only in the state where the human shows their choice of intended goal. For the longer route (Preference 2), this may take up to six episodes, for the shorter route, three episodes already suffice. This is really fast. For medium waiting punishment, the robot is less hesitant to take an action, even if it is not very certain it is correct, as long as it could provide a higher reward. For a higher waiting punishment, the Learner converges to a policy where, in the choice state, it randomly selects a goal, performing similarly to the Imitator. The optimal value for r^0 depends on the scenario, as well as on how careful or daring the human prefers their robot partner to behave.

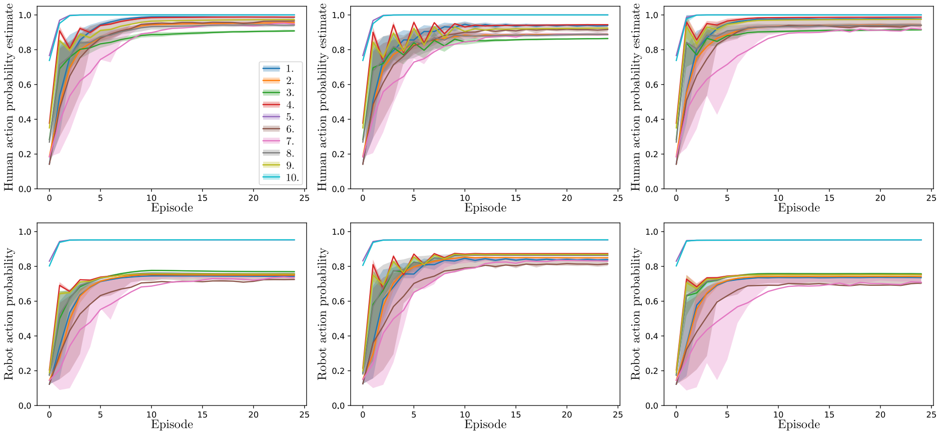
For sufficiently low waiting punishment, the Learner outperforms the Imitator. Depending on the objective (provide as much active support as possible or offer the least wrong support), the Imitator may approach the Learner's performance once it has seen which actions to imitate, if there are few enough possible partner intentions.

The Learner is also naturally able to cope well with cases where the human might start to rely on the robot once it has learned the preference to steer the large part of the



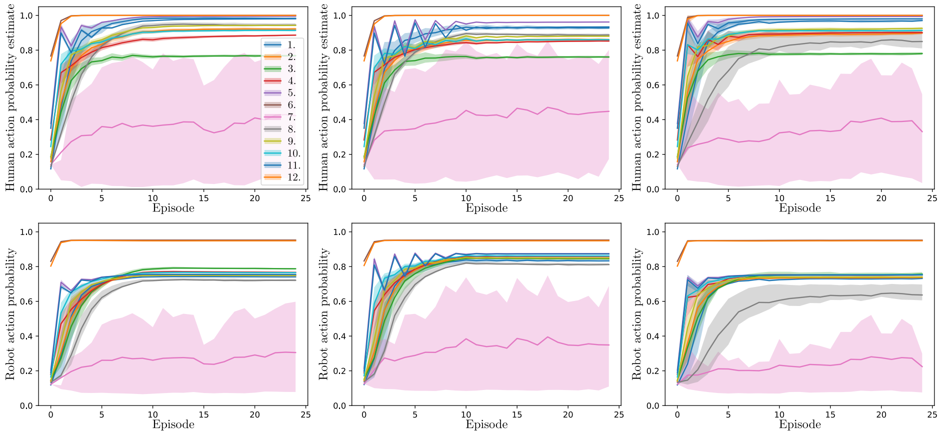
(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

Figure 5.9: Learning behavior compared, mean and interquartile regions comparing the Learner with the Imitator and Passive agent for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time.



(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

Figure 5.10: Learned human policy estimate (top) and resulting robot policy (bottom) of Preference 1 (straight) for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time. The colored lines, with interquartile bounds, correspond to the following state-action pairs, for the intention to go to rack A (Fig. 5.1: 1. s = horizontal right above a stand (B or C), a = move over to rack; 2. s = horizontal low next to rack, a = move up to final height; 3. s = horizontal high next to rack, a = rotate; 4. s = vertical high next to rack, a = put on rack; 5. s = on rack, a = let go of wheel; for the intention to go to a stand (B or C): 6. s = vertical high right next to rack, a = move over to intended stand; 7. s = vertical high above stand, a = move down to just above stand; 8. s = vertical right above stand, a = rotate; 9. s = horizontal right above stand, a = put on stand; 10. s = on stand, a = let go of wheel.



(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

Figure 5.11: Learned human policy estimate (top) and resulting robot policy (bottom) of Preference 2 (detour) for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time. The colored lines, with interquartile bounds, correspond to the following state-action pairs, for the intention to go to rack A (Fig. 5.1: 1. s = horizontal right above a stand (B or C), a = move to comfort height; 2. s = horizontal low next to stand, a = move over to rack; 3. s = horizontal low next to rack, a = move up to final height; 4. s = horizontal high next to rack, a = rotate; 5. s = vertical high next to rack, a = put on rack; 6. s = on rack, a = let go of wheel; for the intention to go to a stand (B or C): 7. s = vertical high right next to rack, a = rotate; 8. s = horizontal high right next to rack, a = move down to comfort height; 9. s = horizontal low next to rack, a = move over to intended stand; 10. s = horizontal low next to stand, a = move up to just above stand; 11. s = horizontal right above stand, a = put on stand; 12. s = on stand, a = let go of wheel.

trajectory and the human can follow passively. The Imitator could be programmed not to update its action table when the human is passive, but this would add another prior. The beauty of the proposed learning algorithm is that it does not need any prior and learns very fast nevertheless.

