Legible Robot Motion Planning

Anca D. Dragan
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Siddhartha Srinivasa, CMU RI (Chair)
Jodi Forlizzi, CMU HCII
Geoff Gordon, CMU MLD
Henrik Christensen, Georgia Tech

CMU-RI-TR-15-15

Abstract

The goal of this thesis is to enable robots to produce motion that is suitable for human-robot collaboration and co-existence. Most motion in robotics is purely *functional*: industrial robots move to package parts, vacuuming robots move to suck dust, and personal robots move to clean up a dirty table. This type of motion is ideal when the robot is performing a task in isolation. Collaboration, however, does not happen in isolation. In collaboration, the robot's motion has an *observer*, watching and interpreting the motion.

In this work, we move beyond functional motion, and introduce the notion of an *observer* into motion planning, so that robots can generate motion that is mindful of how it will be interpreted by a human collaborator. We formalize *predictability* and *legibility* as properties of motion that naturally arise from the inferences that the observer makes, drawing on action interpretation theory in psychology. Predictable motion stems from a goal-to-action inference and matches the observer's expectation, given the robot's goal. Legible motion stems from an action-to-goal inference: the robot is clearly conveying its goal with its ongoing motion. We propose models for these inferences based on the principle of rational action, Bayesian inference, and the principle of maximum entropy. We then use a combination of constrained trajectory optimization and machine learning techniques to enable robots to plan motion that is predictable or legible.

Finally, we verify that the generated motions are more predictable and legible, and evaluate the impact of such motion on a physical human-robot collaboration task. Our results suggest that predictability and legibility do not only increase task performance, but also make the collaboration process more fluent, increasing subjective metrics such as trust or comfort. We also show generalizations of the legibility formalism to deception, gestures, and assistive teleoperation.

Acknowledgements

I could not ask for a better advisor than Sidd Srinivasa. Thank you, Sidd, for being a fantastic mentor, my greatest collaborator, and one of my best friends. Thanks for taking a chance on a 1st year who didn't even know what a Jacobian was, and thanks for all of your help and guidance throughout the years.

I am very grateful to the other members of my committee — Geoff Gordon, Jodi Forlizzi, and Henrik Christensen — for bringing in unique and interdisciplinary perspectives to my work. Geoff: I have benefited tremendously from your Machine Learning expertise, from my first year in grad school till now (and I am sure I'll be pinging you again in the future). Jodi: when I started tackling HRI, you put up with my roboticist ignorance, and taught me so much along the way! And Henrik: you've been an invaluable source of experience; you could tell me what will go wrong before I even tried it!

I've been so fortunate to have many other collaborators and mentors over the years, without whom this work would not have been possible: Carolyn Rose, Drew Bagnell, Andrea Thomaz, Matt Mason, Oussama Khatib, Alison Okamura, Gaurav Sukhatme, Nathan Ratliff, Matt Zucker, Stefie Tellex, Brian Ziebart, Brenna Argall, Maya Cakmak, Katharina Muelling, Henny Admoni, Kyle Strabala, Min Kyung Lee. I look forward to so much more interaction with all of you! I've also mentored students who have added their own new and unique dimensions to the research, and I'd especially like to call out Rachel Holladay, Kenton Lee, Stefanos Nikolaidis, Elizabeth Cha, and Shira Bauman. Rachel, you will always be "Minion 0"!

Working in the Personal Robotics Lab has been an amazing experience: we worked on research together, we worked on demos together, we worked on talks together, we went to the Carribean together. Thanks Mehmet, Alvaro, Dmitry, and Mike 0, for taking care of me coming in. And thanks Shervin, Mike 1&2, Jen, Liz, Kyle, Aaron W, Aaron J, Pras, Pyry, for a unparelleled work environment.

Chris, you've been a wonderful partner in crime, a constant source of support, and a very helpful critic. I feel very lucky to have a significant other who also contributes to my research, and I wouldn't trade our pillow talk on functional gradients for anything.

I'd also like to thank Michael Kohlhase and Herbert Jaeger for starting my path in Computer Science at Jacobs University Bremen, and for giving me a taste for AI. And a special callout to my undergrad friends, Mitko, Lucka, Steffi, Gina, and Oli, who have listened to my problems and lent their moral support over the course of my PhD: you guys are my second family, and our reunions make me so happy!

Going back many years, this all started with Nicole Becheanu's advice to apply to pursue my Bachelor's degree outside of Romania. Nicole, my physics teacher, had a vision for my future that I hadn't even dared dream about. When I left for undergrad, it was the second time I had ever stepped outside of my home country. It's been a heck of an adventure, and it pretty much began in Nicole's living room — thank you Nicole!

My final thanks go to my parents, Liliana and Nelu Dragan, who supported and even encouraged their only child to move all the way across the Atlantic ocean so that she can pursue her passion. I am immensely fortunate to have parents who sacrifice their own happiness for my own, and I'll try my hardest to make it worth their while!

Contents

1 Introduction 9

2 Related Work 15

2.1 Autonomously Generating Motion around Humans 15

2.2 Non-Autonomous Motion around Humans 16

2.3 Human Inferences 17

3 Formalizing Motion Planning with Observer Inferences 21

3.1 Formalizing Predictability and Legibility 21

3.2 Modeling Predictable Motion via Optimization 24

3.3 Modeling Legible Motion via Optimization 25

3.4 From Theory to Real Users 28

3.5 Chapter Summary 34

4 Trajectory Optimization 35

4.1 Functional Gradient Trajectory Optimization 35

4.2 Optimizing with Constraints 44

4.3 Learning from Experience 52

4.4 Chapter Summary 64

5 Generating Predictable Motion 65

5.1 The Predictability Gradient 65

5.2 *Learning from Demonstration* 66
5.3 *Familiarization to Robot Motion* 81
5.4 *Chapter Summary* 95

6 *Generating Legible Motion* 97
6.1 *The Legibility Gradient* 97
6.2 *Trust Region Constraint* 101
6.3 *From Theory to Users* 103
6.4 *Chapter Summary* 107

7 *User Study on Physical Collaboration* 109
7.1 *Motions* 109
7.2 *Hypotheses* 111
7.3 *Experimental Design* 111
7.4 *Analysis* 116
7.5 *Chapter Summary* 122

8 *Generalizations of Legibility* 123
8.1 *Viewpoint, Occlusion, Other DOFs* 123
8.2 *Deception* 124
8.3 *Pointing Gestures* 132
8.4 *Assistive Teleoperation* 138
8.5 *Relation to Language* 146

9 *Final Words* 149

10 *Bibliography* 165

1

Introduction

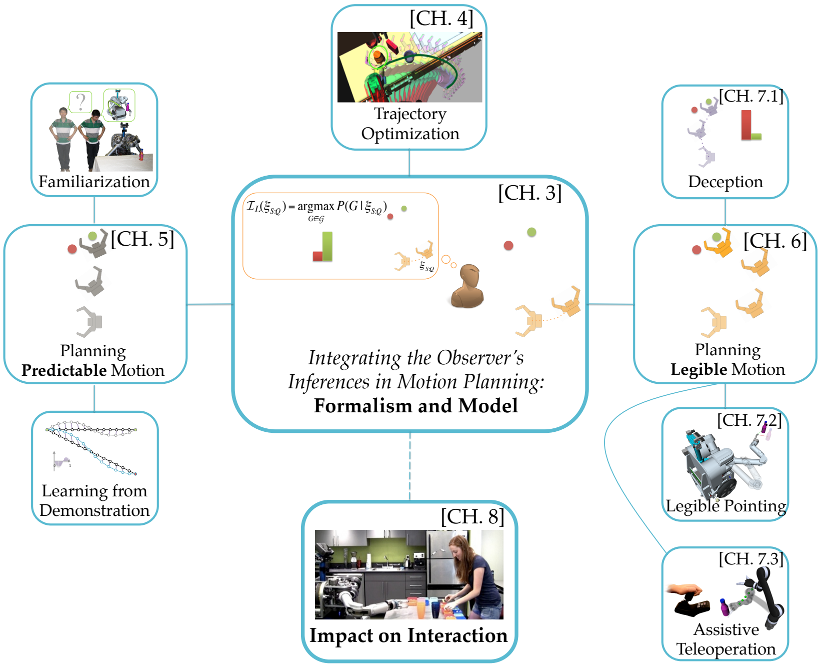

[CH. 4]
Trajectory Optimization
Familiarization
[CH. 5]
Planning
Predictable Motion
Learning from Demonstration
[CH. 3]
$\mathcal{I}_L(\xi_{SQ}) = \arg\max_{G\in\mathcal{G}} P(G|\xi_{SQ})$
ξ_{SQ}
Integrating the Observer's Inferences in Motion Planning:
Formalism and Model
[CH. 7.1]
Deception
[CH. 6]
Planning
Legible Motion
[CH. 7.2]
Legible Pointing
[CH. 7.3]
Assistive Teleoperation
[CH. 8]
Impact on Interaction

Figure 1.1: Thesis overview. We introduce a formalism for robot motion planning with a human observer. We formalize predictability and legibility as properties of motion that enable the observer's goal-to-action and action-to-goal inferences: we first introduce mathematical measures for these properties that are tractable to evaluate, and then use a combination of trajectory optimization and learning techniques to autonomously generate predictable and legible motion. We also show generalizations to deception, pointing gestures, and assistive teleoperation. Finally, we evaluate the impact of this motion in physical interactions.

Collaboration is a delicate dance of prediction and action, where the two agents predict each other's intent, as well as act to make their own intentions clear. When we collaborate, we rely on being able to anticipate our collaborator's next actions, and not be surprised by what comes next. When we clean up the dining room table with someone, as they reach for the empty bottle of water, we anticipate their goal and start reaching for the plate sitting next to it instead. We communicate relentlessly and via numerous channels.

The goal of this thesis is to enable robots to take part in the communication that needs to occur during collaboration, in order to efficiently and fluently collaborate with humans.

We envision personal robots clearing a table with someone in their home, manufacturing robots welding a part together with a human co-worker, or rehabilitation robots assisting spinal cord injury patients with their activities of daily living.

Among the various channels of communication, we focus on *motion* — a channel that naturally arises in physical tasks, and is sometimes the only channel available to a robot, e.g., an industrial manipulator. While communication through motion is natural in animation, dance, or theater, it is understudied in robotics. The key reason for this is that except for specialized motion, like gesture, most motion in robotics is purely *functional*: industrial robots move to package parts, vacuuming robots move to suck dust, and personal robots move to clean up a dirty table.

functional motion solves the piano mover's problem: achieve the goal, avoid collisions

Collaboration, however, demands moving beyond solely functional motion. Functional motion is ideal when robots perform tasks in isolation. But, motion in human-robot collaboration is never performed in isolation. In collaboration, the motion has an *observer*. The robot's motion must communicate to the collaborator, who is observing and interpreting the motion. Thus, understanding and generating motion for human-robot collaboration must consider additional constraints beyond those for functionally completing the task. This is our central tenet.

This thesis integrates the idea of an observer into motion planning, enabling the robot to reason about *how its motion will be interpreted by its observer*. We formalize two properties of motion, *predictability* and *legibility*, based on two complementary inferences that a human observer makes when observing motion (Chapter 3). Predictable motion matches expectation — it matches the observer's inference of the motion from a known intent (a "goal-to-action" inference). Legible motion communicates its intent — it enables the observer's inference of the correct intent from the ongoing motion (an "action-to-goal" inference). To then generate such motion, we propose a cost-based Bayesian *model* for these two inferences, building on tools from machine learning and trajectory optimization.

predictable motion enables the "goal-to-action" inference: it matches expectations

legible motion enables the "action-to-goal" inference: it conveys intent

Predictability and legibility, although often confused in the literature, are fundamentally different: they stem from inferences in *opposing* directions. They are contradictory in ambiguous situations, when the urgency to communicate intent — *to be legible* — is even greater. As a result, planners cannot target predictability and assume that intent will be conveyed as a result: *in situations where conveying intent is important, planners must explicitly reason about the legibility of the motion.*

With this work, we enable robots to plan legible motion, and we test its importance in studies with novice users during physical collaborations. Our results suggest that the interaction does not only become objectively better (e.g., the human-robot team is more efficient), but also subjectively better (e.g., users strongly prefer working with a legible robot, they trust it more, they think it is more capable, etc.)

Finally, we also show the generalization of our formalism beyond legible goal-directed motion, to producing deceptive motion, to generating legible pointing gestures, and to inferring human intent from ongoing action.

Contributions. This thesis makes the following contributions:

Formalizing Observer-Interpretable Motion: We introduce a formalism for motion planning in terms of the inferences made by the motion's observer, leading to two important properties of motion: predictability and legibility.

We propose models for the observer's inferences based on the principle of rational action in the theory of action interpretation, the principle of maximum entropy, and Bayesian inference, leading to quantifiable measures of predictability and legibility. Finally, we propose an approximation to make their evaluation tractable for robots with many degrees of freedom [1], and test the predictions that these metrics make about the motion in a user study (Chapter 3).

Improving Trajectory Optimization: Generating predictable or legible motion relies on optimizing the measures that our formalism introduces. To do so, we build on functional gradient optimization (Section 4.1). We alleviate the challenge of optimizing non-convex cost functions in high-dimensional spaces by capitalizing on the structure found in day-to-day manipulation tasks.

First, manipulation tasks are described by a set of goal configurations, as opposed to a single configuration like in classic motion planning. We enable optimizers to take advantage of goal sets by formalizing goal sets as an instance of trajectory-wide constraints, and deriving an algorithm for optimization with such hard constraints [2] (Section 4.2).

Second, robots perform similar manipulation tasks over and over again, creating a library of previous experiences. We develop an algorithm that learns from these experiences to predict, in a new situation, what a good trajectory initialization would be — i.e., a trajectory that lies in a good basin of attraction. In doing so, we take advantage of the fact that only a few attributes of the trajectory are enough to describe a good basin [3] (Section 4.3).

[1] A.D. Dragan, K.T. Lee, and S.S. Srinivasa. Legibility and predictability of robot motion. In *International Conference on Human-Robot Interaction (HRI)*, 2013

[2] A.D. Dragan, N. Ratliff, and S.S. Srinivasa. Manipulation planning with goal sets using constrained trajectory optimization. In *ICRA*, May 2011

[3] A.D. Dragan, G. Gordon, and S. Srinivasa. Learning from experience in manipulation planning: Setting the right goals. In *ISRR*, 2011

Learning Predictable Motion from Demonstration: Predictable motion matches the observer's expectation. Different observers, however, can have different expectations. Thus, to improve predictability, we rely on demonstrations from the observer, and the ability to generalize them to new situations.

In low-dimensional spaces, the robot can directly learn a cost function to optimize from these demonstrations, via Inverse Optimal Control. In high-dimensional spaces, where this is intractable, we adapt demonstrations locally. To do so, we formalize the adaptation problem as a Hilbert norm minimization, turning it into a trajectory optimization problem [4] (Section 5.2).

We also investigate whether we can invert the teacher-learner relationship: can the robot become the teacher, and train the human observer's expectations? We call this process familiarization [5] (Section 5.3)).

Planning Legible Motion: To move from predictability to legibility, we first derive the functional gradient for the legibility metric. We then introduce a constrained trajectory optimization algorithm to generate motion that is legible, and test its performance in a user study [6] (Chapter 6).

One intuitive result is that the robot starts *exaggerating* its motion to the left of to the right when reaching for an object, in order to better convey whether its goals is the one on the left or the one on the right. Exaggeration is one of the twelve Disney principles of animation, and it is not surprising that it could be useful in expressing intent. However, nowhere did we have to handcode exaggeration as a strategy. The robot figured out that it should exaggerate, and it figured out how to do it:

> *Exaggeration naturally emerged out of the mathematics of legible motion.*

Evaluating Impact on Human-Robot Collaboration: Although our studies test that the robot can indeed generate more legible and more predictable motions, it is crucial to also test the impact of generating such motion on physical collaborations. We do so in a final wrap-up study that brings users in a shared workspace collaboration with the robot, and evaluates the success of the collaboration both objectively and subjectively.

Our results suggest that legible motion leads to more effective and fluent collaborations than predictable motion, which is in turn better than functional motion. However, the difference between legibility and predictability is more subtle (smaller effect) compared to that between predictability and functionality [7] (Chapter 7).

Showing Generalization: Even though we originally developed the

[4] A.D. Dragan, K. Muelling, J.A. Bagnell, and S.S. Srinivasa. Movement primitives via optimization. In *International Conference on Robotics and Automation (ICRA)*, 2015

[5] A.D. Dragan and S.S. Srinivasa. Familiarization to robot motion. In *International Conference on Human-Robot Interaction (HRI)*, 2014

[6] A.D. Dragan and S.S. Srinivasa. Generating legible motion. In *Robotics: Science and Systems (R:SS)*, Berlin, Australia, June 2013

[7] A.D. Dragan, S. Bauman, J. Forlizzi, and S.S. Srinivasa. Effects of robot motion on human-robot collaboration. In *International Conference on Human-Robot Interaction (HRI)*, 2015

legibility formalism for generating goal-directed legible motion, it has been applied across a variety of domains (Chapter 8).

The formalism directly generalizes to changes in observer viewpoint, occluded regions, and using different degrees of freedom on the robot to produce different effects. For instance, the robot will open its hand more than needed as its reaching in order to convey that it is about to grasp the larger object, and close its hand more than needed in order to convey that it is about to grasp the smaller object.

The most direct extension was to go beyond conveying intent, to deceiving: if we can maximize the probability of the user inferring the correct goal, we can also minimize it, or purposefully target ambiguity [8] (Section 8.2).

We also introduced an algorithm for generating legible deictic gestures. When a robot points at an object in a real-world scene, it is not always immediately clear to an observer what it intends to be pointing at. The goal was to address the challenge of finding a final pointing configuration that clearly conveys to a human observer what object the robot is pointing at [9] (Section 8.3).

Moving beyond conveying intent, we used the same model of how humans infer intent to enable the robot to make predictions about the human. We applied this in an assistive teleoperation setting, where the robot predicts the operator's intent from their ongoing inputs, and starts assisting to achieve it by arbitrating between the direct input and the predicted policy. We then studied the arbitration function from both a stability viewpoint, as well as a user-centric viewpoint. Our results suggests that assistance is useful, but the arbitration function should be mediated by the robot's confidence in its prediction, the task difficulty, and the user's personal preferences [10] (Section 8.4).

Finally, we show how the same underlying formalism can be applied to language to produce unambiguous sentences, that the the listener's language grounding process into account and increase the probability that the listener will make the right grounding. We do so by showing the connection between our work and that of Tellex et al. [212].

OVERALL, this thesis takes a first step towards motion planning informed by the inferences that human observers make. It enables tractable motion planning over the human's belief of the robot's goal, resulting in *legible* motions, and shows generalizations of legibility beyond goal-directed motion. Our prediction and hope is that as robots become more and more capable and need to work with and around humans, the need for such algorithms that generate behavior mindful of the human will become more and more prevalent.

---

[8] A.D. Dragan, R. Holladay, and S.S. Srinivasa. An analysis of deceptive robot motion. In *Robotics: Science and Systems (R:SS)*, 2014

[9] R. Holladay, A.D. Dragan, and S.S. Srinivasa. Legible robot pointing. In *International Symposium on Human and Robot Communication (Ro-Man)*, 2014

[10] A.D. Dragan and S.S. Srinivasa. Formalizing assistive teleoperation. In *Robotics: Science and Systems (R:SS)*, Sydney, Australia, July 2012

2

Related Work

Each chapter below touches upon relevant prior work. Here we focus on the context of planning motion with human observers.