In Fig. 5.9, the only prior is the nominal passive policy which we used as a working baseline, but we obtained similar results without it, or when we initialize with a different preference.

Figures 5.10 and 5.11 show the human policy estimate and the robot policy in terms of action probabilities that should be dominant in the states along the preferred trajectory, to each of the cases in Fig. 5.9. We see that in every case, the human policy estimate converges to the same almost equally fast, even when the partner is partially passive.

In Fig. 5.11, we see our model has trouble capturing one specific human action. This explains why our learner struggles more to learn the presented detour case. In that state, the robot remains unsure about which corresponding action to take, resulting in an extra passive action most of the times it passes through that state.

5.8. Conclusion

This chapter presents a novel method for learning a human preference model for intention-aware cooperation from collaborative episodes. This enables our robot system to learn a personalized model of its human partner for improved collaboration.

Our main contribution is a concept for learning human preferences as an explicit function of intention, exploiting Theory of Mind to the second order. The acquired model captures preferences of how to collaboratively move objects, as well as how to infer the human's intention from the collaborative actions. We could show that our model allows the robot to take proactive actions that match both its partner's preferences and intention, with fewer mistakes than an imitation learner would make, or a plain ME-IRL learner without a human model.

A user study revealed that participants using our learning algorithm feel significantly more understood and supported in their preferences and intentions compared to an agent that just imitates their actions. Furthermore, the users felt that the task was much easier to perform with our agent, and felt it improved their performance. The fact that this was observed during only nine episodes, with seven different combinations of start position and intention, demonstrates how the generalizing capabilities of our method make our agent learn really fast.

The proposed concepts come with some limitations and assumptions. Firstly, we overestimate the knowledge of the human of the robot's policy, by giving the model access to the actual most recent robot policy. Secondly, preference learning and intention estimation were restricted to prescribed motions between a small set of predefined waypoints. Future work will focus on relaxing this assumption. Thirdly, the large set of hand-designed features used in the Inverse Reinforcement Learning limits the scalability of the method. Future work should explore and integrate learning of a minimal set of optimal intention-parameterized features, e.g., following Bobu et al. (2022). Despite these assumptions, our methods enable a robot system to learn the user's preferences as well as to estimate their intentions from only a very few interactive episodes. This allows robots to quickly learn how to provide people with personalized proactive support, improving human-robot interaction and physical cooperation.

6

Conclusion & Discussion

In this thesis, I successfully captured personalized models of the people with whom I set our robots to cooperate, whether the model concerned the user's ergonomics, or path or task preferences. In all studies, a sufficient model for the robot to work with, and make a significant difference, was captured requiring only minutes of interaction time with the specific user. All methods were tested with a physical robot that interacted physically with actual people on a physical lab setup (not just in simulation).

In Chap. 2, we measured improved ergonomics in our users, which can be seen as a factor improving comfort. In the user studies of Chap. 3 and 5, which were performed with a larger group of mostly novice users, our users generally reported they felt comfortable physically interacting with our robot. In Chap. 5, our users compared our robot to a neutral baseline during physical cooperation. On several measures indicating comfort, our robot was scored significantly higher. Therefore, I conclude I have been successful in making our two robots learn useful personalized models that capture user behavior from very little data, for the improvement of the user's comfort during cooperation with the robot.

The following two sections provide a more detailed discussion of the conclusions to the presented ergonomic optimization and preference learning. This chapter concludes with an outlook, a vision of of how in the future the topics presented separately in the different chapters may be combined into a single common framework.

6.1. Optimizing Ergonomics

In Chap. 2, we presented a novel concept for computing optimal ergonomics-enhanced plans in cooperative physical human-robot interaction tasks. In a small proof-of-concept user study, we demonstrated that our approach is capable of finding a plan which affords improved ergonomics for people working with a robot. Our predictor captures a pose model that is specific to a person. Yet, there is room for improving the model to predict either the most likely pose (from previous observations) or the most ergonomic configuration. The expected optimum is to be able to predict the most ergonomic pose the person is likely to assume. To capture an accurate model, the interdependence between poses may very likely no longer be neglected. Additionally, people may be more or less consequent and precise in their movements. To accurately capture, or learn, a personalized model of someone's postural behavior, ideally containing parameterized task dependencies, is a whole field of research in itself.

Fortunately, already with a simple ergonomics predictor such as the one presented in Chap. 2, robots can make cooperative plans with minimum, or at least bounded, ergonomic cost. We tested this by letting our users follow the plan the robot had optimized. Future research steps would be to validate the robot plans while leaving the users free to make their own decisions on the task (similar to Vianello et al. (2021), but allowing users to change grasp), and to let the robot update its internal models to improve its predictions.

In the later chapters, we worked towards the latter, although without further considering the ergonomics. From users' responses, we concluded that the assumption that ergonomics can explain people's preferences was too strong.

Nevertheless, users would still benefit when the cooperation is optimized ergonomically. However, the performance of the ergonomic optimization and the quality of the result highly depends on the ergonomics metric used to evaluate the ergonomic cost. On the one hand, current ergonomic metrics which are validated by ergonomics experts are ill-suited for optimization in continuous joint space. On the other hand, metrics that make perfect sense from an engineering perspective have not been verified by ergonomics experts. We chose to stay as close as we could to an expert-validated metric.

Very important for further progress in ergonomic optimization of dynamic tasks, supported by automation, is to have ergonomics models suitable for such optimization verified by ergonomics experts. This is future work that requires collaboration of ergonomists and engineers.

6.2. Learning Preferences

To leave people their freedom of preferences, I built Chap. 3 and 5 on the principle of inverse reinforcement learning (IRL). User studies supported that we were successful in capturing people's preferences in a short window of interaction with our robot while leaving the people the freedom to choose their own preferences.

As it turned out that existing methods still had a hard time including velocity preferences, we set out to achieve that in Chap. 3. We were successful by separating the path and the velocity preferences into two optimization steps. A comparison study showed the sensitivity of both our method and the closest related method from literature, which is also IRL-based, to the choice of features used for learning. In this thesis, we used hand-designed features, as was done in the closest related method in literature, chosen strategically to be able to generalize between the different contexts we presented to our robot and users. Ideally, we would also get rid of this restriction and design an algorithm capable of extracting relevant parameterizations and features from the observations as well, unbiased by the engineer who designed the learning framework. This was out of the scope of this thesis and left for future work.

To be able to learn preferences during physical interaction with a robot, we developed the DAVI controller presented in Chap. 4. In a small user study, we verified its capability to smoothly transition between letting the robot execute the task and letting the human demonstrate alternative, possibly previously unseen, behaviors. The two-way haptic communication between the human and the robot allows intuitive mode switching between human and robot control without taking the human attention off the physical task, the way pressing a button would do. However, not all users agreed on the intuitiveness of the haptic cues. We witnessed different users preferring different cues and (speed of) robot response. This showed the relevance of a new direction for future work: to also learn people's preferences regarding the learning process itself and the communication of it.

In Chap. 5, we were successful in making a robot learn a personalized model of its human partner for improved collaboration. We showed that our model exploiting two-level theory of mind (ToM) reasoning allows the robot to take proactive actions that match both its partner's preferences and intention. The robot's explicit awareness that its partner could have one out of several intentions let our users feel significantly more understood and supported than the baseline algorithm to which we compared, which imitated its previous observations. With the internal models we added, of the task and the human partner, our users, in general, agreed that it was easier to do the task with the robot, and that they were better at it as a team. Further improvement is to be expected when further refining the applied ToM, and when the robot additionally learns its partner's preferences on the trajectory level. This is further discussed in the vision presented in the next and final section of this thesis.

6.3. OUTLOOK

THIS thesis discussed a number of separate topics which combined have the potential to make an impact beyond the sum of the individual contributions. Here, I will build up my vision in reverse topic order.

Consider the following complex task: For moving, all household items have to be packed into a van: boxes, furniture, plants, and other odd items. The overall goal is clear: in the end, everything needs to be packed into the van. The exact intention of what should end up exactly where is not so straightforward for people to decide and not something you would want to have to tell a robot.

Next to common sense (e.g., heavy stuff on which other things can be stably stacked should go at the bottom, fragile objects on top), preferences apply when such rules are less clear (e.g., what may be packed to the back and what is rather kept in front). From that, the final intended goal configuration will become clear as the packing progresses. While packing, a robot learning these preferences can help plan ahead to arrive at a feasible goal configuration that meets the preferences (extending the research presented in Chap. 5).

When the robot has learned what may be stacked where, the person leading the packing can still have multiple intentions of what *should* go where. While helping the person carry, the robot may infer from the trajectory where the person is likely to want to go (applying the intention recognition of Chap. 5 on the trajectory level of Chap, 3). Without knowing what may be stacked where, the number of possible intentions is larger. After stacking a couple of things, patterns may be observed in the recognized intentions for learning the preference of what may be stacked where in the larger plan. This is a hierarchical application of the learning problem treated in Chap. 5.

For moving the objects, it would be of great support if the robot would learn what paths and velocities around the other objects (obstacles) are preferred, from the force feedback (pushing and pulling) it witnesses during the carrying (extending the research of Chap. 3 to the online learning during cooperation setting of Chap. 5).

Ultimately, it would be even better if the robot would additionally be aware of the ergonomics of its partner. Combining all chapters in a common framework, the robot could optimize its partner's ergonomics, for example by taking most of the weight and holding it at the optimal height (Chap. 2) within or on the bounds provided by the preferences it learned. Ideally, the posture prediction presented in Chap. 2 gets updated on the new posture data obtained on the task (preferably without requiring users to wear a sensor suit). As the ergonomics prediction gains accuracy, the robot may subtly pull its partner to more ergonomic poses. If the human adapts, the robot will update its trajectory-level preference model and may be able to continue improving the human ergonomics to the benefit of its partner without annoying its partners by suggesting sudden large changes in behavior.

Many research steps remain to be taken before this vision can become reality. In this thesis, I provided a number of ingredients that may help us get personal physical robot support to that level one day in the future.

Bibliography

Agravante, D. J., Cherubini, A., Sherikov, A., Wieber, P.-B., & Kheddar, A. (2019). Human-humanoid collaborative carrying. *IEEE Trans. Robotics*, *35*(4), 833–846.

Akgun, B., Cakmak, M., Jiang, K., & Thomaz, A. L. (2012). Keyframe-based learning from demonstration. *International Journal of Social Robotics*, *4*(4), 343–355.