We build upon a long history of robots operating in human environments — integrated systems that combine navigation, perception, motion planning, and learning in real-world environments. These include wheeled mobile manipulators [168, 11, 149, 150, 215, 120, 7] and humanoid robots [189, 169, 112, 116, 5, 117].

The main experimental platform in this thesis is our personal robot HERB, a bi-manual mobile manipulator whose pictures are sprinkled throughout the remaining chapters, which joins an active list of personal robots [38, 10, 19, 107, 181].

These robots address several challenges of human environments including navigation in clutter [126], building world models [96], and discovering, recognizing and registering objects [40, 42, 158, 43, 41]. A crucial challenge for accomplishing tasks in these unstructured environments in motion planning in high-dimensional manipulation configuration spaces.

2.1 Autonomously Generating Motion around Humans

Autonomously generating motion that avoids collisions with the environment means solving the *motion planning problem*.

Much of robotic motion planning has focused on functional motion, with sampling-based planners being widely used in high-dimensional spaces [18, 119, 101, 141, 134, 100, 35, 123, 118].

Even producing functional motion is complicated by numerous constraints imposed on the robot's motion, including torque limits, collisions, and most often the pose of the end-effector [205, 130, 232, 233, 68, 25, 231, 44, 199].

However, recent progress in trajectory optimization has made it possible to not just produce feasible motion, but to produce motion that optimizes cost. Optimizing cost is a crucial step towards producing motion mindful of observer inferences.

Among the several trajectory optimization techniques [159, 219, 103, 114, 216], we propose to use CHOMP [184, 185], an algorithm for real-world manipulation problems. CHOMP uses functional gradients [228, 178, 237, 182, 32], which are efficient and effective (with planning times of $20 - 100$ms) and inherit sound properties (e.g., invariance to reparametrization) from variational calculus.

Our main contribution is agnostic to the optimizer, but we do make certain improvement to trajectory optimization for manipulation tasks.

Autonomous motion around humans typically deals with safety. The first step in motion planning when humans are present is to *avoid injuring the human*. Humans move, which means the planner needs to handle dynamic obstacles and be able to replan [221, 175, 72] and adjust the timing of its path [137, 91]. Some techniques *anticipate* the future human motion in order to preemptively plan a successful avoidance path [239, 152].

The human's physical comfort is the next step towards motion planning around humans, that certain planners have begun to address. Planners can ensure the robot is visible [200], or that when it hands an object to the human, the require human configuration is comfortable [153].

In contrast, this thesis tackles human internal state, by introducing motion planners that reason about the inferences that the human needs to make for seamless interaction and collaboration. With prior work tackling physical human state, the time is ripe for motion planning to start addressing the human beyond safety and physical comfort.

In motion planning with human inferences, we instantiate belief space planning [180] because we plan over the human's belief. However, because we specifically look at goal inferences, we can write the state explicitly as a function of the robot's current state (which we assume to be observable): a Markov world separates the goal from past states, conditioned on the current state.

2.2 *Non-Autonomous Motion around Humans*

Motion that does match human expectation, or that communicates, is typically not autonomously planned in a way that generalizes across any environment. However, techniques do exist that require expert input for designing such motion.

HUMAN-LIKE MOTION: Predictable motion is expected, and related to human-like motion. Several animation techniques have been developed to produce natural motion, including keyframing [140], retargetting motion-capture trajectories from a professional actor onto a new character [85, 111, 142]. Animation also uses trajectory optimization [229, 188, 36, 147], in some cases generating natural motion autonomously.

Biomechanics studies have explored spatial and temporal coordination [73, 138] in human motion. Gielniak and Thomaz [82] have developed a metric for human-like motion (spatio-temproal correspondence) that is optimized to generate human-like variations of given motion. Another algorithm generates variations in motion that stay true to the intent of the original [80].

COMMUNICATIVE MOTION: Legible motion communicates intent (enables intention inference [146], "readable" [210], or "understandable" [9]), and is often cited in conjunction to or as an effect of predictability [110, 20, 8, 62, 125]. The robotics literature has developed algorithms inspired by Disney animation principles, that algorithmically change a given motion to add communicative enhancements like "secondary motion", "anticipation", and "exaggeration." [79, 84, 81].

2.3 *Human Inferences*

This thesis looks at the two complementary inferences humans make relating an agent's actions and its goals. This is a subject of study in *action interpretation theory* in psychology.

MOTIVATION IN COLLABORATION THEORY. Collaboration (also referred to as joint action or shared cooperative activity) is a social interaction in which the interactants are mutually responsive to one another, there is a shared goal, and the participants coordinate their plans of action and intentions [30]. The ability to express intent is argued to be a crucial aspect of the collaboration process [218].

In particular, exaggerating motion to better convey intent (an example arising out of legibility optimization in this thesis) is especially acknowledged as enhancing collaboration, and is considered a "coordination smoother", helping the process of prediction and monitoring of the collaborator's activity [225]. Motion or action interpretation has been heavily studied in experiments with infants, establishing the perception of intentionality [17, 34, 21, 160, 77] and the principle of rational action [78]. We build on these theories in our formalism and models for motion (Chapter 3).

Action-to-goal and goal-to-action. Humans have a universal tendency to interpret the behaviors of others as *intentional*, *goal-directed* actions. Very young infants segment complex actions into units corresponding to the initiation and completion of intentional action [17]. They show surprise when confronted with actions that are inefficient in achieving their goals [78], and are more likely to imitate actions that they perceive as intentional than those they perceive as accidental [34, 21]. Older infants have been shown to imitate the demonstrator's actual goals rather than the exact demonstrations [160, 77].

Interpreting the behavior of others as goal directed enables an observer to make sense of this behavior, and it plays a crucial role in collaboration [218]. In one theoretical account, Csibra and Gergely [47] propose *two inferences* fundamental to action interpretation. An *"action-to-goal"* inference is based on understanding the function of an action, and refers to the observer's ability to infer someone's goal state from his ongoing actions (e.g., because he is pouring coffee beans into the grinder, he will eventually hold a cup of coffee). A *"goal-to-action"* inference refers to an observer's ability to predict the actions that someone will take based on his goal (e.g., because he wants to make coffee, he will pour coffee beans into the grinder). As Section 3.1 will reveal, these two inferences in *opposing* directions are fundamental to legibility and predictability.

"action-to-goal": what is the goal of the agent, given its ongoing action?

"goal-to-action": what action will the agent take, given its goal?

One key cognitive mechanism through which both of these inferences may take place is *teleological reasoning*, rooted in the principle of rational action [78, 46, 203, 77] — humans expect others to act rationally and take the actions that are most justifiable [46] or efficient [47] given a particular situation and a particular goal. Therefore, if the goal is known, they can infer the action by asking which action would be most efficient in achieving it. Furthermore, while observing an action, they can infer its likely goal by considering "what end state would be efficiently brought about by the action" [47]. There is ample evidence that even very young infants take efficiency into account when imitating and predicting actions [78, 77]. Teleological reasoning motivates our cost-based models in Section 3.2 and Section 3.3, because (a) it leads to theory well-supported in robotics and machine learning, and (b) it has been shown to extend beyond observing humans [78], including to observing robots [115].

teleological reasoning is a mechanism for these inferences rooted in the expectation of efficient behavior

Bayesian models of intent inference. We are of course not the first to point out the Bayesian relation between the two inferences:

action-to-goal and goal-to-action. A Bayesian approach for intent inference has been introduced in plan recognition [37], cognitive science [16], psychology [176], natural language understanding [90], and perception of human action [239].

One challenge that we faced for inferring goals from ongoing *motion* is that this inference happens in *continuous time* and from ongoing trajectories though a *high-dimensional* configuration space. We introduce the general formulation, along with approximations that make the computation tractable.

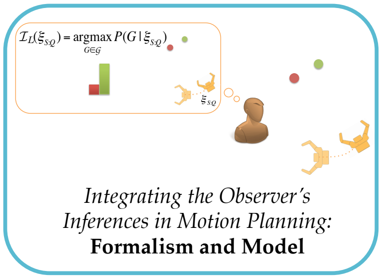
A KEY INSIGHT that legible motion brings about is the difference between inferring intent and conveying intent. Previous work focused on action-to-goal and goal-to-action inferences in a Bayesian setting, where the space to search (goals and actions respectively) and the space over which the probability distributions normalize (goals and actions respectively) match: when inferring goals, we search over goals, and normalize the probability distribution over goals.

It is when we move to conveying intent that the two space no longer match: we normalize over goals (take the candidate goals that the observer might infer into account), but search over actions (trajectories). As a result, actions that are probable given a goal are not the best at conveying that same goal. Legible motion will depart from predictability in order to better convey intent.

3

Formalizing Motion Planning with Observer Inferences

We begin with our formalism for motion interpretable by an observer. We define *functional*, *consistent*, *predictable*, and *legible* motion, with the last two intimately related to the existence of an observer, and stemming from the two (symmetric) inferences the observer makes (Section 3.1). We propose then quantifiable metrics for predictability and legibility based on mathematical models of these inferences, with roots in the principle of maximum entropy and Bayesian inference (Sections 3.2 and 3.3).


$\mathcal{I}_L(\xi_{SQ}) = \underset{G\in\mathcal{G}}{\arg\max}\, P(G|\xi_{SQ})$
ξ_{SQ}
Integrating the Observer's Inferences in Motion Planning:
Formalism and Model

3.1 Formalizing Predictability and Legibility

We focus on goal-directed motion. Here, an actor is given a motion planning problem $P \in \mathcal{P}$ and executes a trajectory $\xi \in \Xi$, with Ξ the Hilbert space of trajectories, towards one goal $G \in \mathcal{G}$ from a set of possible goals. It is perhaps most intuitive to think about legible end effector trajectories, but we broadly define trajectories to mean the full configuration space (full body motion), including even mobile robot trajectories [87]. We use the example in Fig. 3.1, where a robot is extending its hand reaching for the green object to formalize a *taxonomy of motion*. We formalize *functionality*, *consistency*, *predictability* and *legibility*, with the last two intimately dependent on an *observer*.

$\xi : [0,1] \rightarrow \mathcal{Q}$ is a trajectory

$G \in \mathcal{G}$ is a candidate goal

Definition 3.1.1 *Functional motion is that which achieves the goal.*

We formalize *functional* motion as that for which the trajectory ξ satisfies conditions of feasibility, for example, starts at the starting configuration S, achieves (ends at) goal G, and avoids obstacles:

$$\xi \in \Xi^{\mathrm{f}} \subseteq \Xi \tag{3.1}$$

where Ξ^{f} is the subspace of feasible trajectories.

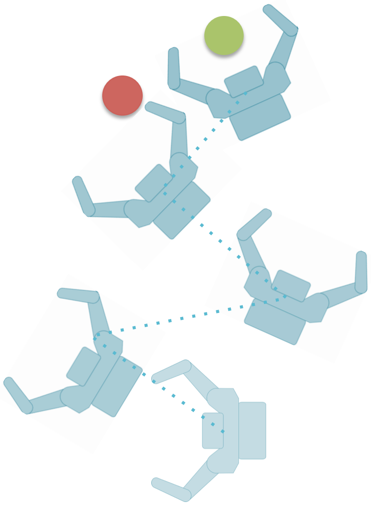


Figure 3.1: Functional motion.

Much of robotic motion planning has focused on functional motion, with sampling-based planners being widely used in high-dimensional spaces [18, 119, 101, 141, 134, 100, 35, 123, 118]. These planners use random sampling to produce plans quickly. But, randomization produces inconsistency, resulting in a different trajectory, like the one from Fig. 3.1, every run. Deterministic sampling [92, 93] produces consistency *within* a problem but not *across* problems. If either goal were ever so slightly moved, the resulting trajectory could be significantly different.

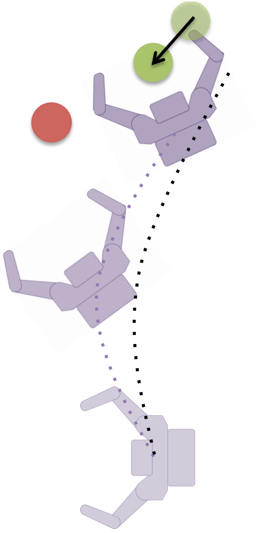


Figure 3.2: Consistent motion.

Definition 3.1.2 *Consistent motion is that where similar problems have similar trajectories.*

We formalize consistent motion as that which is consistent across problems $P \in \mathcal{P}$:

$$P_1 \text{ close to } P_2 \implies \xi_1 \text{ close to } \xi_2 \tag{3.2}$$

This defines a notion of continuity of trajectories across problems. When P and Ξ are endowed with measures of distance, we can formalize this using epsilon-delta closeness as:

for all $\epsilon > 0$ there is $\delta > 0$ such that, whenever $d_{\mathcal{P}}(P_1, P_2) < \epsilon$, then $d_{\Xi}(\xi_1, \xi_2) < \delta$

In contrast, repeatability just requires consistency *within* the same problem.

Trajectory optimization produces consistency by optimizing cost [159, 219, 103, 114, 216, 184]. The trajectory from Fig. 3.2 is consistent. If either object were slightly moved, the trajectory would change only slightly, as shown. It is not, however, predictable or legible, because the optimizer does not reason about the existence of an *observer* watching, expecting or making inferences on the motion. What does the presence of an observer imply for robot motion?

Definition 3.1.3 *Predictable motion is that which matches what an* ***observer*** *would expect, given the goal.*

Imagine someone *observing* the robot, knowing that the hand will reach towards the green goal. Even before any motion, the observer creates an expectation of what trajectory they envision the robot will take. We denote this inference function mapping goals to trajectories as:

$$\mathcal{I}_P : \mathcal{G} \rightarrow \Xi^f \tag{3.3}$$

When motion is *predictable*, the trajectory $\xi_{S \rightarrow G}$ closely matches this inference:

$$\xi_{S \rightarrow G} = \mathcal{I}_P(G) \tag{3.4}$$

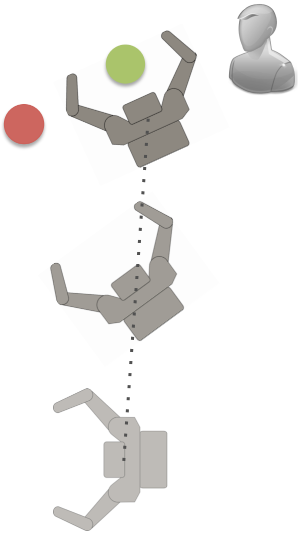


Figure 3.3: Predictable motion.

The trajectory from Fig. 3.3 is predictable to our observer, as it matches what they expected. However, it is also ambiguous: another observer would not be able to tell which object the robot wants to grasp until the very end. Thus, predictable motion is not necessarily legible.

Can consistent motion become predictable with familiarity? We explore this hypothesis in Section 5.3.

Definition 3.1.4 *Legible motion is that which enables an* ***observer*** *to quickly and confidently infer the goal.*

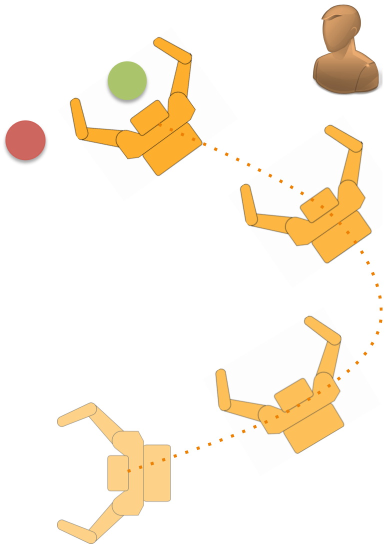


Figure 3.4: Legible motion.

Finally, imagine someone observing the robot as it executes the trajectory from Fig. 3.4. As the robot's hand starts moving along the trajectory, the observer is running an inference, predicting which of the two goals it is reaching for. We denote this inference function that maps (snippets of) trajectories from the set of all trajectories Ξ to goals as:

$$\mathcal{I}_L : \Xi \rightarrow \mathcal{G} \tag{3.5}$$

At the very start of the trajectory from Fig. 3.4, the observer has little *confidence*. However, the intended goal becomes clear *quickly*. This quick and confident inference is the hallmark of legibility.

We thus formalize *legible* motion as motion that enables an observer to *confidently* infer the *correct* goal G after observing a trajectory snippet $\xi_{S\rightarrow Q}$, from S to $Q = \xi(t)$:

$$\mathcal{I}_L(\xi_{S\rightarrow Q}) = G \tag{3.6}$$

This unifies terms like "readable" [210], "understandable" [9], and "anticipatory" [84]. The legible trajectory from Fig. 3.4 is very different from the predictable trajectory from Fig. 3.3, as they stem from inferences in *opposing* directions: $\mathcal{I}_P$ maps goals to trajectories, while $\mathcal{I}_L$ maps trajectories to goals. This is our key insight:

> *Predictability and legibility stem from inferences in opposing directions, which makes them fundamentally different and often contradictory properties of motion.*

THE THEORY OF ACTION INTERPRETATION has a natural connection to our formalism. In goal-directed motion, actions are trajectories and goals are goal configurations. Thus the inference occurring in legibility, from trajectory to goal, $\xi_{S\rightarrow Q} \mapsto G$, relates naturally to the "action-to-goal" inference. Likewise, the inference occurring in predictability, from goal to trajectory, $G \mapsto \xi_{S\rightarrow G}$, relates naturally to "goal-to-action".

We summarize action interpretation in Section 2.3

We present a summary of the connection to psychology in Table 3.1.

In what follows, we present models for the two inferences that enable a robot to quantify predictability and legibility of motion in terms of costs on trajectories that can be optimized. The models

Human Inference Type	**Example**	**Analogy in Motion**	**Property of Motion**
action ↦ goal	... pour beans in grinder ↦ coffee	$\xi_{S\to Q} \mapsto G$	*legibility*
goal ↦ action	coffee ↦ ... pour beans in grinder ...	$G \mapsto \xi_{S\to G}$	*predictability*

Table 3.1: Legibility and predictability as enabling inferences in opposing direction.

are based on cost optimization, maximum entropy, and Bayesian inference, they resonate with the principle of rational action [78, 46], and echo earlier works on action understanding via inverse planning [16].

3.2 *Modeling Predictable Motion via Optimization*

INVOKING THE PRINCIPLE OF RATIONAL ACTION. We model our observer as expecting the robot to act according to the principle of rational action [78, 46, 203, 77]: humans expect other agents, including robots, to act rationally and take the actions that are most justifiable [46] or efficient [47] given a particular situation and a particular goal.

We model the notion of "efficiency" via a cost functional defining what it means to be efficient, as in Fig. 3.5 (top). For example, if the observer expected the robot's hand to move directly towards the object it wants to grasp (as opposed to taking an unnecessarily long path to it), then "efficiency" would be defined by the cost functional penalizing the trajectory's length.

Throughout this thesis, we will refer to the cost functional modeling the observer's expectation as C:

$$C : \Xi \to \mathbb{R}^+$$

with lower costs signifying more "efficient" trajectories.

$C : \Xi \to \mathbb{R}^+$ is the cost functional that models the cost that the observer expects the robot to optimize

C is called a *functional* because it maps functions (trajectories ξ) onto scalars.

A RUNNING EXAMPLE FOR C that we use throughout this work is the integral over squared velocities:

$$C[\xi] = \frac{1}{2}\int ||\dot{\xi}||^2 dt \tag{3.7}$$

Applying the Euler-Lagrange formula for this C, we get $\ddot{\xi} = 0$, meaning the optimal ξ has zero acceleration, thus it has constant velocity and is a linear function of time, $\xi = at + b$. In this example, our observer expects the robot to approximately move in a straight line at constant velocity.

However, if the human observer expects human-like motion, the animation (e.g., [140, 229, 85]) or biomechanics (e.g., [73, 138]) literature can serve to provide better approximations for C.