AnyBody Technology A/S. (2017). www.anybodytech.com

Avaei, A., Spaa, L. v. d., Peternel, L., & Kober, J. (2023). An incremental inverse reinforcement learning approach for motion planning with separated path and velocity preferences. *Robotics*, *12*(2), 61.

Bai, H., Cai, S., Ye, N., Hsu, D., & Lee, W. S. (2015). Intention-aware online POMDP planning for autonomous driving in a crowd. *2015 IEEE International Conference on robotics and automation (ICRA)*, 454–460.

Bajcsy, A., Losey, D. P., O'Malley, M. K., & Dragan, A. D. (2017). Learning robot objectives from physical human interaction. *Conference on Robot Learning*.

Baker, C. L., & Tenenbaum, J. B. (2014). Modeling human plan recognition using bayesian theory of mind. *Plan, activity, and intent recognition: Theory and practice*, 177–204.

Belardinelli, A., Kondapally, A. R., Ruiken, D., Tanneberg, D., & Watabe, T. (2022). Intention estimation from gaze and motion features for human-robot shared-control object manipulation. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9806–9813.

Bıyık, E., Huynh, N., Kochenderfer, M., & Sadigh, D. (2020). Active preference-based Gaussian process regression for reward learning. *Robotics: Science and Systems*.

Bıyık, E., Losey, D. P., Palan, M., Landolfi, N. C., Shevchuk, G., & Sadigh, D. (2022). Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, *41*(1), 45–67.

Bobu, A., Wiggert, M., Tomlin, C., & Dragan, A. D. (2022). Inducing structure in reward learning by learning features. *The International Journal of Robotics Research*, 02783649221078031.

Boularias, A., Kober, J., & Peters, J. (2011). Relative entropy inverse reinforcement learning. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 182–189.

Buehler, M. C., & Weisswange, T. H. (2018). Online inference of human belief for cooperative robots. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 409–415.

Busch, B., Toussaint, M., & Lopes, M. (2018). Planning ergonomic sequences of actions in human-robot interaction. *IEEE International Conference on Robotics and Automation (ICRA)*.

Buşoniu, L., Babuška, R., & De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In D. Srinivasan & L. C. Jain (Eds.), *Innovations in multi-agent systems and applications - 1* (pp. 183–221). Springer Berlin Heidelberg.

Bütepage, J., Kjellström, H., & Kragic, D. (2018). Anticipating many futures: Online human motion prediction and synthesis for human-robot collaboration. *IEEE International Conference on Robotics and Automation (ICRA)*.

Calinon, S., & Lee, D. (2019). Learning control. In P. Vadakkepat & A. Goswami (Eds.), *Humanoid robotics: A reference* (pp. 1–52). Springer. https://doi.org/10.1007/978-94-007-7194-9_68-1

Choudhury, R., Swamy, G., Hadfield-Menell, D., & Dragan, A. D. (2019). On the utility of model learning in hri. *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 317–325.

da Costa, B. R., & Vieira, E. R. (2010). Risk factors for work-related musculoskeletal disorders: A systematic review of recent longitudinal studies. *American Journal Industrial Medicine*, *53*(3), 285–323.

David, G. (2005). Ergonomic methods for assessing exposure to risk factors for work-related musculoskeletal disorders. *Occupational Medicine*, *55*(3), 190–199.

DelPreto, J., & Rus, D. (2019). Sharing the load: Human-robot team lifting using muscle activity. *IEEE International Conference on Robotics and Automation (ICRA)*.

Duchaine, V., & Gosselin, C. M. (2007). General model of human-robot cooperation using a novel velocity based variable impedance control. *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'07)*, 446–451.

Ewerton, M., Maeda, G., Kollegger, G., Wiemeyer, J., & Peters, J. (2016). Incremental imitation learning of context-dependent motor skills. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 351–358.

Fahad, M., Chen, Z., & Guo, Y. (2018). Learning how pedestrians navigate: A deep inverse reinforcement learning approach. *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Ficuciello, F., Villani, L., & Siciliano, B. (2015). Variable impedance control of redundant manipulators for intuitive human–robot physical interaction. *IEEE Trans. Robotics*, *31*(4), 850–863.

Fitter, N. T., Mohan, M., Kuchenbecker, K. J., & Johnson, M. J. (2020). Exercising with baxter: Preliminary support for assistive social-physical human-robot interaction. *Journal of neuroengineering and rehabilitation*, *17*, 1–22.

Franzese, G., Celemin, C., & Kober, J. (2020). Learning interactively to resolve ambiguity in reference frame selection. *Conference on Robot Learning (CoRL)*.

Franzese, G., Mészáros, A., Peternel, L., & Kober, J. (2021). ILoSA : Interactive learning of stiffness and attractors. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Gams, A., Ijspeert, A. J., Schaal, S., & Lenarčič, J. (2009). On-line learning and modulation of periodic movements with nonlinear dynamical systems. *Autonomous Robots*, *27*(1), 3–23.

Gams, A., Petrič, T., Do, M., Nemec, B., Morimoto, J., Asfour, T., & Ude, A. (2016). Adaptation and coaching of periodic motion primitives through physical and visual interaction. *Robotics and Autonomous Systems*, *75*, 340–351.

Gaz, C., Cognetti, M., Oliva, A., Giordano, P. R., & De Luca, A. (2019). Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization. *IEEE Robotics and Automation Letters*, *4*(4), 4147–4154.

Gienger, M., Janssen, H., & Goerick, C. (2005). Task-oriented whole body motion for humanoid robots. *5th IEEE-RAS International Conference on Humanoid Robots*.