One challenge is that efficiency of robot motion can have different meanings for different observers. If the observer were willing to provide examples of what they expect, the robot could learn a better C via Inverse Optimal Control [2, 183, 238].

Our user study in Section 3.4 suggests that different people have different expectations about how the *same* robot will move.

Although IOC works for low degree of freedom robots (e.g., mobile robots), it is not tractable in higher dimensional spaces. To produce predictable motion in such spaces beyond our approximation of C from Eq. 3.7, we develop local adaptation methods for demonstrations, as well as study familiarizing users to the robot's motion in Chapter 6.

THE PREDICTABILITY INFERENCE $\mathcal{I}_L$. We model the observer as expecting that the robot will approximately be minimizing C. More precisely, we assume that the observer has some expectation of how costly the robot's trajectory will be:

$$E[C[\xi]] = K$$

If $K = \arg\min_{\xi} C[\xi]$, then the observer is certain that the robot will produce the optimal trajectory accruing to C. A higher K captures more uncertainty that the observer might have.

An expected value implies a probability distribution that the observer has over the space of trajectories (from the starting configuration to the goal). There are many probability distributions that satisfy the constraint above. To select one, we apply the principle of maximum entropy and recover the least biased distribution:

$$\max_{P} \int -P[\xi] \log P[\xi] d\xi \qquad (3.8)$$
$$s.t.\ E[C[\xi]] = K$$

Solving the above results in $P[\xi] \propto \exp\big(-\lambda C[\xi]\big)$ — a Bolzmann distribution. Absorbing the Lagrange multiplier λ into C, we define the following score for predictability:

$$\text{PREDICTABILITY}[\xi] = \exp\big(-C[\xi]\big) \qquad (3.9)$$

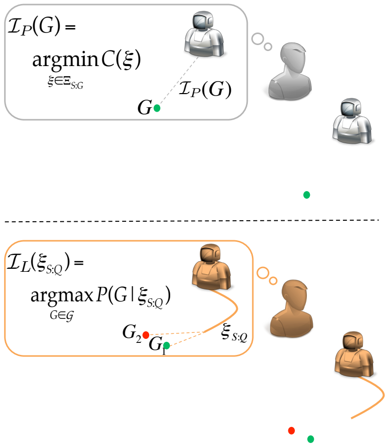


Figure 3.5: We model the observer's expectation as the optimization of a cost function C (above). The observer identifies based on C the most probable goal given the robot's motion so far (below).

Therefore, the observer infers the trajectory with highest probability, i.e., lowest cost, given a goal G — the most *predictable* trajectory:

$$\mathcal{I}_P(G) = \arg \min_{\xi\in\Xi_{S\to G}} C[\xi] \qquad (3.10)$$

We discuss how to optimize C in Chapter 4.

3.3 *Modeling Legible Motion via Optimization*

THE LEGIBILITY INFERENCE $\mathcal{I}_L$. To model $\mathcal{I}_L$ is to model how the observer infers the goal from a snippet of the trajectory $\xi_{S\to Q}$. One way to do so is by assuming that the observer compares the possible goals in the scene in terms of how probable each is given $\xi_{S\to Q}$. This is supported by action interpretation: Csibra and Gergeley [47] argue, based on the principle of rational action, that humans assess which end state would be most efficiently brought about by the observed ongoing action.Taking trajectory length again as an example for the observer's expectation, this translates to predicting a goal because $\xi_{S\to Q}$ moves directly toward it and away from the other goals, making them less probable.

One model for $\mathcal{I}_L$ is to compute the probability for each goal candidate G and to choose the most likely, as in Fig. 3.5(bottom):

$$\mathcal{I}_L(\xi_{S\to Q}) = \arg\max_{G\in\mathcal{G}} P(G|\xi_{S\to Q}) \tag{3.11}$$

Using Bayes' rule, we get:

$$\mathcal{I}_L(\xi_{S\to Q}) = \arg\max_{G\in\mathcal{G}} P[\xi_{S\to Q}|G]P(G) \tag{3.12}$$

We assume a uniform prior. Context from the task and previous actions could be used to obtain a more informed prior.

with $P(G)$ the prior that the observer has over the set of goals $\mathcal{G}$.

The probability of a trajectory snippet $\xi_{S\to Q}$ given a goal is equal to the probability mass of all trajectories going through the snippet and then ending up at the goal, over the probability mass of all trajectories from start to goal (Fig. 3.6). Using that $P[\xi]$ is a Boltzmann distribution, and assuming that the cost C can be different for different goals, leading to a cost C_G for each goal G, we get:

$$P[\xi_{S\to Q}|G] = \frac{\exp\left(-C_G[\xi_{S\to Q}]\right)\int_{\xi_{Q\to G}}\exp\left(-C_G[\xi_{Q\to G}]\right)}{\int_{\xi_{S\to G}}\exp\left(-C_G[\xi_{S\to G}]\right)} \tag{3.13}$$

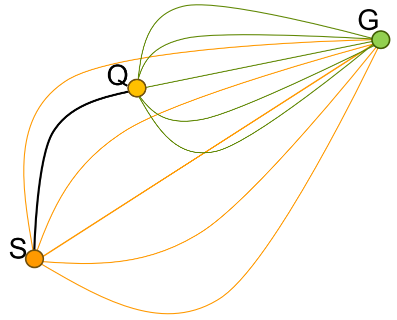


Figure 3.6: $\xi_{S\to Q}$ in black, examples of $\xi_{Q\to G}$ in green, and further examples of $\xi_{S\to G}$ in orange. Trajectories more costly w.r.t. C are less probable.

In low-dimensional spaces, Eq. 3.13 can be evaluated exactly through soft-maximum value iteration [239]. In high-dimensional spaces, where this is expensive, an alternative is to approximate the integral over trajectories using Laplace's method.

First, we approximate $C[\xi_{X\to Y}]$ by its second order Taylor series expansion around $\xi^*_{X\to Y} = \arg\min_{\xi_{X\to Y}} C[\xi_{X\to Y}]$:

$$\begin{aligned} C[\xi_{X\to Y}] \approx C[\xi^*_{X\to Y}] + \nabla C[\xi^*_{X\to Y}]^T(\xi_{X\to Y} - \xi^*_{X\to Y}) + \\ \frac{1}{2}(\xi_{X\to Y} - \xi^*_{X\to Y})^T\nabla^2 C[\xi^*_{X\to Y}](\xi_{X\to Y} - \xi^*_{X\to Y}) \end{aligned} \tag{3.14}$$

Since $\nabla C[\xi^*_{X\to Y}] = 0$ at the optimum, we get

$$\begin{aligned} \int_{\xi_{X\to Y}} \exp\left(-C[\xi_{X\to Y}]\right) \approx \exp\left(-C[\xi^*_{X\to Y}]\right) \\ \int_{\xi_{X\to Y}} \exp\left(-\frac{1}{2}(\xi_{X\to Y} - \xi^*_{X\to Y})^T H_{X\to Y}(\xi_{X\to Y} - \xi^*_{X\to Y})\right) \end{aligned} \tag{3.15}$$

$H_{X\to Y}$ the Hessian of the cost function around $\xi^*_{X\to Y}$.

Evaluating the Gaussian integral leads to

$$\int_{\xi_{X\to Y}} \exp\left(-C[\xi_{X\to Y}]\right) \approx \exp\left(-C[\xi^*_{X\to Y}]\right)\frac{\sqrt{2\pi^k}}{\sqrt{|H_{X\to Y}|}} \tag{3.16}$$

The probability becomes

$$P[\xi_{S\to Q}|G] \approx \frac{\exp\big(-C_G[\xi_{S\to Q}] - C_G[\xi^*_{Q\to G}]\big)\sqrt{|H_{Q\to G}|}}{\exp\big(-C_G[\xi^*_{S\to G}]\big)\sqrt{|H_{S\to G}|}} P(G) \qquad (3.17)$$

If the cost function is quadratic, the Hessian is constant and we get goal predictions by

$$P(G|\xi_{S\to Q}) \propto \frac{\exp\big(-C[\xi_{S\to Q}] - V_G(Q)\big)}{\exp\big(-V_G(S)\big)} P(G) \qquad (3.18)$$

$V_G(Q)$ is the value function. i.e., the cost of the optimal trajectory $\xi^*_{Q\to G}$

Predicting goals by computing the $\arg\max_G P[G|\xi_{S\to Q}]$ using the formula above implements an intuitive principle: if the actor appears to be taking (even in the optimistic case) a trajectory that is a lot costlier than the optimal one to that goal, the goal is likely not the intended one.

Much like teleological reasoning suggests [1], this evaluates how efficient (w.r.t. C) going to a goal is through the observed trajectory snippet $\xi_{S\to Q}$ relative to the most efficient (optimal) trajectory, $\xi^*_{S\to G}$.

In ambiguous situations like the one in Fig. 3.7, a large portion of $\xi^*_{S\to G}$ is also optimal (or near-optimal) for a different goal, making both goals almost equally likely along it. *This is why legibility does not also optimize C — rather than matching expectation, it manipulates it to convey intent.*

The Legibility Cost Functional Based on C. A legible trajectory is one that enables quick and confident predictions. A score for legibility therefore tracks the probability assigned to the robot's actual goal G_R across the trajectory: trajectories are more legible if this probability is higher, with more weight being given to the earlier parts of the trajectory via a function $f(t)$:

$$\text{Legibility}[\xi] = \frac{\int P(G_R|\xi_{S\to\xi(t)})f(t)dt}{\int f(t)dt} \qquad (3.19)$$

$f(t)/\int f(t)dt$ can be analogous to a discount factor in an MDP.

$P(G_R|\xi_{S\to\xi(t)})$ can be computed using C, as in (3.18).

Legibility is Not Predictability. Legibility optimizes a different functional than predictability. This difference in optimization criteria supports the formalism's prediction that the two properties are fundamentally different, and that increasing a trajectory's score with respect to one property can mean decreasing the score with respect to the other.

Chapter 6 discusses optimizing the legibility functional.

The implication of this contradiction is that in planning, a robot cannot assume that being predictable will automatically mean that

[1] Gergely Csibra and GyÃűrgy Gergely. Obsessed with goals: Functions and mechanisms of teleological interpretation of actions in humans. *Acta Psychologica*, 124(1):60 – 78, 2007

it is conveying its intent: *in situations where intentionality is important, such as collaborative tasks, robots should explicitly reason about legibility.*

In what follows, we present an experiment testing this theoretical contradiction in practice, when real users evaluate how legible or predictable a trajectory is.

3.4 *From Theory to Real Users*

The mathematics of predictability and legibility imply that being more legible can mean being less predictable and vice-versa. We set out to verify that this is also true in practice, when we expose subjects to robot motion. We ran an experiment in which we evaluated two trajectories — a theoretically more predictable one ξ_P and a theoretically more legible one ξ_L — in terms of how predictable and legible they are to novices.

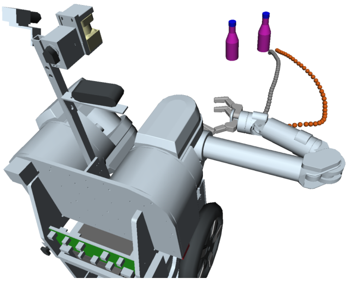


Figure 3.7: The end effector trace for the HERB predictable (gray) and legible (orange) trajectories.

3.4.1 *Hypothesis*

There exist two trajectories ξ_L and ξ_P for the same task such that ξ_P is more predictable than ξ_L and ξ_L is more legible than ξ_P.

3.4.2 *Experimental Setup*

WE CHOSE A TASK like the one in Fig. 3.8: reaching for one of two objects present in the scene. The objects were close together in order to make this an ambiguous task, in which we expect a larger difference between predictable and legible motion.

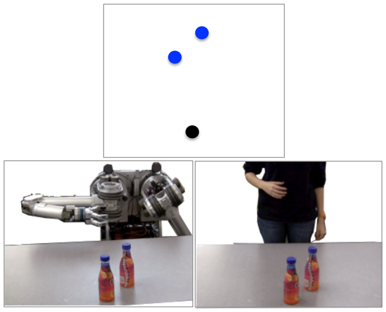


Figure 3.8: We use three characters: a point robot (dot on the screen), a bi-manual manipulator, and a human actor.

WE MANIPULATED TWO VARIABLES: the trajectory type, and the character executing it.
Character: We chose to use three characters for this task (Fig. 3.8) — a simulated point robot, our bi-manual mobile manipulator named HERB [204], and a human — because we wanted to explore the difference between humans and robots, and between complex and simple characters.
Trajectory: We hand designed (and recorded videos of) trajectories ξ_P and ξ_L for each of the characters such that PREDICTABILITY$(\xi_P) >$ PREDICTABILITY(ξ_L) according to Eq. 3.9, but LEGIBILITY$(\xi_P) <$ LEGIBILITY(ξ_L) according to Eq. 3.19.

With the HERB character, we controlled for effects of timing, elbow location, hand aperture and finger motion by fixing them across both trajectories. For the orientation of the wrist, we chose to rotate the wrist according to a profile that matches studies on natural human motion [138, 70]), during which the wrist changes angle more quickly

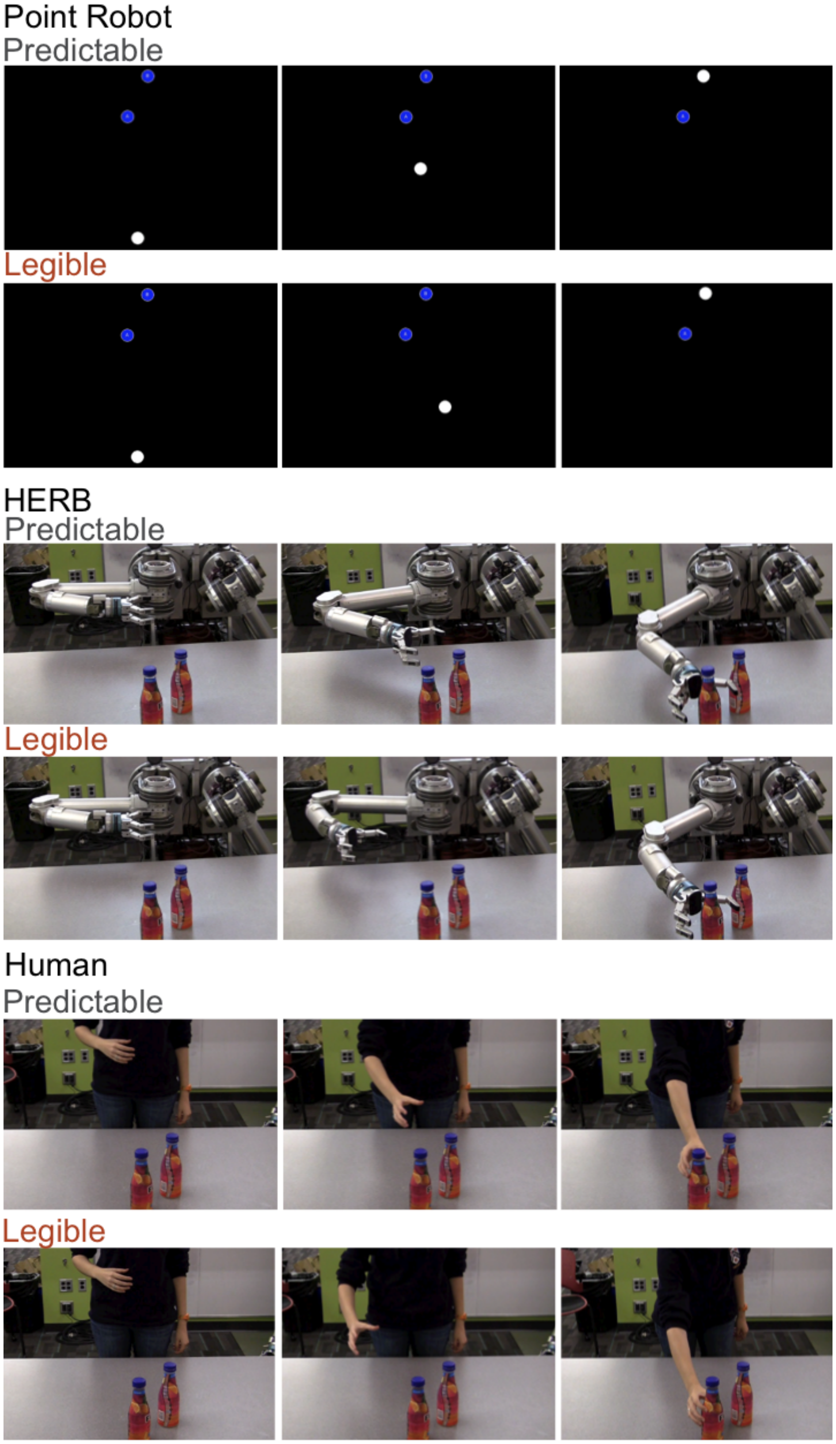

Point Robot
Predictable
Legible
HERB
Predictable
Legible
Human
Predictable
Legible

Figure 3.9: The trajectories for each character.

in the beginning than it does at the end of the trajectory. Fig. 3.7 plots the end effector trace for the HERB trajectories.

The gray trajectory has a larger predictability score ($0.54 > 0.42$), while the orange one has a higher legibility score ($0.67 > 0.63$).

With the human character, we used a natural reach for the predictable trajectory, and we used a reach that exaggerates the hand position to the right for the legible trajectory (much like with HERB or the point robot). We cropped the human's head from the videos to control for gaze effects.

Fig. 3.9 shows the start, end, along with an intermediate waypoint for each trajectory.

We used two dependent measures: predictability and legibility.
Predictability: Predictable trajectories match the observer's expectation. To measure how predictable a trajectory is, we showed subjects the character in the initial configuration and asked them to imagine the trajectory they expect the character will take to reach the goal. We then showed them the video of the trajectory and asked them to rate how much it matched the one they expected, on a 1-7 Likert scale. To ensure that they take the time to envision a trajectory, we also asked them to draw what they imagined on a two-dimensional representation of the scene before they saw the video. We further asked them to draw the trajectory they saw in the video as an additional comparison metric.

We measure *predictability* by asking participants how much the trajectory matched what they predicted.

Legibility: Legible trajectories enable quick and confident goal prediction. To measure how legible a trajectory is, we showed subjects the video of the trajectory and told them to stop the video as soon as they knew the goal of the character. We recorded the time taken and the prediction, and whether they were correct. This measure draws on the protocol used by Gielniak et al. [2] in their research on anticipatory motion.

We measure *legibility* by asking participants to stop the motion when they are confident in the goal.

Subject Allocation. We split the experiment into two subexperiments with different subjects: one about measuring predictability, and the other about measuring legibility.

For the predictability part, the character factor was between-subjects because seeing or even being asked about trajectories for one character can bias the expectation for another. However, the trajectory factor was within-subjects in order to enable relative comparisons on how much each trajectory matched expectation. This lead to three subject groups, one for each character.

We counter-balanced the order of the trajectories within a group to avoid ordering effects.

For the legibility part, both factors were between-subjects because the goal was the same (further, right) in all conditions. This leads to six subject groups.

To eliminate users that do not pay attention to the task and provide random answers, we added a control question, e.g., "What was the color of the point robot?" and disregarded the users who gave wrong answers from the data set.

We recruited a total of 432 subjects (distributed approximately evenly between groups) through Amazon's Mechanical Turk, all from

[2] M.J. Gielniak and A.L. Thomaz. Generating anticipation in robot motion. In *RO-MAN*, pages 449 –454, 31 2011-aug. 3 2011

the United States and with approval rates higher than 95%.

3.4.3 *Analysis*

Overall, the predictable motions were evaluated as more predictable in practice as well. The exception was for HERB, which points to a need of using a better cost C (Chapter 5).

Predictability. In line with our hypothesis, a factorial ANOVA revealed a significant main effect for the trajectory: subjects rated the predictable trajectory ξ_P as matching what they expected better than ξ_L, $F(1,310) = 21.88$, $p < .001$. The main effect of the character was only marginally significant, $F(1,310) = 2{,}91$, $p = .056$. The interaction effect was significant however, with $F(2,310) = 10.24$, $p < .001$. The post-hoc analysis using Tukey corrections for multiple comparisons revealed, as Fig. 3.10 shows, that our hypothesis holds for the point robot (adjusted $p < .001$) and for the human (adjusted $p = 0.28$), but not for HERB.

The trajectories the subjects drew confirm this (Fig. 3.11): while for the point robot and the human the trajectory they expected is, much like the predictable one, a straight line, for HERB the trajectory they expected splits between straight lines and trajectories looking more like the legible one.

For HERB, ξ_L was just as (or even more) predictable than ξ_P.

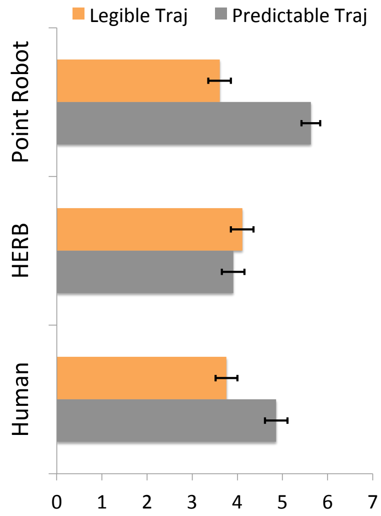


Figure 3.10: Ratings (on Likert 1-7) of how much the trajectory matched the one the subject expected.

Follow-Up Study 1. We conducted an exploratory follow-up study with novice subjects from a local pool to help understand this phenomenon. We asked them to describe the trajectory they would expect HERB to take in the same scenario, and asked them to motivate it. Surprisingly, all 5 subjects imagined a different trajectory, motivating it with a different reason.

Two subjects thought HERB's hand would reach from the right side because of the other object: one thought HERB's hand is too big and would knock over the other object, and the other thought the robot would be more careful than a human. This brings up an interesting possible correlation between legibility and obstacle avoidance. However, as Fig. 6.8 shows, a legible trajectory still exaggerates motion away from the other candidate objects even in if it means getting closer to a static obstacle like a counter or a wall.

Another subject expected HERB to not be flexible enough to reach straight towards the goal in a natural way, like a human would, and thought HERB would follow a trajectory made out of two straight line segments joining on a point on the right. She expected HERB to move one joint at a time. We often saw this in the drawn trajectories with the original set of subjects as well (Fig. 3.11, HERB, Expected).

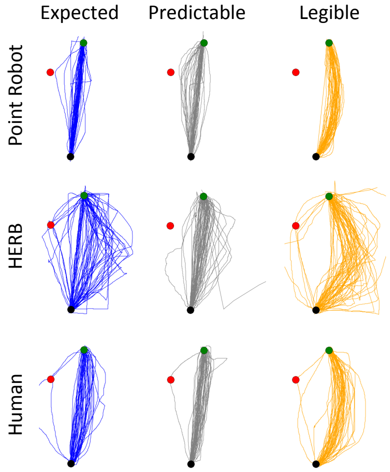


Figure 3.11: The drawn trajectories for the expected motion, for ξ_P (predictable), and for ξ_L (legible).

The other subjects came up with interesting strategies: one thought HERB would grasp the bottle from above because that would work

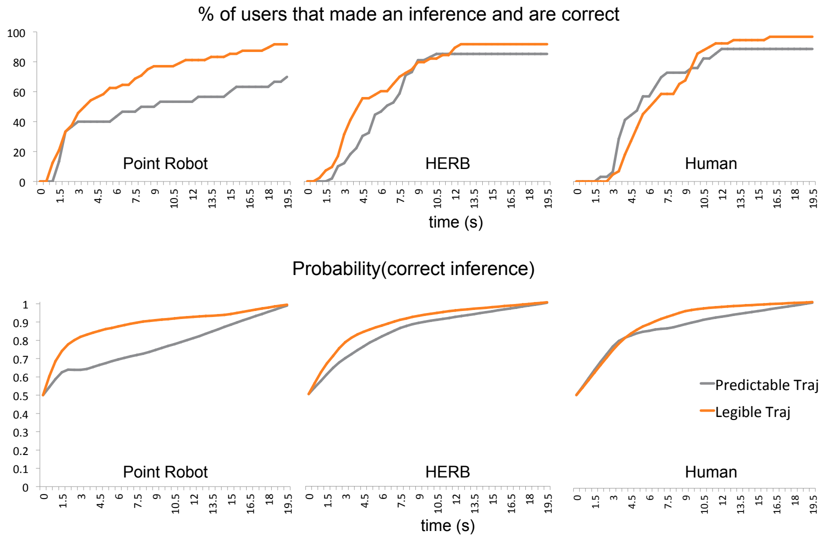


Figure 3.12: Cumulative number of users that responded and were correct (above) and the approximate probability of being correct (below).

better for HERB's hand, while the other thought HERB would use the other object as a prop and push against it in order to grasp the bottle.

Follow-Up Study 2. Many of the participants from the first follow-up study had misconceptions about how the robot would move. This inspired us to test whether it would make a difference to expose participants to a robot motion from a different situation before we ask them which trajectory they expect in the new situation.

We thus ran a last study online ($N = 16$), where participants first watched a video of a predictable motion in a different situation (different location of the target object). This was meant to help give participants some context for how the robot's different joints can move. Then, participants saw two videos, of ξ_P and ξ_L, and chose which better matched their expectation.

Unlike in the original study, now 70% of the participants selected ξ_P as more predictable. A binomial test showed this to be significantly higher than chance ($p = .0251$).

Section 5.3 studies the idea of familiarizing users to robot motion and its limitations in more detail.

Legibility. We collected from each subject the time at which they stopped the trajectory and their guess of the goal. Fig. 3.12 (above) shows the cumulative percent of the total number of subjects assigned to each condition that made a correct prediction as a function of time along the trajectory. With the legible trajectories, more of the subjects tend to make correct predictions faster.

To compare the trajectories statistically, we unified time and correctness into a typical score inspired by the Guttman structure (e.g., [24]): guessing wrong gets a score of 0, and guessing right gets a higher score if it happens earlier.

A factorial ANOVA predicting this score revealed, in line with our hypothesis, a significant effect for trajectory: the legible trajectory had a higher score than the predictable one, $F(1, 241) = 5.62, p = .019$. The means were 6.75 and 5.73, much higher than a random baseline of making a guess independent of the trajectory at uniformly distributed time, which would result in a mean of 2.5 — the subjects did not act randomly. No other effect in the model was significant.

Although a standard way to combine timing and correctness information, this score rewards subjects that gave an incorrect answer 0 reward. This is equivalent to assuming that the subject would keep making the incorrect prediction. However, we know this not to be the case. We know that at the end (time T), every subject would know the correct answer. We also know that at time 0, subjects have a probability of 0.5 of guessing correctly. To account for that, we computed an approximate probability of guessing correctly given the trajectory so far as a function of time — see Fig. 3.12(below). Each subject's contribution propagates (linearly) to 0.5 at time 0 and 1 at time T. The result shows that indeed, the probability of making a correct inference is higher for the legible trajectory at all times.

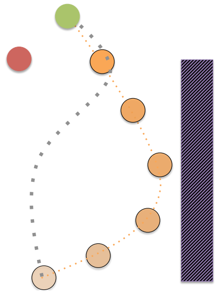


Figure 3.13: Legibility is not obstacle avoidance. Here, in the presence of an obstacle that is not a potential goal, the legible trajectory still moves towards the wall, unlike the obstacle-avoiding one (gray trace).

This effect is strong for the point robot and for HERB, and not as strong for the human character. We believe that this might be a consequence of the strong bias humans have about human motion — when a human moves even a little unpredictably, confidence in goal prediction drops. This is justified by the fact that subjects did have high accuracy when they responded, but responded later compared to other conditions.

Chapter 6 discusses trading off legibility with keeping the motion efficient and preventing over-exaggeration.

In summary, the legible trajectories tended to be more legible, and the predictable trajectories tended to be more predictable (especially when they were not the first motions the participants ever saw the character performed, as in our follow-up). The character factor did not have a significant effect, but it deed seem to influence the outcome to a limited extent.

Limitations. The main limitation is the dependance on C — on the one hand, our example C can make a good straw-man across users; on the other hand, our results suggests that customizing C can be beneficial (Chapter 5). There is also a need to establish how legible to be, and, as we will see in Chapter 6, our model's assumptions only hold in some region beyond which users start predicting a "something else" hypothesis.

Additional studies are required to further test the model in different situations, beyond the one setup we used in this experiment.

3.5 Chapter Summary

This chapter introduced a mathematical formalism for predictable and legible motion. We started with the assumption that the observer expects the robot to approximately optimize a cost function. This induced a probability density function over the space of trajectories given a goal. Bayesian inference and an approximation for tractability then led to a mathematical measure for how legible a trajectory is.

$$\begin{tikzcd}
C \arrow[d] \\
P[\xi|G] \arrow[d] \\
P(G|\xi_{S\to Q}) \arrow[d] \\
\textsc{Legibility}[\xi] \arrow[d] \\
\text{user study}
\end{tikzcd}$$

We also discussed the results of a study designed to test the formalism's prediction that legibility and predictability can be contradictory. Overall, a trajectories more predictable in theory were also more predictable in practice, and trajectories more legible in theory were also more legible in practice.

One exception was HERB's predictable trajectory, which many participants originally perceived as less predictable than its legible counterpart. However, our follow-up study showed that this is no longer the case as soon as participants get to see another example trajectory. In that case, the majority of participants do find the predictable trajectory more predictable.

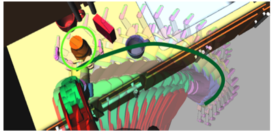
Still, the motion with a lower predictability cost when using the example cost functional C falls short of being predictable enough to all users. 30% of users still vote against it, even after having seen a similar motion before, and, as our in-person study reveals, different people have different notions of what C is.

Chapter 5 introduces two ways to alleviate this issue: learning motion from user-demonstrated trajectories instead of assuming a C, and familiarizing the human with the robot's C — an idea we already used in our follow-up study, but which we will test more thoroughly and for which we will analyze its limitations. But first, we take a first step towards generating predictable and legible motion in the next chapter, by focusing on a common ingredient they both require: trajectory optimization for motion planning (Chapter 4).

4

Trajectory Optimization

So far, we introduced mathematical measures for predictability and legibility of goal-directed motion. In order to generate motion with these properties, the robot needs the ability to generate trajectories with high Predictability and Legibility scores, i.e., the ability to perform *trajectory optimization*.



Trajectory Optimization

In this chapter, we build on a local functional trajectory optimization method (Section 4.1), and introduce two complementary ways of alleviating convergence to poor local optima: expanding good local basins of attraction by adding the flexibility of goal sets (Section 4.2), and learning to initialize the optimizer in a good basin of attraction using prior experience (Section 4.3).

4.1 Functional Gradient Trajectory Optimization

The goal of trajectory optimization in the context of motion planning is to generate a trajectory that optimizes some cost or utility functional (like C from Eq. 3.7 or Legibility from Eq. 3.19) while avoiding collisions with the environment and self-collisions.

Our approach to trajectory optimization is motivated by two ideas:

1. **Gradient information is often available and can be computed inexpensively.** This includes gradients regarding utility, as well as gradients regarding obstacle avoidance, which can be used to actively push the trajectory out of collision. Therefore, our approach uses gradients to iteratively improve an initial (possibly infeasible) trajectory, like a straight line through the configuration space.

   By iteratively following the gradient direction, the trajectory optimizer can start with an infeasible trajectory and bend it out of collision.

   We define our cost functional $\mathcal{U}$ as a combination of a prior term $\mathcal{U}_{prior}$(this will become the predictability or legibility measures in later chapters) and an obstacle avoidance term $\mathcal{U}_{obs}$.

   The obstacle functional is developed as a line integral of a scalar cost field c, defined so that it is invariant to re-timing. Consider a robot arm sweeping through a cost field, accumulating cost as is moves. Regardless of how fast or slow the arm moves through the field, it must accumulate the exact same cost.

The physical intuition of an arm sweeping through a cost field, hints at a further simplification. Instead of computing the cost field in the robot's high-dimensional configuration space, we compute it in its workspace (typically 2-3 dimensional) and use body points on the robot to accumulate workspace cost to compute $\mathcal{U}_{obs}$.

2. **The inner product in the Hilbert space Ξ of trajectories need not be Euclidean.** In fact, because non-Euclidean inner products couple time along the trajectory, they can much better capture the structure of motions, in which any current time point is intimately related to the previous and the next.

   Changing the inner product changes the gradient direction. We use the *natural* gradient, which is *covariant* to reparametrization. The effect is that Euclidean changes are no longer applied independently, but *propagated to the rest of the trajectory*.

Think of the trajectory as an infinite vector. Then $\langle \xi_1, \xi_2 \rangle = \xi_1^T M \xi_2$, with M and Hermitian positive-definite matrix. If $M = I$, the inner product is Euclidean and treats each time point independently. Off-diagonal elements of M represent relations between different time points.

The *natural* gradient is an operator analogous to the ordinary gradient, but depends on the geometry of the space (the Riemannian metric) and not on the parametrization. When the space is a curved manifold, the natural gradient is the steepest direction of the target function, and is equal to the ordinary gradient transformed by the inverse of the Riemannian metric tensor, as we show in Eq. 4.8. The natural gradient is commonly used in MLE with the Fisher metric — here we use it with a metric that better captures the geometry of the trajectory space than the Euclidean inner product.

In what follows, we introduce the obstacle cost functional, a useful inner product choice, and the functional gradient descent algorithm (named CHOMP — "Covariant Hamiltonian Optimization for Motion Planning"; first introduced in [184], journal version followed [240]).

4.1.1 *The Cost Functional*

The cost functional separately measures two complementary aspects of motion planning:

$$\mathcal{U}[\xi] = \mathcal{U}_{prior}[\xi] + \alpha \mathcal{U}_{obs}[\xi]$$

THE FIRST TERM, $\mathcal{U}_{prior}$, is a prior term. It dictates how the robot should move in the absence of any obstacle information. This can be the predictability and legibility measures, or any term that captures efficiency for the robot or smoothness, encompassing dynamical properties like velocities, accelerations, jerk, etc.[1]

We discuss gradients on this prior term separately for predictability (Chapter 5) and legibility (Chapter 6).

THE SECOND TERM, $\mathcal{U}_{obs}$, is the obstacle term. It encourages collision free trajectories by penalizing parts of the robot that are close to obstacles, or already in collision.

Let $\mathcal{B} \subset \mathbb{R}^3$ be the set of points on the exterior body of the robot and let $x : \mathcal{C} \times \mathcal{B} \to \mathbb{R}^3$ denote the forward kinematics, mapping a robot configuration $q \in \mathcal{Q}$ and a particular body point $u \in \mathcal{B}$ to a point $x(q, u)$ in the workspace. Furthermore, let $c : \mathbb{R}^3 \to \mathbb{R}$ be a

$\mathcal{B}$: the set of body points
x: the forward kinematics mapping
c: the workspace obstacle cost

---

[1] An example prior is C from Eq. 3.7. Our theory extends straightforwardly to higher-order derivatives such as accelerations or jerks.

workspace cost function that penalizes the points inside and around the obstacles. We define this workspace cost function in terms of the Euclidean distance to the boundaries of obstacles, $\mathcal{D}(x)$:

$$c(x) = \begin{cases} -\mathcal{D}(x) + \frac{1}{2}\varepsilon, & \text{if } \mathcal{D}(x) < 0 \\ \frac{1}{2\varepsilon}(\mathcal{D}(x) - \varepsilon)^2, & \text{if } 0 < \mathcal{D}(x) \leq \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

The cost of a point in the workspace smoothly drops to zero as a distance of the allowable threshold ε is reached.

The obstacle objective $\mathcal{U}_{obs}$ is an integral that collects the cost encountered by each workspace body point in $\mathcal{B}$ on the robot as it sweeps across the trajectory. Specifically, it computes the *arc length* parametrized line integral of each body point's path through the workspace cost field and integrates those values across all body points:

$$\mathcal{U}_{obs}[\xi] = \int_0^1 \int_{\mathcal{B}} c\left(\, x(\xi(t), u)\, \right) \left\| \frac{d}{dt} x(\xi(t), u) \right\| du\, dt \tag{4.2}$$

The workspace cost function c is multiplied by the norm of the workspace velocity of each body point, transforming what would otherwise be a simple line integral into its corresponding arc-length parametrization.

The arc-length parametrization ensures that the obstacle objective is invariant to re-timing of the trajectory (i.e., moving along the same path at a different speed). The benefit is that the objective functional provides no incentive to directly alter the trajectory's speed through the workspace for any point on the robot.

The functional Euclidean gradient of the obstacle term, derived in [184], is

$$\bar{\nabla}\mathcal{U}_{obs}[\xi] = \int_{\mathcal{B}} J^{\mathrm{T}} \|x'\| \left[(I - \hat{x}'\hat{x}'^{\mathrm{T}})\nabla c - c\kappa \right] du, \tag{4.3}$$

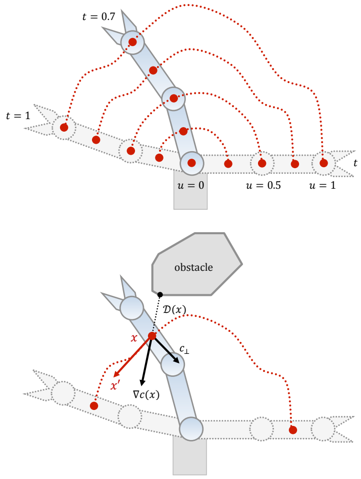


Figure 4.1: The obstacle cost tracks a set of body points through time. Each body point at each time point has a workspace gradient, which Eq. 4.3 compounds in a trajectory gradient.

4.1.2 *An Example Inner Product*

Optimizers often implicitly use a Euclidean inner product:

$$\langle \xi_1, \xi_2 \rangle_I = \int \xi_1(t)^T \xi_2(t) dt \tag{4.4}$$

If we write trajectories as (possibly infinitely long) vectors of configurations this becomes:

$$\langle \xi_1, \xi_2 \rangle_I = \xi_1^T \xi_2 \tag{4.5}$$

Euclidean inner products treat each point in time along the trajectory as being independent from the rest. Consider the norm of a trajectory — the inner product with itself. Time t only interacts with itself, it is not affected by any other time. With a Euclidean inner product, trajectories are no more than sequences of independent configurations, each one amnesic of the past and ignorant of the future.

However, trajectories should be more than that. Time forces configurations along the way to relate to each other. The inner product can capture this by relating time t with more than just itself. An example is

$$\langle \xi_1, \xi_2 \rangle_A = \xi_1^T A \xi_2 \tag{4.6}$$

with A having off-diagonal non-zero elements that relate the current time point with the previous and the next.

A particular A that we use throughout the thesis is the Hessian of integral over squared velocities along the trajectory:

$$A = \nabla^2 \int ||\dot{\xi}||^2 dt = K^T K \tag{4.7}$$

with K the finite differencing matrix (accounting for a constant start and goal configuration):

$$K = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ & & & \dots & & \\ 0 & 0 & 0 & \dots & -1 & 1 \\ 0 & 0 & 0 & \dots & 0 & -1 \end{bmatrix}$$

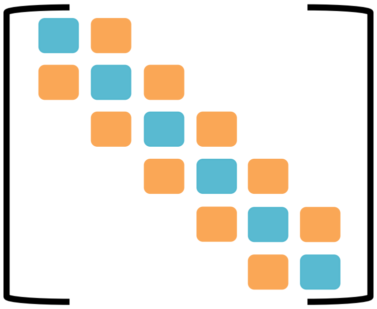


Figure 4.2: A couples time along the trajectory, turning the trajectory into an elastic band: when a Euclidean gradient would pull one single point away from the rest of the trajectory, the natural gradient pulls the entire trajectory with it (details in Section 4.1.3.