Gienger, M., Ruiken, D., Bates, T., Regaieg, M., Meißner, M., Kober, J., Seiwald, P., & Hildebrandt, A.-C. (2018). Human-robot cooperative object manipulation with contact changes. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Gribovskaya, E., Kheddar, A., & Billard, A. (2011). Motion learning and adaptive impedance for robot control during physical interaction with humans. *IEEE International Conference on Robotics and Automation (ICRA)*.

Groten, R., Feth, D., Klatzky, R. L., & Peer, A. (2012). The role of haptic feedback for the integration of intentions in shared task execution. *IEEE Transactions on Haptics*, *6*(1), 94–105.

Hadfield-Menell, D., Russell, S. J., Abbeel, P., & Dragan, A. (2016). Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 3909–3917.

Haninger, K., Hegeler, C., & Peternel, L. (2022). Model predictive control with gaussian processes for flexible multi-modal physical human robot interaction. *2022 International Conference on Robotics and Automation (ICRA)*, 6948–6955.

Hanna, A., Larsson, S., Götvall, P.-L., & Bengtsson, K. (2022). Deliberative safety for industrial intelligent human–robot collaboration: Regulatory challenges and solutions for taking the next step towards industry 4.0. *Robotics and Computer-Integrated Manufacturing*, *78*, 102386.

Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (task load index): Results of empirical and theoretical research. In *Advances in psychology* (pp. 139–183). Elsevier.

Hawkins, K. P., Bansal, S., Vo, N. N., & Bobick, A. F. (2014). Anticipating human actions for collaboration in the presence of task and sensor uncertainty. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2215–2222.

Hignett, S., & McAtamney, L. (2000). Rapid entire body assessment (REBA). *Applied Ergonomics*, *31*(201), 205.

Hogan, N. (1984). Impedance control: An approach to manipulation. *American Control Conference*.

Hoque, R., Balakrishna, A., Putterman, C., Luo, M., Brown, D. S., Seita, D., Thananjeyan, B., Novoseller, E., & Goldberg, K. (2021). LazyDAgger: Reducing context switching in interactive imitation learning. *IEEE 17th International Conference on Automation Science and Engineering (CASE)*.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., & Amodei, D. (2018). Reward learning from human preferences and demonstrations in Atari. *Advances in neural information processing systems, 31.*

Ijspeert, A. J., Nakanishi, J., & Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots. *IEEE International Conference on Robotics and Automation.*

Jain, A., Sharma, S., Joachims, T., & Saxena, A. (2015). Learning preferences for manipulation tasks from online coactive feedback. *The International Journal of Robotics Research, 34*(10), 1296–1313.

Jara-Ettinger, J. (2019). Theory of mind as inverse reinforcement learning. *Current Opinion in Behavioral Sciences, 29*, 105–110.

Jeon, H. J., Milli, S., & Dragan, A. (2020). Reward-rational (implicit) choice: A unifying formalism for reward learning. *Advances in Neural Information Processing Systems, 33*, 4415–4426.

Jin, Z.-j., Qian, H., Chen, S.-y., & Zhu, M.-l. (2011). Convergence analysis of an incremental approach to online inverse reinforcement learning. *Journal of Zhejiang University SCIENCE C, 12*(1), 17–24.

Karami, A.-B., Jeanpierre, L., & Mouaddib, A.-I. (2009). Partially observable markov decision process for managing robot collaboration with human. *2009 21st IEEE International Conference on Tools with Artificial Intelligence*, 518–521.

Katz, S. M., Maleki, A., Bıyık, E., & Kochenderfer, M. J. (2021). Preference-based learning of reward function features. *arXiv preprint arXiv:2103.02727.*

Khoramshahi, M., & Billard, A. (2020). A dynamical system approach for detection and reaction to human guidance in physical human–robot interaction. *Autonomous Robots, 44*(8), 1411–1429.

Kim, W., Lee, J., Peternel, L., Tsagarakis, N., & Ajoudani, A. (2018). Anticipatory robot assistance for the prevention of human static joint overloading in human–robot collaboration. *IEEE Robotics and Automation Lett., 3*(1), 68–75.

Kirby, R., Simmons, R., & Forlizzi, J. (2009). Companion: A constraint-optimizing method for person-acceptable navigation. *18th IEEE International Symp. on Robot and Human Interactive Communication.*

Koert, D., Pajarinen, J., Schotschneider, A., Trick, S., Rothkopf, C., & Peters, J. (2019). Learning intention aware online adaptation of movement primitives. *IEEE Robotics and Automation Letters, 4*(4), 3719–3726.

Koppula, H. S., Jain, A., & Saxena, A. (2016). Anticipatory planning for human-robot teams. *Experimental Robotics*, 453–470.

Kretzschmar, H., Spies, M., Sprunk, C., & Burgard, W. (2016). Socially compliant mobile robot navigation via inverse reinforcement learning. *The International Journal of Robotics Research, 35*(11), 1289–1307.

Kulvicius, T., Biehl, M., Aein, M. J., Tamosiunaite, M., & Wörgötter, F. (2013). Interaction learning for dynamic movement primitives used in cooperative robotic tasks. *Robotics and Autonomous Systems, 61*(12), 1450–1459.

Lai, Y., Paul, G., Cui, Y., & Matsubara, T. (2022). User intent estimation during robot learning using physical human robot interaction primitives. *Autonomous Robots, 46*(2), 421–436.

Losey, D. P., Bajcsy, A., O'Malley, M. K., & Dragan, A. D. (2022). Physical interaction as communication: Learning robot objectives online from human corrections. *The International Journal of Robotics Research*, *41*(1), 20–44.

Losey, D. P., & O'Malley, M. K. (2019). Learning the correct robot trajectory in real-time from physical human interactions. *ACM Transactions on Human-Robot Interaction (THRI)*, *9*(1), 1–19.

Maeda, G., Ewerton, M., Neumann, G., Lioutikov, R., & Peters, J. (2017). Phase estimation for fast action recognition and trajectory generation in human–robot collaboration. *International Journal of Robotics Research*, *36*(13-14), 1579–1594.

Malik, D., Palaniappan, M., Fisac, J., Hadfield-Menell, D., Russell, S., & Dragan, A. (2018). An efficient, generalized Bellman update for cooperative inverse reinforcement learning. *International Conference on Machine Learning*, 3394–3402.

Marin, A. G., Shourijeh, M. S., Galibarov, P. E., Damsgaard, M., Fritzsche, L., & Stulp, F. (2018). Optimizing contextual ergonomics models in human-robot interaction. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

MathWorks. (2018). Waypoint trajectory generator [Accessed on 10/08/2021].

Maurice, P., Padois, V., Measson, Y., & Bidaud, P. (2017). Human-oriented design of collaborative robots. *International Journal of Industrial Ergonomics*, *57*, 88–102.

McAtamney, L., & Corlett, E. N. (1993). RULA: A survey method for the investigation of work-related upper limb disorders. *Applied Ergonomics*, *24*(2), 91–99.

Mehr, N., Wang, M., Bhatt, M., & Schwager, M. (2023). Maximum-entropy multi-agent dynamic games: Forward and inverse solutions. *IEEE Transactions on Robotics*.

Mülling, K., Kober, J., Kroemer, O., & Peters, J. (2013). Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, *32*(3), 263–279.

Nemec, B., Likar, N., Gams, A., & Ude, A. (2018). Human robot cooperation with compliance adaptation along the motion trajectory. *Auton. Robots*, *42*(5), 1023–1035.

Nikolaidis, S., Hsu, D., & Srinivasa, S. (2017). Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, *36*(5-7), 618–634.

Nikolaidis, S., Nath, S., Procaccia, A. D., & Srinivasa, S. (2017). Game-theoretic modeling of human adaptation in human-robot collaboration. *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*, 323–331.

Ong, S. C., Png, S. W., Hsu, D., & Lee, W. S. (2009). Pomdps for robotic tasks with mixed observability. *Robotics: Science and systems*, *5*.

Palan, M., Shevchuk, G., Charles Landolfi, N., & Sadigh, D. (2019). Learning reward functions by integrating human demonstrations and preferences. *Robotics: Science and Systems*.

Parekh, S., Habibian, S., & Losey, D. P. (2022). Rili: Robustly influencing latent intent. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 01–08.

Park, J. S., Park, C., & Manocha, D. (2019). I-planner: Intention-aware motion planning using learning-based human motion prediction. *The International Journal of Robotics Research*, *38*(1), 23–39.

Peternel, L., Kim, W., Babič, J., & Ajoudani, A. (2017). Towards ergonomic control of human-robot co-manipulation and handover. *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*.

Peternel, L., Petrič, T., Oztop, E., & Babič, J. (2014). Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach. *Autonomous Robots*, *36*(1), 123–136.

Peternel, L., Tsagarakis, N., & Ajoudani, A. (2016). Towards multi-modal intention interfaces for human-robot co-manipulation. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Peters, L., Rubies-Royo, V., Tomlin, C. J., Ferranti, L., Alonso-Mora, J., Stachniss, C., & Fridovich-Keil, D. (2023). Online and offline learning of player objectives from partial observations in dynamic games. *The International Journal of Robotics Research*, 02783649231182453.

Pezzulo, G., Roche, L., & Saint-Bauzel, L. (2021). Haptic communication optimises joint decisions and affords implicit confidence sharing. *Scientific Reports*, *11*(1), 1–9.

Ranatunga, I., Cremer, S., Popa, D. O., & Lewis, F. L. (2015). Intent aware adaptive admittance control for physical human-robot interaction. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 5635–5640.

Ratliff, N. D., Bagnell, J. A., & Zinkevich, M. A. (2006). Maximum margin planning. *23rd International Conference on Machine Learning*.

Rhinehart, N., & Kitani, K. (2018). First-person activity forecasting from video with online inverse reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*.

Roetenberg, D., Luinge, H., & Slycke, P. (2009). *Xsens MVN: Full 6dof human motion tracking using miniature inertial sensors* (tech. rep.). Xsens Motion Technologies BV.

Roveda, L., Haghshenas, S., Caimmi, M., Pedrocchi, N., & Molinati Tosatti, L. (2019). Assisting operators in heavy industrial tasks: On the design of an optimized cooperative impedance fuzzy-controller with embedded safety rules. *Frontiers Robotics and AI*, *6*, 75.

Sadigh, D., Sastry, S., Seshia, S. A., & Dragan, A. D. (2016). Planning for autonomous cars that leverage effects on human actions. *Robotics: Science and Systems*, *2*.

Schmerling, E., Leung, K., Vollprecht, W., & Pavone, M. (2018). Multimodal probabilistic model-based planning for human-robot interaction. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 1–9.

Schwarting, W., Pierson, A., Alonso-Mora, J., Karaman, S., & Rus, D. (2019). Social behavior for autonomous vehicles. *Proceedings of the National Academy of Sciences*, *116*(50), 24972–24978.