Using A instead of the Euclidean inner product changes how we compute distances between trajectories. For instance, looking at Fig. 4.3:

$$||a - b||_I < ||a - c||_I$$

but

$$||a - b||_A > ||a - c||_A$$

The second property is useful because trajectory c is a smoother deformation of a than b is. Our optimization algorithm in Section 4.1.3 is informed by this preference and takes gradient steps in the correct Hilbert space.

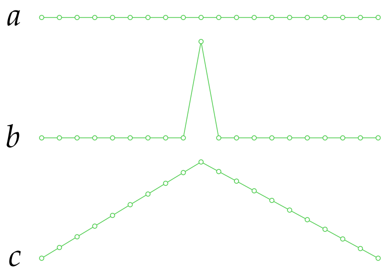


Figure 4.3: A Euclidean inner product makes trajectory b closer to a than c is. In contrast, our example inner product makes c closer.

This inner product takes advantage of the underlying geometry of the space of trajectories: that time along the trajectory is not independent. More complex geometries can also be used, including those that do not correspond to a fixed inner product A, but that have a different A for any two trajectories. Examples include metrics that act differently around obstacles or singularities, or metrics in the workspace. Workspace metrics in particular depend on the forward kinematics mapping, which changes with the configuration.

4.1.3 *Algorithm (CHOMP)*

WE PERFORM NATURAL GRADIENT DESCENT. We start with an initial trajectory, ξ_0, and iteratively move through Ξ following the direction of the natural gradient.

Let $\nabla_{\xi_i}\mathcal{U}$ be the Euclidean gradient of $\mathcal{U}$ about ξ_i, computable by the Euler-Lagrange formula:

$$\frac{\partial \mathcal{U}}{\partial \xi} - \frac{d}{dt}\frac{\partial \mathcal{U}}{\partial \dot{\xi}}$$

Let $\nabla^A_{\xi_i}\mathcal{U}$ be the gradient in the Hilbert space in which A is the inner product. Then the following holds:

$$\nabla^A_{\xi_i}\mathcal{U} = A^{-1}\nabla_{\xi_i}\mathcal{U} \tag{4.8}$$

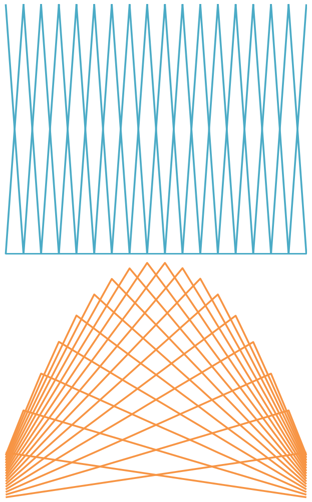


Figure 4.4: The top plots the columns of the identity matrix (each time point is independent), whereas the bottom plots the columns of A^{-1}, for $A = K^T K$ (a change at one time point leads to a propagation to the rest of the trajectory).

One way to see this is to write the first order Taylor series expansion in two ways: one using the Euclidean gradient and inner product, and one using the natural gradient and its associated inner product:

$$\mathcal{U}[\xi] \approx \mathcal{U}[\xi_i] + \langle \xi - \xi_i, \nabla_{\xi_i}\mathcal{U}\rangle_I$$
$$\mathcal{U}[\xi] \approx \mathcal{U}[\xi_i] + \left\langle \xi - \xi_i, \nabla^A_{\xi_i}\mathcal{U}\right\rangle_A$$

Therefore, the two gradients are related by:

$$(\xi - \xi_i)^T \nabla_{\xi_i}\mathcal{U} = (\xi - \xi_i)^T A \nabla^A_{\xi_i}\mathcal{U},\ \forall \xi \in \Xi$$

This relation implies that the natural gradient $\nabla^A_{\xi_i}\mathcal{U}$ satisfies Eq. 4.8.

At each iteration, we follow the direction of the natural gradient:

$$\xi_{i+1} = \xi_i - \frac{1}{\eta}\nabla^A_{\xi_i}\mathcal{U} = \xi_i - \frac{1}{\eta}A^{-1}\nabla_{\xi_i}\mathcal{U} \tag{4.9}$$

η controls the step size.

AN ALTERNATIVE DERIVATION of the update rule comes from minimizing the first order approximation of $\mathcal{U}$ about the current trajectory, ξ_i, subject to a regularization term that prevents the optimizer from going too far away from ξ_i:

$$\min_{\xi} \mathcal{U}[\xi_i] + \langle \xi - \xi_i, \nabla_{\xi_i}\mathcal{U}\rangle_I + \frac{\eta}{2}||\xi - \xi_i||^2_A \tag{4.10}$$

This is similar to a second-order Newton method, but uses a fixed norm A instead of computing the Hessian.

This is a quadratic cost in ξ, and we obtain the global optimum by taking its gradient and setting it to 0:

$$\nabla_{\xi_i}\mathcal{U} + \eta A(\xi - \xi_i) = 0 \tag{4.11}$$

$$\xi_{i+1} = \xi_i - \frac{1}{\eta}A^{-1}\nabla_{\xi_i}\mathcal{U} \tag{4.12}$$

THE UPDATE RULE HAS A VERY INTUITIVE INTERPRETATION. It propagates the entries of the Euclidean gradient, at each time t, down to the start and the end of the trajectory. The propagation is dictated by the inverse of the norm. Fig. 4.4 shows how the norm propagates the gradient for two different choices: the Euclidean gradient (no propagation), and the A from Eq. 4.7.

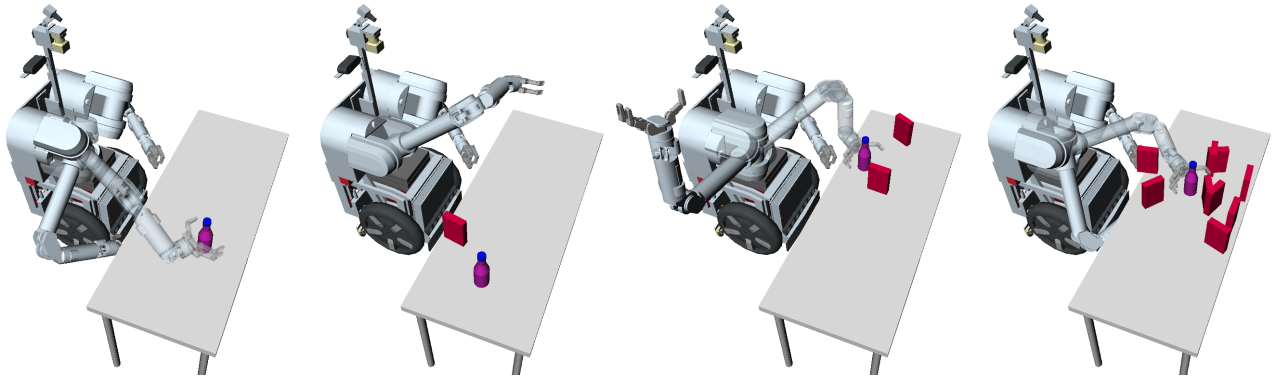


Figure 4.5: Grasping in clutter scenes, with different starting configurations, target object locations, and clutter distribution (from left to right: no clutter, low, medium and high clutter).

4.1.4 *Gradient-Based Optimization Experiments*

For solving motion planning problems in high-dimensional spaces, a historical dichotomy exists between trajectory optimization (e.g., CHOMP) and sampling-based approaches (e.g., the Rapidly-exploring Random Tree [136]). Recently, algorithms such as RRT* [118] have brought optimization to sampling-based planners.

Here, we evaluate the performance of CHOMP on motion planning problems commonly encountered in real-world manipulation domains, and comparing it with such sampling-based approaches. We focus on a motion planning problem which arises in common manipulation tasks: *planning to a pre-grasp pose among clutter*.[2]

EXPERIMENTAL DESIGN. We explore day-to-day manipulation task of grasping in cluttered environments. For the majority of our experiments, we use a canonical grasping in clutter problem: the robot is tasked with moving to grasp a bottle placed on a table among a varying number of obstacles, as in Fig. 4.5.

Note on experimental design: It is important to observe that the experiments presented in this section are not "apples to apples" comparisons in that we are juxtaposing a (local) optimization algorithm with global randomized search algorithms. Obviously the effectiveness of our approach depends strongly upon the inherent structure underlying the planning problem, including the sparsity and regularity of obstacles. Certainly, there exist any number of maze-like motion planning problems for which CHOMP is ill-suited. However, as we hypothesize below, it can come to fill some of the space which has until recently been occupied by sampling-based methods; hence, we feel the comparison between heterogeneous systems is well motivated.

We test the following hypotheses:

H1: *CHOMP can solve many structured, day-to-day manipulation tasks, and it can do so very fast.*

H2: *For a large number of structured, day-to-day manipulation tasks, CHOMP obtains a feasible, low-cost trajectory in the same time that an RRT obtains a feasible, high-cost trajectory.*

We compare CHOMP, RRT and RRT*. To ensure fairness of the comparison, we conduct the experiments in a *standard* simulation environment — OpenRAVE [55] version 0.6.4, and use the *standard* implementations for RRT (bi-directional RRT-Connect) and RRT* from the Open Motion Planning Library (OMPL) version 0.10.2[3].

We run each algorithm on each problem 20 times, for 20 seconds each (single-thread CPU time). The RRT shortcuts the path (using the shortcutting method available in OpenRAVE) until the final time is

[2] A pre-grasp pose is an arm configuration which positions the hand in a pre-grasp position relative to an object.

[3] The bug fix for RRT* (path improvement) in 0.11.1 did not alter the results on our problems, for our time intervals. We did verify that the issue was indeed fixed by finding problems on which the RRT* did eventually improve the path.

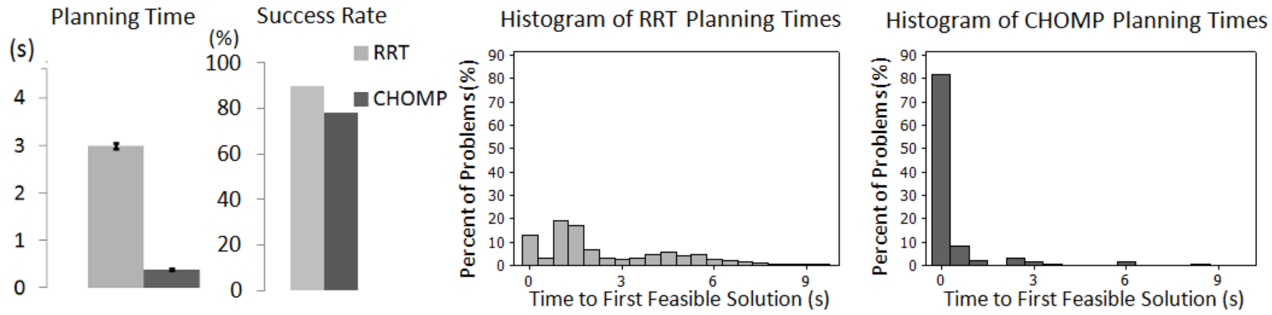


Figure 4.6: From left to right: a paired time comparison between RRT and CHOMP when both algorithms succeed, success rates for both algorithms within the 20 s time interval and the planning time histograms for both algorithms. In the time comparison chart on the left, each data point is one run of the RRT algorithm vs. the discrete run of CHOMP on a problem. Due to the large number of data points, the standard error on the mean is very small.

reached. We measure at each time point if the algorithm has found a *feasible* solution, and we use the *path length* for cost to ensure a fair comparison, biased towards the randomized planners. This is the cost that the RRT shortcutting method optimizes, but not directly the cost that CHOMP optimizes. Instead, CHOMP minimizes sum squared velocities (which correlates to, but is different from, path length), while pulling the trajectory far from obstacles.

We created grasping in clutter problems with varying features: starting configurations, target locations and clutter distributions. We split the problems into a training and testing set, such that no testing problem has any common features with a training one. This is important, as it tests true generalization of the parameters to different problems. We used the training set to adjust parameters for all algorithms, giving each the best shot at performing well. We had 170 testing problems, leading to 3400 runs of each algorithm. Below, we present the results for the deterministic version of CHOMP vs. RRT, and then discuss the comparison with RRT*.

Time to Produce a Feasible Solution. Supporting **H1**, CHOMP (the deterministic version) succeeded on about 80% of the problems, with an average time of 0.34s ($SEM = 0.0174$). On problems where both CHOMP and RRT succeed, CHOMP found a solution 2.6 seconds faster, and the difference is statistically significant (as indicated by a paired t-test, $t(2586) = 49.08$, $p < 0.001$). See Fig. 4.6 for the paired time comparison.

Overall, CHOMP has a lower success rate than an RRT on these problems. When it does succeed, it does so faster.

The CHOMP times do not include the time to compute the Signed Distance Field from the voxelized world (which the robot acquires in practice through a combination of cached voxelized static environments and voxel grids obtained online via laser scans). The SDF computation takes an average of 0.1 seconds.

Collision Checking — The Grain of Salt. The time taken by the RRT heavily depends on the time it takes to perform collision checks. Our implementation uses OpenRAVE for collision checking, and the average time for a check was approximately 444 microseconds (averaged over 174 million checks).[4]

RRT may improve with recent, more advanced collision checkers (e.g., FCL [171]). *For example, if collision checking were 5 times faster (an optimistic estimate for state-of-the-art performance), the difference in success rate would be much higher in favor of the RRT, and the planning time when both algorithms succeed would become comparable, with an estimated average difference of only* 0.2*s in favor of CHOMP.*

H2, as we will see in following section, would remain valid: for many problems (namely 78%), CHOMP produces a low-cost feasible trajectory in the same time that an RRT produces a high-cost feasible trajectory.

Overall, CHOMP produces a better solution faster on the majority of problems in our test set.

COST AND FEASIBILITY COMPARISON WHEN THE RRT RETURNS ITS FIRST SOLUTION. 3067 of the 3400 RRT runs yielded feasible trajectories. For every successful RRT run, we retrieved the CHOMP trajectory from the same problem at the time point when the RRT obtained a solution. In 78% of the cases, the CHOMP trajectory was feasible, and its path length was on average 57% lower. This difference in path length was indeed significant ($t(2401) = 65.67$, $p < 0.001$): *in* 78% *of the cases, in the same time taken by an RRT to produce a feasible solution, CHOMP can produce a feasible solution with significantly lower cost (**H2**).*[5]

TIME BUDGETS. In practice, planners are often evaluated within fixed time budgets. In this comparison, we take that perspective and allow each planner a fixed planning time, and evaluate its result (for both feasibility and path length).

We found that the relative performance of CHOMP and RRT depends greatly on the time budget allotted (and of course, on the collision checker). For CHOMP, we run iterations until such time that a final full-trajectory collision check will finish before the given budget; the check is then performed, and the result is reported.[6] For the RRT, we simply stop planning or shortcutting at the end of the budget. We evaluated time budgets of 1, 2, 3, 5, 10, and 20 s. The summary of these results is shown in Table 4.1.

The results illustrate the differences between the planners. For short time budgets (< 5 s), the deterministic CHOMP has a higher success rate than the RRT; however, it plateaus quickly, and does not exceed 75% for any budget. The RRT continues to improve, with a 90.2% success rate within the longest budget. Across all feasible solutions for all budgets, CHOMP significantly outperforms the RRT when evaluated by path length.

---

[4] This is faster than the times reported in the benchmark comparison from [186] for an easier problem, indicating that our results for the RRT are indicative of its typical performance.

[5] Note that that the CHOMP trajectories evaluated here were not the ones with the smallest path length: the algorithm is optimizing a combination of a smoothness and an obstacle cost. Therefore, CHOMP is increasing the path length even after the trajectory becomes feasible, in order to keep it far from obstacles.

[6] Note that a CHOMP trajectory can oscillate between feasible and infeasible during optimization; it may be the case that an infeasible CHOMP result was in fact feasible at an earlier time, but the algorithm is unaware of this because it only performs the expensive collision check right before it returns.

Time	Success (Percentage)		Average Path Length (radians)	
Budget	RRT	CHOMP	RRT	CHOMP
1 s	16.5	24.7	4.37	3.69
2 s	47.0	68.2	6.85	4.89
3 s	57.1	70.6	6.64	4.94
5 s	66.3	74.1	6.69	5.00
10 s	88.0	74.7	6.79	5.03
20 s	90.2	74.7	6.58	5.03

Table 4.1: Comparison of CHOMP and RRT for different time budgets.

Comparison with Randomized Optimal Motion Planning (RRT*). We compared the performance of CHOMP and Bi-Directional RRT-Connect to the RRT* implementation in OMPL. The RRT* range (step size) parameter was set equal to that of the RRT (corresponding to a workspace distance of 2 cm). We chose other algorithm parameters (goal bias, ball radius constant, and max ball radius) as directed by the implementation documentation.

RRT* had a 5.97% success rate on our testing suite of clutter problems. When it did succeed, it found its first feasible solution after an average of 6.34s, and produced an average path length of 11.64 rad. On none of our testing problems was it able to improve its first path within the 20s time budget (although we did verify that for other problems, this does happen with a long enough time budget).

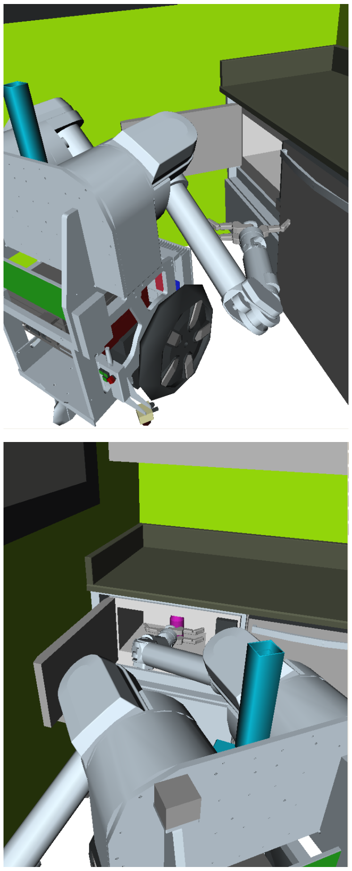


Figure 4.7: The start and the goal for a complex problem of reaching into the back of a narrow microwave. The robot is close to the corner of the room, which makes the problem particularly challenging because it gives the arm very little space to move through. The goal configuration is also very different from the start, requiring an "elbow flip". Two starts were used, one with a flipped turret (e.g., J1 and J3 offset by π, and J2 negated), leading to very different straight-line paths.

Beyond Grasping in Clutter. Our experiments so far focused on grasping an object surrounded by clutter. But how does CHOMP perform on more complex tasks, defined by narrow spaces? To explore this question, we investigated the algorithm's performance on the problem setup depicted in Fig. 4.7: reaching to the back of a narrow microwave placed in a corner, with little free space for the arm to move through. We ran CHOMP and BiRRT-Connect for 8 different scenarios (with different start and goal IK configurations). CHOMP was able to solve 7 of the 8 scenarios, taking an average of 1.04 seconds. The RRT had a total success rate of 67.1%, taking an average of 63.36 seconds to first-feasible when it succeeds. On the problem for which CHOMP failed, the RRT had a 10% success rate. A collision check here took an average of 2023 microseconds (requiring a speed up of 60x to make the RRT first-feasible time equal to that of CHOMP).