Schweitzer, P. J., & Seidmann, A. (1985). Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, *110*(2), 568–582.

Sendhoff, B., & Wersing, H. (2020). Cooperative intelligence-a humane perspective. *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*, 1–6.

Shafti, A., Ataka, A., Lazpita, B. U., Shiva, A., Wurdemann, H. A., & Althoefer, K. (2019). Real-time robot-assisted ergonomics. *IEEE International Conference on Robotics and Automation (ICRA)*.

Shih, A., Ermon, S., & Sadigh, D. (2022). Conditional imitation learning for multi-agent games. *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 166–175.

Shivaswamy, P., & Joachims, T. (2015). Coactive learning. *Journal of Artificial Intelligence Research*, *53*, 1–40.

Siemens PLM Software. (2019). *Jack*. https://www.plm.automation.siemens.com/store/en-us/jack/

Stouraitis, T., Chatzinikolaidis, I., Gienger, M., & Vijayakumar, S. (2018). Dyadic collaborative manipulation through hybrid trajectory optimization. *2nd Conference Robot Learning*.

Stouraitis, T., Chatzinikolaidis, I., Gienger, M., & Vijayakumar, S. (2020). Online hybrid motion planning for dyadic collaborative manipulation via bilevel optimization. *IEEE Transactions on Robotics*, *36*(5), 1452–1471.

Tian, R., Tomizuka, M., Dragan, A. D., & Bajcsy, A. (2023). Towards modeling and influencing the dynamics of human learning. *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, 350–358.

Tijsma, A. D., Drugan, M. M., & Wiering, M. A. (2016). Comparing exploration strategies for Q-learning in random stochastic mazes. *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1–8.

Vasquez, D., Okal, B., & Arras, K. O. (2014). Inverse reinforcement learning algorithms and features for robot navigation in crowds: An experimental comparison. *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Verplanken, B., & Orbell, S. (2022). Attitudes, habits, and behavior change. *Annual review of psychology*, *73*, 327–352.

Vianello, L., Mouret, J.-B., Dalin, E., Aubry, A., & Ivaldi, S. (2021). Human posture prediction during physical human-robot interaction. *IEEE Robotics and Automation Letters*, *6*(3), 6046–6053.

Vijayakumar, S., & Schaal, S. (2000). Locally weighted projection regression: An $O(n)$ algorithm for incremental real time learning in high dimensional space. *17th International Conference on Machine Learning (ICML)*.

Villani, V., Pini, F., Leali, F., & Secchi, C. (2018). Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics*, *55*, 248–266.

Wang, W. Z., Shih, A., Xie, A., & Sadigh, D. (2022). Influencing towards stable multi-agent interactions. *Conference on robot learning*, 1132–1143.

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning* (Vol. 2). MIT Press Cambridge, MA.

Wirth, C., Akrour, R., Neumann, G., & Fürnkranz, J. (2017). A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, *18*(136), 1–46.

Xie, A., Losey, D., Tolsma, R., Finn, C., & Sadigh, D. (2021). Learning latent representations to influence multi-agent interaction. *Conference on robot learning*, 575–588.

Zhifei, S., & Joo, E. M. (2012). A review of inverse reinforcement learning theory and recent advances. *Evolutionary Computation (CEC), 2012 IEEE Congress on*, 1–8.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., & Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. *Aaai*, *8*, 1433–1438.

Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., & Srinivasa, S. S. (2013). CHOMP: Covariant Hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, *32*(9-10), 1164–1193.

Zyner, A., Worrall, S., & Nebot, E. (2019). Naturalistic driver intention and path prediction using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems*.

Acknowledgements

This PhD thesis would not have been without the support of a number of people. First of all, I need to thank my promotors Jens and Robert, and my external supervisor Michael for taking me on for this project and supporting me all the way through. A special thanks to Jens for not only supporting me on my PhD project, but also giving me the room to follow my great passion for music next to it. It has always been an important energy source to me, partially powering also my research. Thank you David for making me realize that in time and giving me the courage and support to ask for it.

Additionally, I must thank Michael and also Dirk for their support with the HRI DexCoop Robot soft- and hardware. Also thanks to my colleague on the project at HRI and coauthor Tamas for the support and inspiring discussions we had starting off together on our mission to improve the HRC skills of the DexCoop Robot, each in our own way.

In the lab in Delft, many thanks go to my colleague and coauthor Giovanni for teaching me how to handle the Panda robots, and for his undying support whenever I, or my student, needed help with those robots. This leads me to next thank Armin, whose MSc thesis I supervised, for the many inspiring discussions and the contribution he made to my project. Furthermore, I want to thank Luka for the co-supervision of Armin and the additional support and co-authorship to get the work published.

Wouter and Tim were my first office mates and colleagues under Jens' supervision, whom I want to thank for the warm welcome and support starting as a PhD candidate. No less do I want to thank my later office mates (in order of joining the same office): Ajith, Hongpeng, Carlos, Jihong, Giovanni, Rodrigo, Álvaro. Thank you all for the many merry, political, and philosophical discussions we had in the office, adding new perspectives to many a subject. To Carlos, I owe additional special thanks for his continued strong and reliable support as a good friend, never letting me down, in many occasions providing support before I even realized how much I could use it.

Thanks also to my other colleagues on the team, in CoR, and in HRI, for all support, on- and off-topic discussions, and the shared time.

Thanks to Andreea Bobu for the open discussion of her experiences with preference learning from human feedback and the additional reviewing of the paper to Chap. 3, and to Anna for reviewing the English language of the paper. Thanks to my external committee members Joost Broekens and Serena for their thorough review of the previous version of this manuscript.

Thanks to Joost de Winter for his quick advice to the user studies I conducted. Since all research I did I tested sooner or later with people (and of course a robot), I owe a lot of thanks to all who participated in the user studies I conducted.

Thanks to Heike for supporting and motivating me to quickly finish up my dissertation, and giving me the time for it next to the work she contracted me to do for her; and to Andy and Christina for their moral support in the lab.

Thank you my paranimphs Padmaja and Iris, both dear friends and colleague women in engineering, sharing my field, for standing with me all those years and physically at the defense of this thesis.

Thanks to many other friends: to Claudia with her experience in research and her mother Amelie, who provided me with a nice place to stay and good care whenever I visited the lab at HRI; to Olfert and to Joost Heijink for their respective listening ears and practical help when I needed it; to Anouk and Marit for their strong moral support; to Paul for additional moral and practical support; to Georgios for believing in me and showing that the impossible can be done; to Knut Roar for his motivating hard working example and being a stable factor in my final year finishing up my research and dissertation; and to Marenka, Alida, Ande, Rens, Bart, Frank, Tom, Jacco, and all others who heartened and inspired me additionally on so many occasions.

My parents, Marianne and Han, I want to thank especially for how they raised me and let me grow into the person I am to date. No matter how young I was, they took me serious as a person (without losing playfulness). They paid close attention to my interests and supported me in pursuing those that were genuine. I was aware that resources were limited, but if there was a strong motivation, there generally was a way.

When, upon entering primary school, I was hit by the gender biases of my class mates (and possibly teachers as well), I decided then and there (at the age of 5) that I would be no less than the boys around me. Yet, a young child is so easily influenced by the world around. Thankfully, my parents paid close attention and made me aware when I was copying behavior from friends and classmates, so I could choose to learn or discard, to choose my own path consciously. With the support they gave me, I was not afraid to be different.

They laid a very important basis. The search for who I am, and how to unify that person with who I want to be, continues. –It should continue always, because if I find it in a steady state to which I can converge, it means I stopped growing.– It isn't always easy, and the time is past that my parents can point me the right way. But they, together with all my friends, supervisors, colleagues, and life in general, continue to give me directions which I can choose to consider or ignore, which present me a map on which I can navigate, hopefully to continue to converge, however slowly, to the nonlinear time-variant state that is me, and that will give me the strength to face life while staying true to myself and the people and world around me.

An important lesson in life I'd like to share:

Where there is a will, there is a way. It may not be an easy one. It may not even get you where you thought you were going, but that need not be wrong. In the end it is the journey that counts. But if you don't try, you know for sure that you will never reach.

Curriculum Vitæ

Linda Fiona van der Spaa

15-04-1993 Born in Utrecht, The Netherlands.

Education

2004–2010
Grammar School
Utrechts Stedelijk Gymnasium, Utrecht (2004–2010)
Junior College, Utrecht University, Utrecht (2008–2010)

2010–2013
Bachelor of Science in Mechanical Engineering (cum Laude)
Delft University of Technology
Minor: Robotics
Thesis: Actuator design of a flapping wing MAV
Honours Prog.: AI for Robotics

2013–2017
Master of Science (cum Laude)
in Systems and Control &
in Mechanical Engineering
Delft University of Technology
Thesis: System dynamic design and control of the Plugless Robot Arm: Towards energy neutral robotics
Supervisor: Dr. ir. W.J. Wolfslag
Supervisor: Dr. ir. M. Wisse
Honours Prog.: Optimizing an optimization algorithm

2017–2022
PhD candidate
Delft University of Technology
Thesis: Learning Human Preferences for Physical Human-Robot Cooperation
Promotor: Dr.-ing. J. Kober
Promotor: Prof. dr. R. Babuška
Supervisor (ext.): Dr.-ing. M. Gienger

2022–2023	Scientific researcher Delft University of Technology	
	Project:	Control design of a new bicycle simulator
	Professor:	Prof. dr.-ing. H. Vallery

AWARDS

2017	3mE Best Graduate 2017, Delft University of Technology
2019	TU Delft Best Graduate 2017, Bataafsch Genootschap der Proefondervindelijke Wijsbegeerte

List of Publications

Journal papers

Simultaneously Learning Intentions and Preferences during Physical Human-Robot Cooperation.
Linda van der Spaa, Jens Kober, and Michael Gienger
Autonomous Robots (under review)

An Incremental Inverse Reinforcement Learning Approach for Motion Planning with Separated Path and Velocity Preferences.
Armin Avaei*, Linda van der Spaa*, Luka Peternel, and Jens Kober
MDPI Robotics, Vol. 12, No. 2, 2023

Conference papers

Predicting and Optimizing Ergonomics in Physical Human-Robot Cooperation Tasks.
Linda van der Spaa, Michael Gienger, Tamas Bates, and Jens Kober
IEEE International Conference on Robotics and Automation, 2020

Workshop papers

Disagreement-Aware Variable Impedance Control for Online Learning of Physical Human-Robot Cooperation Tasks.
Linda van der Spaa, Giovanni Franzese, Jens Kober, and Michael Gienger
IEEE International Conference on Robotics and Automation full day workshop –
Shared Autonomy in Physical Human-Robot Interaction: Adaptability and Trust, 2022

*These authors contributed equally to this work.