In summary, our results suggest that for many real-world problems, a trajectory optimizer will often retrieve better paths than a randomized motion planner, given the same time budget.

LIMITATIONS. Trajectory optimization is local and there are certainly tasks on which convergence to high-cost local optima is problematic, and on which a randomized motion planner would be much more effective.

Because interaction with humans demands optimization, and because randomized optimal motion planning did not outperform gradient optimization for the types of problems we tested on, in what follows we focus on ways to alleviate the issue of convergence to high-cost optima.

4.2 *Optimizing with Constraints*

In many real-world problems, the ability to plan a trajectory from a starting configuration to a goal configuration that avoids obstacles is sufficient. However, there are problems that impose additional constraints on the trajectory, like carrying a glass of water that should not spill, lifting a box with both hands without letting the box slip, or not becoming too unpredictable when optimizing for legibility[7].

In this section, we derive an extension of the optimizer that can handle trajectory-wide equality constraints, and show its intuitive geometrical interpretation. We then focus on a special type of constraint, which only affects the endpoint of the trajectory. This type of constraint enables the optimizer to plan to a set of possible goals rather than the typical single goal configuration, which adds more flexibility to the planning process and increases the chances of converging to a low-cost trajectory, as in Fig. 4.8.

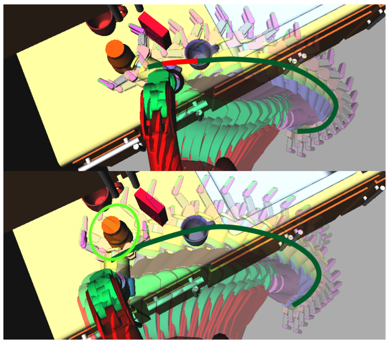


Figure 4.8: Top: The trajectory found when using specified single goal. The optimizer cannot avoid collision with the red box. Bottom: A feasible trajectory found by an optimizer that can take advantage of a goal set.

4.2.1 *Trajectory-Wide Constraints*

We assume that we can describe a constraint on the Hilbert space of trajectories in the form of a nonlinear differentiable vector valued function $\mathcal{H} : \Xi \to \mathbb{R}^k$, for which $\mathcal{H}[\xi] = 0$ when the trajectory ξ satisfies the required constraints.

At every step, we optimize the regularized linear approximation of $\mathcal{U}$ from (4.10), subject to the nonlinear constraints $\mathcal{H}[\xi] = 0$:

$$\xi_{i+1} = \arg\min_{\xi \in \Xi} \mathcal{U}[\xi_i] + \mathcal{U}[\xi_i]^T(\xi - \xi_i) + \frac{\eta}{2}\|\xi - \xi_i\|_A^2 \quad (4.13)$$
$$s.t. \;\; \mathcal{H}[\xi] = 0$$

We can also handle inequality constraints by tracking which constraints are active at every iteration.

WE FIRST OBSERVE that this problem is equivalent to the problem of taking the unconstrained solution in Eq. 4.12 and projecting it onto the constraints. This projection, however, measures distances not with respect to the Euclidean norm, but with respect to the Hilbert space norm A. To show this, we rewrite the objective:

---

[7] we discuss this in Section 6.2

$$\min \mathcal{U}[\xi_i] + \nabla\mathcal{U}[\xi_i]^T(\xi - \xi_i) + \frac{\eta}{2}\|\xi - \xi_i\|_A^2 \Leftrightarrow$$

$$\min \nabla\mathcal{U}[\xi_i]^T(\xi - \xi_i) + \frac{\eta}{2}(\xi - \xi_i)^T A(\xi - \xi_i) \Leftrightarrow$$

$$\min \left(\xi_i - \frac{1}{\eta_i}A^{-1}\nabla\mathcal{U}[\xi_i] - \xi\right)^T A \left(\xi_t - \frac{1}{\eta_i}A^{-1}\nabla\mathcal{U}[\xi_i] - \xi\right)$$

.

The problem can thus be written as:

$$\begin{aligned} \xi_{t+1} = \arg\min_{\xi\in\Xi} \|\xi_t \overbrace{-\frac{1}{\eta}A^{-1}\nabla\mathcal{U}[\xi_i]}^{\textbf{unconstr.}\ (4.12)} -\xi\|_A^2 \\ s.t.\ \ \mathcal{H}[\xi] = 0 \end{aligned} \tag{4.14}$$

Project the unconstrained step onto the constraint: find the closest trajectory to the one obtained by taking an unconstrained step, subject to the constraint.

This interpretation will become particularly relevant in the next section, which uncovers the insight behind the update rule we will obtain by solving Eq. 4.13.

To DERIVE A CONCRETE UPDATE RULE for Eq. 4.13, we linearize $\mathcal{H}$ around ξ_i:

$$\mathcal{H}[\xi] \approx \mathcal{H}[\xi_i] + \frac{\partial}{\partial\xi}\mathcal{H}[\xi_i](\xi - \xi_i) = B(\xi - \xi_i) + b$$

$B = \frac{\partial}{\partial\xi}\mathcal{H}[\xi_i]$ is the Jacobian of the constraint functional evaluated at ξ_t and $b = \mathcal{H}[\xi_i]$.

The Lagrangian of the constrained gradient optimization problem in Eq. 4.13, now with linearized constraints, is

$$\mathcal{L}_g[\xi,\lambda] = \mathcal{U}[\xi_i] + \nabla\mathcal{U}[\xi_i]^T(\xi - \xi_i) + \frac{\eta}{2}\|\xi - \xi_i\|_A^2 + \lambda^T(B(\xi - \xi_i) + b)$$

and the corresponding first-order optimality conditions are:

$$\begin{cases} \nabla_\xi\mathcal{L}_g = \nabla\mathcal{U}[\xi_i] + \eta A(\xi - \xi_i) + B^T\lambda = 0 \\ \nabla_\lambda\mathcal{L}_g = B(\xi - \xi_i) + b = 0 \end{cases} \tag{4.15}$$

Since the linearization is convex, the first order conditions completely describe the solution, enabling the derivation of a new update rule in closed form. If we denote $\frac{\lambda}{\eta} = \gamma$, from the first equation we get:

$$\xi = \xi_i - \frac{1}{\eta}A^{-1}\nabla\mathcal{U}[\xi_i] - A^{-1}B^T\gamma$$

Substituting in the second equation:

$$\gamma = (BA^{-1}B^T)^{-1}(b - \frac{1}{\eta}BA^{-1}\nabla\mathcal{U}[\xi_i])$$

Using γ in the first equation, we solve for ξ:

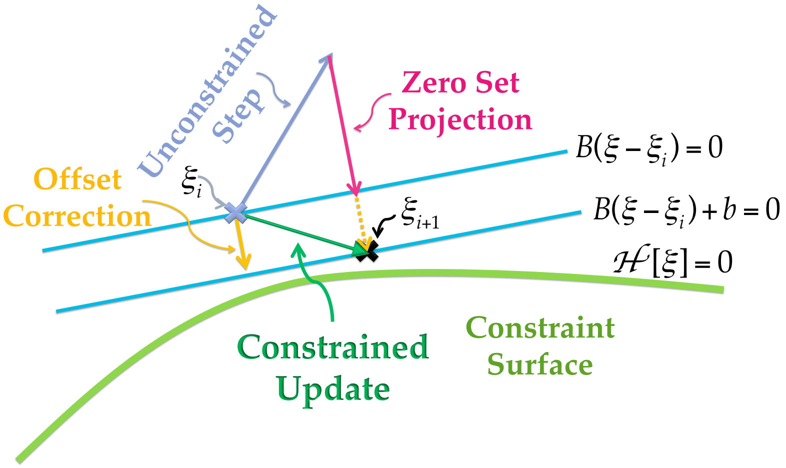


Figure 4.9: The constrained update rule takes the unconstrained step and projects it w.r.t. A onto the hyperplane through ξ_i parallel to the approximated constraint surface (given by the linearization $B(\xi-\xi_t)+b=0$). Finally, it corrects the offset between the two hyperplanes, bringing ξ_{i+1} close to $\mathcal{H}[\xi]=0$.

$$\xi = \xi_i \overbrace{-\frac{1}{\eta}A^{-1}\nabla\mathcal{U}[\xi_i]}^{\textbf{unconstr. } (4.12)} + \overbrace{\frac{1}{\eta}A^{-1}B^T(BA^{-1}B^T)^{-1}BA^{-1}\nabla\mathcal{U}[\xi_i]}^{\textbf{zero set projection}} \overbrace{-A^{-1}B^T(BA^{-1}B^T)^{-1}b}^{\textbf{offset correction}} \tag{4.16}$$

The labels on the terms above hint at the goal of the next section, which provides an intuitive geometrical interpretation for this update rule.

4.2.2 *Geometrical Interpretation*

Looking back at the constrained update rule in Eq. 4.16, we can explain its effect by analyzing each of its terms individually. Gaining this insight not only leads to a deeper understanding of the algorithm, and relates it to an intuitive procedure for handling constraints in general. By the end of this section, we will have mapped the algorithm indicated by Eq. 4.16 to the projection problem in Eq. 4.14: take an unconstrained step, and then project it back onto the feasible region.

The components of the update rule from Eq. 4.16 can be mapped to taking an unconstrained step, and then projecting it onto the approximation of the constraint manifold (in two stages), as predicted by Eq. 4.14.

We split the update rule in three parts, depicted in Fig. 4.9: take the unconstrained step, project it onto a hyperplane that passes through the current trajectory and is parallel to the approximation of the constraint surface, and finally, correct the offset between these two hyperplanes:

1. The first term computes the **unconstrained** step: smooth the unconstrained Euclidean gradient $\nabla\mathcal{U}[\xi_i]$ through A^{-1} and scale it, as in Eq. 4.12. Intuitively, the other terms will need to adjust this step, such that the trajectory obtained at the end of the iteration,

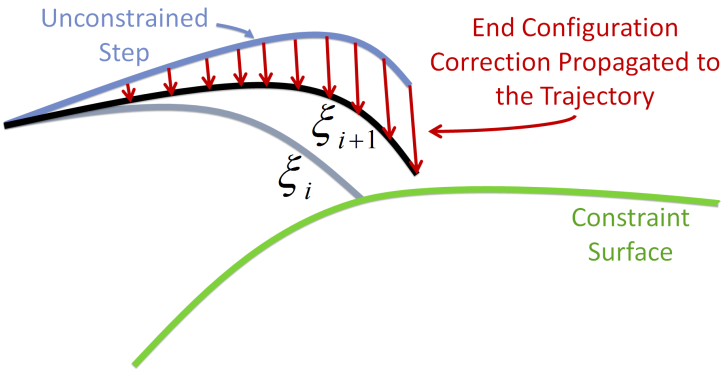


Figure 4.10: One iteration of the goal set version of the optimizer: take an unconstrained step, project the final configuration onto the constraint surface, and propagate that change to the rest of the trajectory.

ξ_{i+1}, is feasible. Therefore, these terms must implement the projection onto the constraint with respect to A, as shown in Eq. 4.14.

2. Linearizing $\mathcal{H}$ provides an approximation of the constraint surface, given by $B(\xi - \xi_i) + b = 0$. The current trajectory, ξ_i, lies on a parallel hyperplane, $B(\xi - \xi_i) = 0$.[8] What the second term in the update rule does is to **project** the unconstrained increment onto the **zero set** of $B(\xi - \xi_i)$ with respect to the metric A, as depicted in Fig. 4.9.

   Formally, the term is the solution to the problem that minimizes the adjustment to the new unconstrained trajectory (w.r.t. A) needed to satisfy $B(\xi - \xi_i) = 0$:

$$\min_{\Delta\xi} \ \frac{1}{2}\|\Delta\xi\|^2_A \tag{4.17}$$
$$s.t. \ \ B\left(\left(\xi_i - \frac{1}{\eta}A^{-1}\nabla\mathcal{U}[\xi_i] + \Delta\xi\right) - \xi_i\right) = 0$$

Find the smallest $\Delta\xi$ to add to the unconstrained trajectory in order to bring the resulting trajectory onto the zero set $B(\xi - \xi_i) = 0$.

   Therefore, the second term projects the unconstrained step onto the zero set of $B(\xi - \xi_i)$. If $b \neq 0$, the trajectory is still not on the approximation to the constraint surface, and the third step makes this correction.

3. The first two steps lead to a trajectory on $B(\xi - \xi_i) = 0$, at an **offset** from the hyperplane that approximates the feasible region, $B(\xi - \xi_i) + b = 0$. Even if the Euclidean gradient $\nabla\mathcal{U}[\xi_i]$ is 0 and the previous two terms had no effect, the trajectory ξ_i might have been infeasible, leading to $b \neq 0$. The third term subtracts this offset, resulting in a trajectory that lies on the approximate constraint surface. It is the solution to the problem that minimizes the adjustment to ξ_i (again, w.r.t. the norm A) such that the trajectory

[8] When ξ_i is feasible, $b = 0$ and the two are identical, intersecting the constraint surface at ξ_i.

gets back onto the target hyperplane:

$$\min_{\Delta\xi} \ \frac{1}{2}\|\Delta\xi\|_A^2 \qquad (4.18)$$
$$s.t. \ \ B\left(\ (\xi_i + \Delta\xi) - \xi_i \ \right) + b = 0$$

Find the smallest $\Delta\xi$ to add to ξ_i in order to correct its offset from the constraint manifold $B(\xi - \xi_i) + b = 0$.

As Fig. 4.9 shows, adding the third term to the result of the previous two steps[9] brings the trajectory onto the approximation of the constraint surface.

IN SUMMARY, the algorithm can be thought of as first taking an unconstrained step in the direction dictated solely by the cost function, and then projecting it onto its guess of the feasible region in two steps, the last of which aims at correcting previous errors. For the special case of endpoint constraints, which the next section addresses, the projection further simplifies to a purely Euclidean operator, which is then smoothed through the matrix A.

4.2.3 *Goal Set Constraints*

Goal sets are a special instance of trajectory-wide constraints. Goal sets are omnipresent in manipulation: picking up objects, placing them on counters or in bins, handing them off — all of these tasks encompass continuous sets of goals.

Sampling-based planners do exist that can plan to a goal set [23]. However, the optimizer described thus far plans to a single goal configuration rather than a goal set. This single goal assumption limits its capabilities: goal sets enlarge the space of candidate trajectories, and, as Section 4.2.4 will show, enable the optimizer to converge to better solutions.

We use constrained optimization to enable the optimizer to take advantage of the additional flexibility induced by the existence of goal sets. This is one of two ways we discuss for alleviating convergence to high-cost local optima.

In order to exploit goal sets, the trajectory endpoint, which is a constant in the original optimizer, becomes a variable. That is, we use trajectory functions ξ defined on $(0, 1]$ as opposed to $(0, 1)$. This leads to a small change in the finite differencing matrix K from Section 4.1[10].

The goal set variant thus becomes a version of the constrained optimizer from Eq. 4.13, in which the trajectories satisfying $\mathcal{H}[\xi] = 0$ are the ones that end on the goal set.

Constraints that affect only the goal are a special case of trajectory constraints, for which $\mathcal{H}[\xi] = \mathcal{H}_1(\xi(1))$ (the constraint is a function of only the final configuration of the trajectory). Therefore, a large portion of the update rule will focus on the last configuration. Since $B = [0, \ldots, 0, \widetilde{B}]$, in this case B only affects the last block-row of A^{-1}, which we denote by A_1. Also note that the last $d \times d$ block in A^{-1}, is in fact of the form βI_d, since there are no cross-coupling terms between the joints.

---

[9] The result is ξ_i when the unconstrained step is zero, and it lies somewhere else along $B(\xi - \xi_i) = 0$ otherwise.

[10] K now has an additional column at the end because there is an additional point in the trajectory. This column has a 1 as its last entry, contributing to the last finite difference.

Therefore, the update rule becomes:

$$\xi_{i+1} = \xi_i - \frac{1}{\eta}A^{-1}\nabla\mathcal{U}[\xi_i] + \frac{1}{\eta\beta}A_1^T\widetilde{B}^T(\widetilde{B}\widetilde{B}^T)^{-1}\widetilde{B}A_1\nabla\mathcal{U}[\xi_i] + \frac{1}{\beta}A_1^T\widetilde{B}^T(\widetilde{B}\widetilde{B}^T)^{-1}b \tag{4.19}$$

Although not the simplest version of this update rule, this form lends itself to an intuitive geometrical interpretation. As depicted in Fig. 4.10, the update follows the "take an unconstrained step and project it" rule, only this time the projection is much simpler: it is a configuration-space projection with respect to the *Euclidean norm*, rather than a trajectory-space projection with respect to the *Hilbert norm* A.

The same projection from Fig. 4.9 now applies only to the end-configuration of the trajectory. To see this, note that $\frac{1}{\eta}A_1\nabla\mathcal{U}$ is a term that simply retrives the unconstrained step for the end configuration from $\frac{1}{\eta}A^{-1}\nabla\mathcal{U}$. Then, $\widetilde{B}^T(\widetilde{B}\widetilde{B}^T)^{-1}\widetilde{B}$ projects it onto the row space of $\widetilde{B}$. This correction is then propagated to the rest of the trajectory, as illustrated by Fig. 4.10, through $\frac{1}{\beta}A_1^T$.

The entries of A_1, on each dimension, interpolate linearly from 0 to β. Therefore, $\frac{1}{\beta}A_1^T$ linearly interpolates from a zero change at the start configuration to the correction at the end point. Since A_1^T multiplies the last configuration by β, $\frac{1}{\beta}$ scales everything down such that the endpoint projection applies exactly.

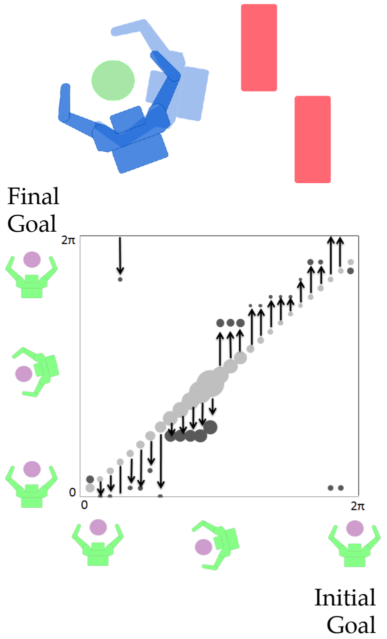


Figure 4.11: Changing the goal decreases cost. The goal set algorithm modifies the trajectory's goal in order to reduce its final cost. The figure plots the initial vs. the final goals obtained by the single goal and the goal set algorithm on a grasping in clutter problem. The area of each bubble is proportional to the cost of the final trajectory.

In summary, we showed that the projection onto a linearized version of the goal set constraint simplifies to a two step procedure. We first project the final configuration of the trajectory onto the linearized goal set constraint with respect to the Euclidean metric in the configuration space, which gives us a desired perturbation Δq of that final configuration. We then smooth that desired perturbation linearly back across the trajectory so that each configuration along the trajectory is perturbed by a fraction of Δq.

4.2.4 *Goal Set Experiments*

So far we derived the optimization algorithm under trajectory-wide constraints, and analyzed the particular case of constraints that affect only the endpoint of the trajectory. This type of constraint enables relaxing the constant goal assumption made in Section 4.1 and allows the optimizer the flexibility of a set of goal configurations. In this section, we test this with simulation experiments.

Experimental Setup. We design an experiment to test the following hypothesis:

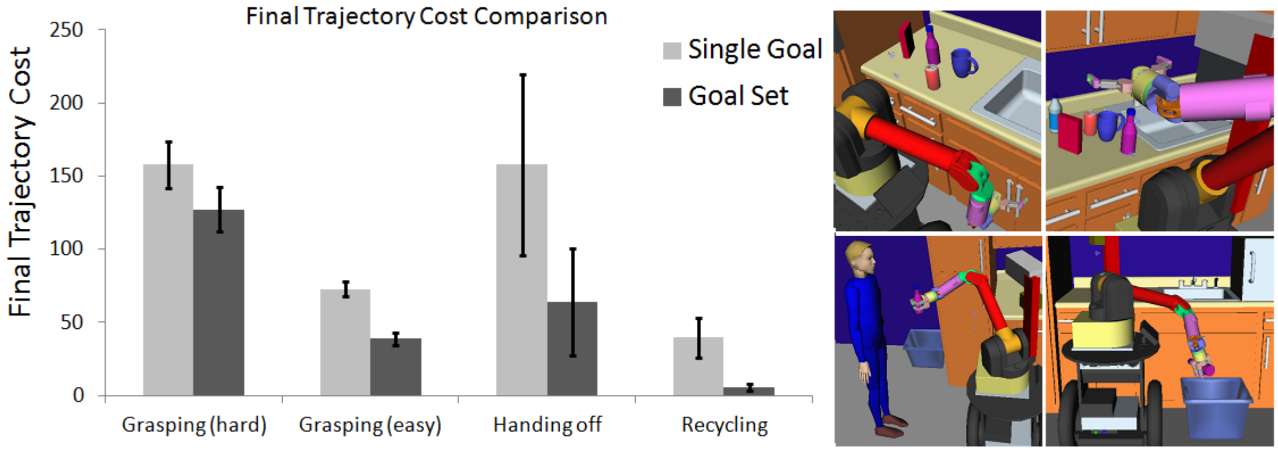


Figure 4.12: A cost comparison of the single goal with the goal set variant of CHOMP on problems from four different environment types: grasping in clutter from a difficult, and from an easy starting configuration, handing off an object, and placing it in the recycle bin.

Hypothesis: *Taking advantage of the goal set describing manipulation tasks during optimization results in final trajectories with significantly lower cost.*

We focus on day-to-day manipulation tasks, and define four types of tasks: grasping in cluttered scenes with both an easy and a difficult starting pose of the arm, handing off an object, or placing it in the recycle bin — see Fig. 4.12. We set up various scenarios that represent different obstacle configurations and, in the case of hand-offs and recycling, different initial poses of the robot. Each scenario is associated with a continuous goal set. e.g., the circle of grasps around a bottle, or the rectangular prism in workspace that ensures the object will fall into the bin.

We use a 7-DOF Barrett WAM mounted atop of a Segway base for most of this section of our experiments. To ensure a fair comparison, we use the same parameter set for both algorithms.

We compare the algorithms starting from straight line trajectories to each goal in a discretized version of this set. This reduces the variance and avoids biasing the comparison towards one algorithm or the other, by selecting a particularly good goal or a particularly bad one. For each scenario and initial goal, we measure the cost of the final trajectory produced by each algorithm.

Results and Analysis. We ran CHOMP and Goal Set CHOMP for various scenarios and goals, leading to approximately 1300 runs of each algorithm. Fig. 4.12 shows the results on each task type: the Goal Set Algorithm does achieve lower costs.

We used a two-tailed paired t-test on each task to compare the performances of the two algorithms, and found significant differences in three out of the four: on all task but grasping in clutter from a hard starting configuration, taking advantage of goal sets led to significantly better trajectories ($p < 0.05$). Across all tasks, we found a 43% improvement in cost in favor of Goal Set CHOMP, and the difference

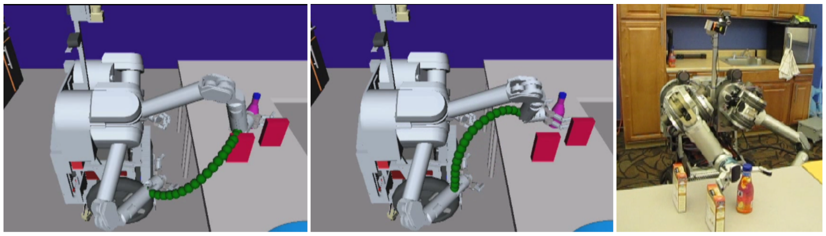


Figure 4.14: The end effector trajectory before and after optimization with Goal Set CHOMP. The initial (straight line in configuration space) trajectory ends at a feasible goal configuration, but collides with the clutter along the way. The final trajectory avoids the clutter by reaching from a different direction.

was indeed significant ($p < 0.001$), confirming our hypothesis.

We did find scenarios in which the average performance of Goal Set CHOMP was in fact worse than that of CHOMP. This can be theoretically explained by the fact that both these algorithms are local methods, and the goal set one could make a locally optimal decision which converges to a shallower local minima. At the same time, we do expect that the average performance improves by allowing goal sets. A further analysis of these scenarios suggested a different explanation: although on most cases the goal set version was better, there were a few runs when it did not converge to a "goal-feasible" trajectory (and therefore reported a very high cost of the last feasible trajectory, which was close to the initial one). We noticed that this is mainly related to the projection being impeded by joint limits. Formalizing joint limits as trajectory constraints and projecting onto both constraint sets at the same time would help mediate this problem.

Fig. 4.11 shows one of the successful scenarios. Here, the target object is rotationally symmetric and can be grasped from any direction. The figure depicts how Goal Set CHOMP changed the grasp direction and obtained lower costs (as indicated by the size of the bubbles).

The next figure, Fig. 4.14, shows a similar setup for a different mobile manipulator. Although the initial trajectory ends at a collision-free goal, it intersects with the clutter. Goal Set CHOMP converges to a trajectory ending at a goal in free space, which is much easier to reach.

IN SUMMARY, exploiting goal sets in optimization leads to lower-cost trajectories.

LIMITATIONS. A main limitation is that the optimizer is still local, and adding goal sets does not necessarily mean every situation will be improved. In fact, there are scenarios in which not allowing the end point to change results in a better trajectory. Furthermore, joint limit constraints can interfere with the projection onto the goal manifold.

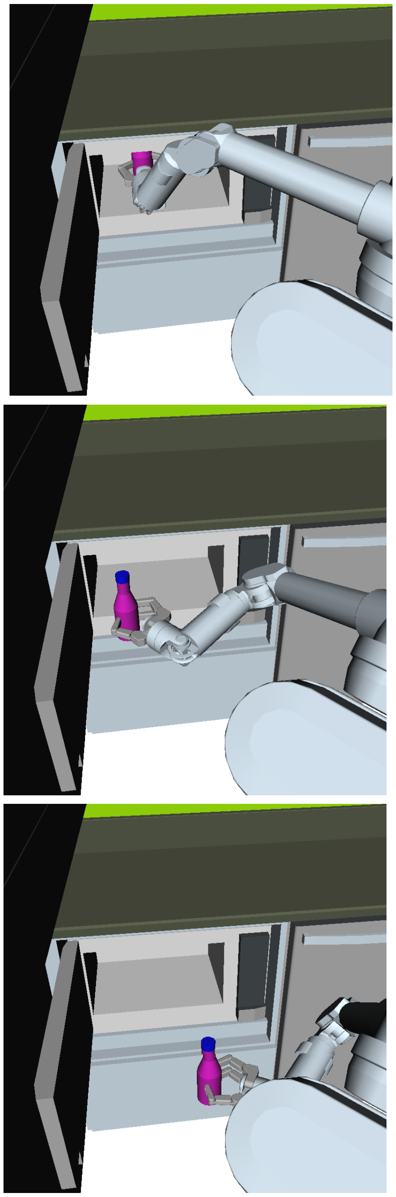


Figure 4.13: The trajectory obtained by CHOMP for extracting the bottle from the microwave while keeping it upright (a trajectory-wide constraint).

4.2.5 Trajectory-Wide Constraints Implementation

Our experience with trajectory-wide constraints on CHOMP has been mixed. CHOMP does successfully find collision-free trajectories that abide by such constraints, as the theory shows. For example, we solved the task of bimanually moving a box by enforcing a fixed relative transform between the robot's hands, and the task of keeping an object upright while extracting it from the microwave (Fig. 4.13). However, this is computationally expensive when the constraint affects all points along the trajectory. Every iteration requires the inversion of a new matrix $BA^{-1}B^T$, an $O((nd)^{2.376})$ operation (where n is the number of trajectory points and d is the dimensionality of the constraint at each point). For example, for the task in Fig. 4.13, d is 2 and CHOMP solves the problem in 17.02 seconds.

Furthermore, handling joint limits separately, as CHOMP usually does, can sometimes oppose the constraint projection: joint limits need to also be handled as hard constraints, and the unconstrained step needs to be projected on both constraints at once.

4.3 Learning from Experience

Constrained optimization enables taking advantage of goal sets, and the previous section showed that goal sets can improve optimization by giving the optimizer additional flexibility. However, optimization is still local, and can still converge to high-cost minima. In this section, we discuss a complementary approach: rather than improving the optimizer itself, we will learn to initialize it in a good basin of attraction. The optimizer will converge to a local minimum, but the initialization will aim to ensure that it will be a *low-cost* minimum.

As a complementary approach to widening good basins of attraction, we also focus on learning to initialize the optimizer in a good basin in the first place: convergence to local optima is not bad, as long as it is low-cost optima.

So how does the robot acquire a trajectory-generating oracle? In designing the oracle, we take advantage of three key features: the optimization process itself, the repetition in the tasks, and the structure in the scenes.

The optimization process relieves us from the need to produce low-cost initial trajectories. *The cost of the trajectory is irrelevant, as long as it lies in the basin of attraction of a low-cost trajectory*. Repetition or similarity in tasks allows the oracle to *learn from previous experience* how to produce trajectories. Finally, structure in the scenes suggests that we can use *qualitative attributes* to describe trajectories. For example, in a kitchen, we could say "go left of the microwave and grasp the object from the right." These attributes provide a far more compact representation of trajectories than a sequence of configurations.

This work combines all three features and proposes a learning algorithm that, given a new situation, can generate trajectories in the basin of attraction of a low-cost trajectory by predicting the values of qualitative attributes that this trajectory should posses.[11]

THE IDEA OF USING PREVIOUS EXPERIENCE to solve similar problems is not new. In Artificial Intelligence, it is known as Case-Based Reasoning [224], where the idea is to use the solution to the most similar solved problem to solve a new problem. In the MDP domain, Konidaris and Barto [132] looked at transferring the entire value function of an MDP to a new situation. Stolle and Atkeson constructed policies for an MDP by interpolating between trajectories [206], and then used local features around states to transfer state-action pairs to a new problem [207]. In motion planning, learning from experience has included reusing previous collision-free paths [29] or biasing the sampling process in randomized planners [157] based on previous environments.

Jetchev and Toussaint [108] explored trajectory prediction in deterministic and observable planning problems. They focused on predicting globally optimal trajectories: given a training dataset of situations and their globally optimal trajectories, predict the globally optimal trajectory for a new situation. Much like Case-Based Reasoning, their approach predicted an index into the training dataset of trajectories as the candidate trajectory [108, 54] or clustered the trajectories and predicted a cluster number [108, 109].[12]

Our approach differers in two key ways. First, we take advantage of the necessity of the optimization stage, and focus on the easier problem of predicting trajectories that fall in the basin of attraction of low-cost minima. Second, by predicting low-dimensional attributes instead of whole past trajectories, we are able to generate trajectories beyond the database of previous experience, allowing us to generalize further away from the training set.

RELATION TO COMPUTER VISION. Although the dataset-indexing techniques are a promising start in the field of learning from experience for trajectory optimization, they are limited: they are reminiscent of earlier works in computer vision, where one way to classify an image is to find the closest image in the training set according to some features and predict its label (or find a set of closest images and verify their predictions in post-processing).

Our work takes inspiration from attribute prediction in computer vision.

In 2006, the vision community started thinking about learning the distance metric between images [76], and this is the state at which trajectory prediction is now. In 2009 however, the object recognition community started changing this classification paradigm and shifting towards a much more general way of recognizing objects based on a simple idea: predict attributes of the object instead of the object itself,

[11] As a consequence, instead of focusing on every single voxel of a scene at once, we first make some key decisions based on previous experience, and then refine the details during the optimization.

[12] Since prediction is not perfect, a post-processing stage, where a local optimizer is initialized from the prediction is used to converge to the closest local minimum.

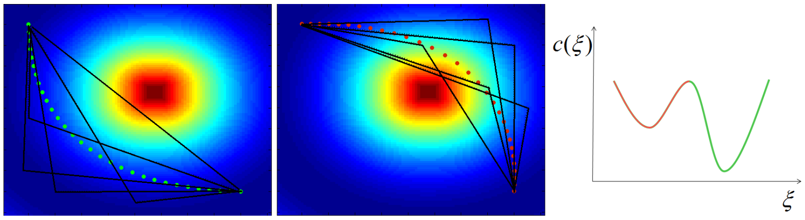


Figure 4.15: A toy example that exemplifies the idea of attributes: there are two basins of attraction, and a simple attribute (the decision of going right vs. left) discriminates between them.

and then use the attributes to predict the object [139, 71]. This not only improved recognition of known objects, but also *allowed learners to recognize objects they had never seen before.*[13]

We propose to do the same for trajectory prediction: rather than predicting trajectories directly, we predict qualitative attributes of the trajectories first, such as where their goal point is or which side of an obstacle they choose, and then map these qualitative attributes into initial guesses for a local optimizer.

4.3.1 Trajectory Attribute Prediction

Previous work [108, 109] proposed solving this problem by learning to index into a dataset of examples. This approach is limited by the dataset of previously executed trajectories, much like, for example, the earlier work in object recognition was limited by labeled images it used. In our work, we combine the idea of using a lower dimensional representation of trajectories rather than the full dimensional representation with the ability to predict new trajectories that generalize to more different situations.

The term "trajectory prediction" refers to the problem of mapping situations S (task descriptions) to a set of trajectories Ξ that solve them:

$$\tau : S \to \Xi \tag{4.20}$$

Our approach to solving the problem takes advantage of the capabilities of the optimizer. Since this optimizer is local, it will not produce the globally optimal trajectory independent of initialization, but it can produce various local minima with different costs. The training data set therefore contains not only the best trajectory found for the scene, but it can also include various other local optima. We also emphasize that trajectory prediction serves as an initialization stage for the optimizer, which leads to the following crucial observation: *In order to predict the optimal trajectory, we can predict any trajectory in its basin of attraction, and let the optimizer converge.*

We propose that there often exist some lower-dimensional trajectory attributes such that predicting these attribute values, rather than a full-dimensional trajectory, places the optimizer in the desired basin of attraction. The insight is that in producing a trajectory, a planner is faced with a few key decisions that define the topology of the trajectory. Once the right decisions are made, producing a good trajectory comes down to local optimization from any initialization that satisfies those decisions. This implies that we can reduce the problem of predicting a good trajectory to that of predicting these core attributes, and then mapping these core attributes to a trajectory.

Insight: It is enough to learn to predict low-dimensional attributes of a trajectory, which then place the optimizer in a good basin of attraction.

---

[13] A similar technique was used in [170] to recognize from brain scans words that a subject was thinking, by using physical attributes of the words as an intermediate representation.

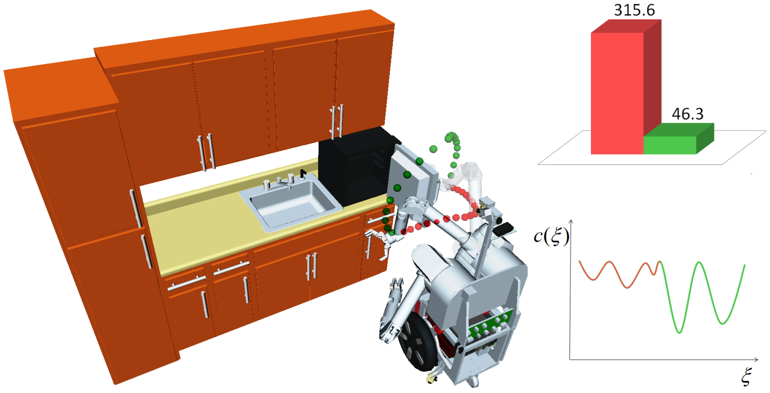


Figure 4.16: High-dimensional problems are described by many basins of attraction, but there are often attributes of the trajectory that can discriminate between low cost basins and high cost basins. In this case, such an attribute is around vs. above the fridge door.

We will discuss each of these two subproblems in turn.

To explain the idea of attribute prediction, we start with the toy world from Figure 4.15: a point robot needs to get from a start to a goal while minimizing cost . If we run CHOMP in this world, we get two solutions depending on the initial trajectory: a low and a high cost one. In order to converge to the low-cost trajectory, we can start with any trajectory to the right of the obstacle. Predicting the optimal trajectory reduces to predicting a single bit of information: right vs. left of the obstacle.

In Fig. 4.15, it is enough to predict whether to go left or right of the obstacle.

In higher dimensional problems, there are many basins of attractions and instead of globally optimal trajectories we can talk about good local minima vs. high-cost and sometimes infeasible local minima. In this setting, it is often the case that the lower-cost basins are still described by simple decisions (i.e., low-dimensional, even discrete, trajectory attributes). Figure 4.16 shows an example where going above an obstacle vs. around it will determine whether the optimizer converges to a low cost trajectory vs. a high cost one. In this case, a single bit of information will place the optimizer in a good basin of attraction. An optimizer like CHOMP can be initialized with a simple trajectory that satisfies this property, such as the one in Figure 4.17, and, as exemplified in the same figure, will bend it out of collision to a low-cost trajectory.

In Fig. 4.16, it is enough to predict whether to go above or around the fridge door.

We propose changing the trajectory prediction paradigm based on this observation, to a trajectory attributes prediction problem where we first predict key attributes that a good trajectory should have:

$$\tau : S \to \mathcal{A}(\Xi, S) \tag{4.21}$$

$\mathcal{A}(\Xi, S)$ denotes the trajectory attributes, which are conditioned on the situation, e.g., “in front of the shelf” or “elbow up around the cabinet”. These attributes implicitly define a subset of trajectories $\Xi_{\mathcal{A}} \subseteq \Xi$, and as a second step the optimizer is initialized from any trajectory $\xi \in \Xi_{\mathcal{A}}$.

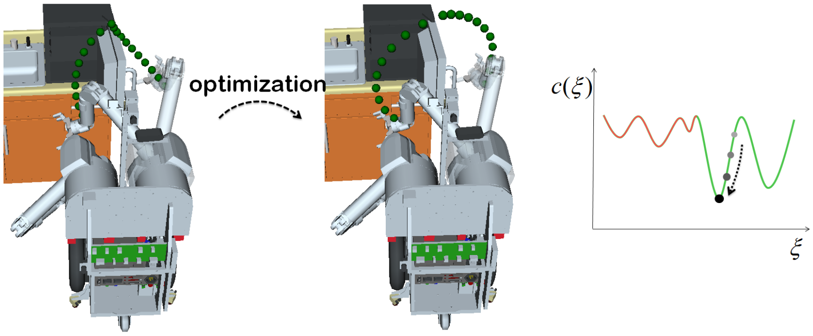


Figure 4.17: Once the right choice is made (above the fridge door), we can easily create a trajectory that satisfies it. This trajectory can have high cost, but it will be in the basin of attraction of a low-cost solution, and running a local optimizer (e.g., CHOMP) from it produces a successful trajectory.

The overall framework is

$$S \to \mathcal{A}(\Xi, S) \to \xi \in \Xi_{\mathcal{A}} \to \xi^*$$

with ξ^* the locally optimal trajectory in the basin of attraction of ξ.

Constructing a trajectory from a set of attributes ($\mathcal{A}(\Xi, S) \to \xi \in \Xi_{\mathcal{A}}$) can be cast as solving a simple constrained optimization problem: starting from a straight line trajectory, we want to keep it short while satisfying certain constraints on a few of its way-points. Since this problem is convex, generating a trajectory from attributes is very fast. As an example of such a problem, "above X and then to the left of Y" translates into two constraints on two way-points of a piecewise linear trajectory. The example from Figure 4.17 is an instantiation of that, with one constraint on one mid-point, above the fridge door, which generates two straight line segments in configuration space. Similarly, a goal attribute will be a constraint on the final end-point of the trajectory.

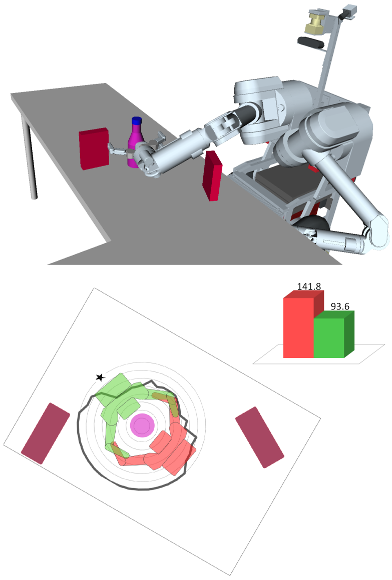


Figure 4.18: Top: the robot in one of the goal configurations for grasping the bottle. Bottom: for the same scene, the black contour is a polar coordinate plot of the final cost of the trajectory that the optimizer converges to as a function of the goal it starts at; goals that make it hard to reach the object are associated with higher cost; the bar graph shows the difference in cost between the best goal (shown in green and marked with *) and the worst goal (shown in red).

4.3.2 *Goals as Attributes*

Even thought the constrained optimizer from the previous section can take advantage of goal sets, it is still local. The initial goal choice (the goal the initial trajectory ends at) still has a high impact on the final cost of the trajectories.

Figure 4.18 plots this final cost for a variety of initial choices, in the problem of reaching for a target object in a small amount of clutter. Because of the large difference illustrated in the figure, the choice of a goal is a crucial component in the optimizer's initialization process. Here, we discuss several methods for taking advantage of previous experience[14].

FEATURES. To enable learning, we designed features that capture potential factors in deciding how good a goal is. These are indicators of how much free space there is around the goal and how hard it is

[14] Previous experience means data from previous goal initializations in different situations along with the cost of the resulting optimized trajectory.

to reach it. A subset of these features are depicted in Figures 4.19, 4.20, 4.21, and 4.22. We constructed these indicators with simplicity in mind, as a test of what can be done with very little input. We do however believe that much higher performance is achievable with a larger set of features, followed perhaps by a feature selection approach. We are also excited about the possibility of producing such features from a much rawer set using feature learning, although important questions, such as informing the algorithm about the kinematics of the robot, are still to be teased out.

We use a minimalist set of features:

- The distance in configuration space from the starting point to the goal: $||\xi(1) - \xi(0)||$. Shorter trajectories tend to have lower costs, so minimizing this distance can be relevant to the prediction.
- The obstacle cost of the goal configuration: the sum of obstacle costs for all body points on the robot, $\int c(x(\xi(1), b))db$, with c the obstacle cost in the workspace and x the forward kinematics function.
- The obstacle cost of the straight-line trajectory from the start to the goal $\bar{\xi}$: $\int\int c(x(\bar{\xi}(t), b))dbdt$. If the straight line trajectory goes through the middle of obstacles, it can potentially be harder to reach a collision-free solution.
- The goal radius: a measure of the free space around the goal in terms of how many goals around it have collision-free inverse kinematics solutions. For example, the goal set of grasping a bottle can be expressed as a Workspace Goal Region [22] with a main direction of freedom in the yaw of the end effector (this allows grasping the bottle from any angle, as in Figure 4.18). In this case, the feature would compute how many goals to the left and to the right of the current one have collision-free inverse kinematics solutions, and select the minimum of those numbers as the goal radius. The closer the clutter will be to the goal, the smaller this radius will be. It has the ability to capture the clutter at larger distances than the second feature can.
- The elbow room: the maximum radius of a collision-free sphere located at the elbow, indicating how much free space the elbow has around it for that particular goal configuration. Configurations that restrict the motion of the elbow are potentially harder to reach.
- The target collision amount: the percent of the last m configurations of the initial trajectory that are colliding with the target object. This feature is another factor in how easy it is to reach the

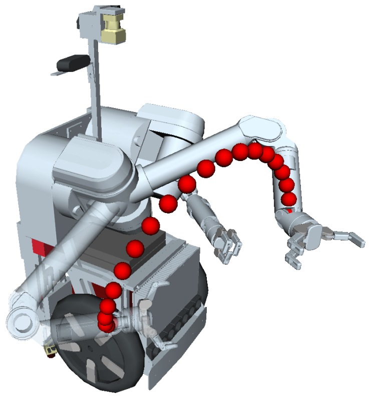


Figure 4.19: Feature 1: the length of the straight line trajectory.

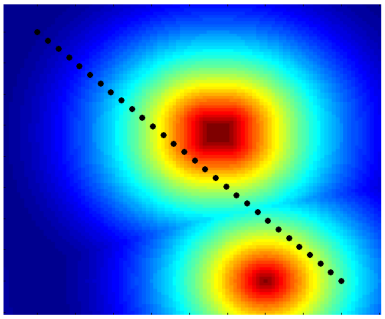


Figure 4.20: Features 2 and 3: the obstacle cost of the goal and of the straight line trajectory.

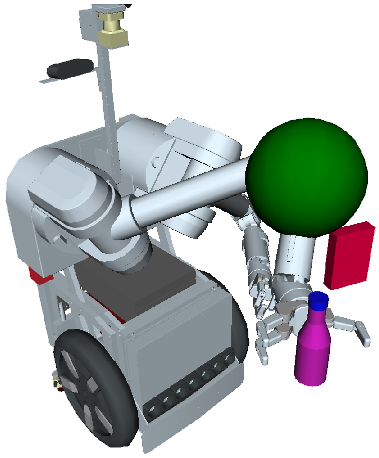


Figure 4.21: Feature 5: the free space radius around the elbow.

goal — if the initial trajectory passes through the target object, bending it out of collision could be too difficult.

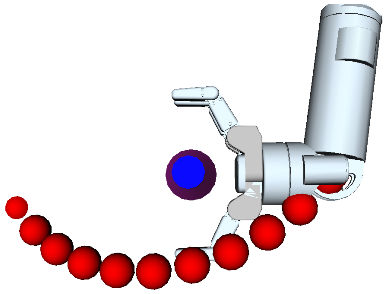


Figure 4.22: Feature 6: collision with the target object.

Domain Adaptation. Among the features, the distance from the start as well as the initial trajectory cost can differ substantially between different scenes, and so may cause difficulty for generalization. A classical approach to deal with this problem is standardization, which we can not do directly because of the large difference between our training and test set statistics. The test set contains some scenes that are considerably harder, and some that are far easier than any in the training set: training data will never capture the entire diversity of the situations the robot will face.

We still need to generalize to these situations, so we normalize the distance and cost features in a situation — this makes all situations have the same range of costs, allowing the learner to distinguish among them. We then add in the mean values of these two features, to give the learner access to how difficult the scene is, and only then standardize.[15]

Learners [Classification].

a) The Vanilla Version: The easiest way to approach the problem of deciding which goal is optimal is to directly predict if a goal will be optimal or not. For every situation, we assign the goal corresponding to the minimum final cost the value 1, and 0 to all the other goals.

We can now train a standard classifier, such as a Support Vector Machine, to predict optimality of a goal. In a new scene, given a set of goal configurations, this classifier will select any number of goals to be optimal, and we will select a random one of these as the initial guess for the optimizer. If the classifier predicts that none of these goals are optimal, then we select randomly among all goals, i.e., the classifier has not given the optimizer any information.

b) The Data-Efficient Version: Since we have access to costs and not just to the binary decision of "is optimal", another approach is to allow the classifier to predict any goal within a certain percent of the minimum cost. This can help by softening the data for the classifier, but there is of course a trade-off with predicting higher cost goals.[16]

Learners [Inverse Optimal Control].

a) The Vanilla Version: A different way to look at the problem is to treat the best goals as expert demonstrations. In Inverse Optimal Control, we want to create a cost function that explains why the experts are optimal — in our case, we want a cost function c_{IOC} in feature space such that the best goal does have the best cost in every situation. Once we have this function, we can apply it to the goals

[15] More sophisticated domain adaptation strategies (e.g., [28]) are an area of future work.

[16] We determined the value for this trade-off (the percent cutoff) on a validation set.

in a new scene and choose the goal $g^* = \arg\min_g c_{IOC}(f_g)$ (here f_g denotes the features associated with goal g).

Taking the Maximum Margin Planning approach [17], we want to find a cost function $c_{IOC} = w^T f$ that makes the optimal goal have the lowest cost by some margin. To improve generalization, we will require a larger margin for goals that are farther away from the expert: in particular, we define $l(g, g')$ to be the *structured margin*, which is zero when $g = g'$ and large when g and g' are far apart. Then saying that some goal g is optimal means $w^T f_g \leq w^T f_{g'} \quad \forall g'$. Adding in our structured margin, penalizing constraint violations with a slack variable, and regularizing w, we have:

$$\min_w \sum_s \left(w^T f_{g^s_{exp}} - \min_i \left(w^T f_{g^s_i} - l(g^s_i, g^s_{exp})\right) \right) + \frac{\lambda}{2}||w||^2 \qquad (4.22)$$

where g^s_i denotes goal i in situation s, and $l(g, g') = ||f_g - f'_g||^2$ is the structured margin which penalizes solutions from being far away in *feature space* from the expert in situation s. Overall, w pays a penalty for allowing non-expert goals to have low costs.

Taking the subgradient of (4.22) yields the following update rule:

$$w \leftarrow w - \alpha \left(\sum_s \left(f_{g^s_{exp}} - f_{g^{s*}} \right) + \lambda w \right) \qquad (4.23)$$

$$\text{where} \quad g^{s*} = \arg\min_{g_i} \left(w^T f_{g^s_i} - l(g^s_i, g^s_{exp})\right) \qquad (4.24)$$

This algorithm is targeted at identifying the minimum cost goal (4.24), ignoring the costs associated with all other goals. It gains efficiency as it does not waste resources trying to explain what happens with other goals. Our experiments will test whether this focus on the expert pays off (Section 4.3.3).

b) The Data-Efficient Version: With IOC, there exists a way of introducing the true cost information (which we do have, unlike typical IOC problems which are only given expert examples), without losing the focus on the expert. By changing the margin l_s to be the true cost difference between the goal and the expert goal rather than the distance in features, $l(g^s_i, g^s_{exp}) = U(\xi^{final}_{g_{exp}}) - U(\xi^{final}_{g_i})$, the algorithm will ensure that the minimum with respect to its new cost is close in true cost to the expert, i.e., has low cost.

LEARNERS [REGRESSION].

a) The Vanilla Version A third way to predict the minimum cost goals is to predict the final cost associated to each of the goals:

$$f^s_{g_i} \to U(\xi^{final}_{g_i})$$

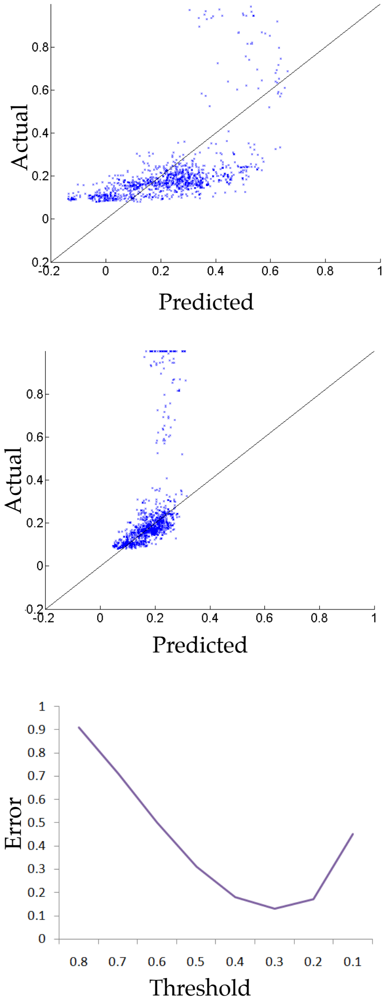


Figure 4.23: From left to right: the actual vs. predicted cost without thresholding, the actual vs. predicted cost with thresholding, and the dependence of the fit error of a validation set of medium and low cost examples on the threshold (on the left of the minimum, the regressors pays too much attention to high costs, on the right it uses too little data.

---

[17] N. Ratliff, J. A. Bagnell, and M. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning (ICML)*, 2006

with $\xi_{g_i}^{final}$ the final trajectory obtained by initializing the optimizer with the straight line to goal g_i, and choose the best one:

$$g^* = \arg\min_{g_i} U(\xi_{g_i}^{final})$$

This is sometimes referred to as arg min-regression. We looked at three different regressors:

- Linear Regression: $w = F^{\dagger}C$, with F a matrix concatenating every feature vector on every situation, one per row, and C a vector concatenating all the final costs obtained by the goal set variant of our optimizer, one per row.
- Gaussian Process: A wide Gaussian radial basis kernel performed the best, since we need far knowledge transfers.
- Neural Network: We used the Back-Propagation Neural Network with one hidden layer. We determined the number of nodes in this layer, as well as the weight decay coefficient based on performance on a validation set.

b) The Data-Efficient Version: Looking at the initial performance of Linear Regression on the training set (Figure 4.23, left), it becomes apparent that there are a lot of data points with very high cost, and predicting that cost accurately is not only unnecessary, but leads to not being able to identify the good solutions from the mediocre ones. This suggests that even these regressors should not use all the data, but rather focus their efforts on discriminating among the lower-cost solutions by truncating the cost at some threshold.

We selected this threshold based on a validation set as shown in Figure 4.23 (right). The plot shows that a very low threshold degrades performance by confusing the learner to pay attention to the high-cost outliers, and a very high threshold also degrades performance by starving the learner of data. Figure 4.23 (center) portrays the new predictions based on the learned threshold, forming a much better fit for the solutions we are interested in, while keeping the high-cost predictions sufficiently high.[18]

4.3.3 *Learning from Experience Experiments*

We conducted three experiments: two that analyze how well we can generalize to new situations, and another that presents a more realistic evaluation of the day-to-day performance.

GENERALIZATION FROM LIMITED DATA. In a first experiment, we wanted to test how well we can generalize to new situations, going beyond the exemplars already executed. We used only two

[18] We also tried to do the thresholding per scene instead of on the entire training data, but this did not cause a significant improvement, because the effect on how well the regressors can fit the data is minimal.

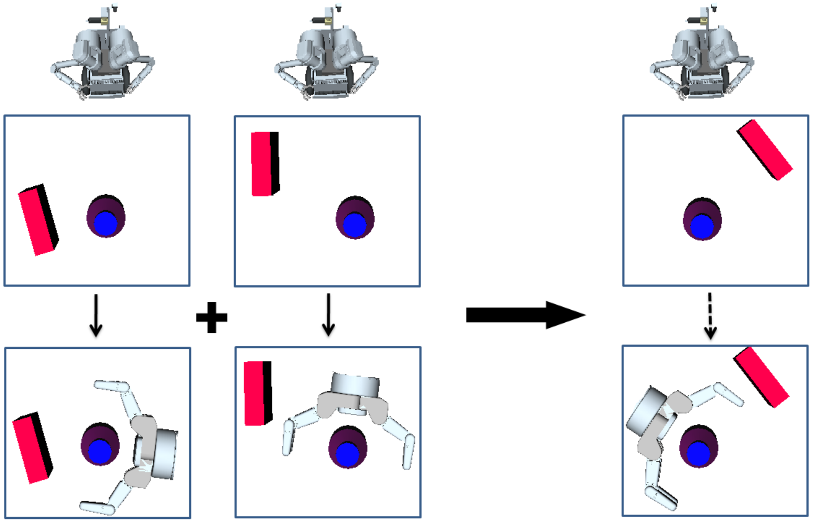


Figure 4.24: Two training situations along with their corresponding best goal, and a test situation in which the correct goal is predicted. If the learner were constrained to the set of previously executed trajectories, it would not have been able to generalize to this new scene.

scenes for training, shown in Figure 4.24, where the goal was the grasp the bottle while avoiding the table holding the object, as well as the box placed next to the target. We ran CHOMP to each goal in a discretization of the goal set, and recorded the final cost. Figure 4.24 shows the goals that produced the best cost for each of the scenes. We then trained a neural network to predict this cost given only the first three features.

Predicting attribute values instead of full trajectories enables the learner to predict options that go beyond its library of previous experience. Even if grasping an object from the left was never the optimal strategy, the learner can predict it on a test problem by evaluating features of this attribute value for the new environment.

For testing, we moved the object to a very different location than in the training examples, also shown in Figure 4.24. With a Nearest-Neighbor approach, the robot would identify one of the training scenes as closest, and initialize the optimizer from the best final trajectory for that scene. In this case, all the trajectories go to a goal that is sub-optimal or even colliding with the environment. The trajectory attributes approach, however, allows us to go beyond these previously executed trajectories. The learner predicts that goal shown on the right of Figure 4.24 will produce the best cost. This goal has never been optimal in the training examples, yet because it stays away from clutter while maintaining a short distance from the starting configuration, the learner will recognize it as better than the other choices. Indeed, when initializing the optimizer from the straight line trajectory to that goal, the final cost is only 1% higher than the best path we were able to find using multiple initializations of the optimizer to the different goals.

GENERALIZATION DEPENDENCE ON TRAIN-TEST SIMILARITY In this next experiment, we were interested in testing how far away from the training data we can transfer knowledge to. We created one testing situation, and trained two of the regressors (the Neural

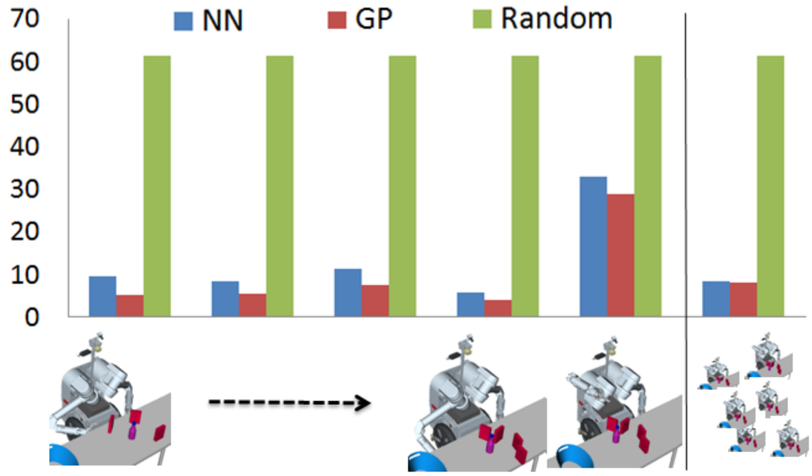


Figure 4.25: The loss over the minimum cost on the same test set when training on scenes that are more and more different, until everything changes drastically in the scene and performance drops significantly. However, the loss decreases back to around 8% when training on a wide range of significantly different scenes, showing that the algorithm can do far transfers if given enough variety in the training data.

Network and the Gaussian Process) on situations that are more and more different from the testing one. In Figure 4.25, we plot the performance in these cases as the percent of degradation of cost over the minimum that the optimizer can reach — the final cost corresponding to initializing the optimizer with a straight line trajectory to the best goal. These performances, averaged across 15 different clutter configurations, are compared with our baseline: what happens if we randomly choose a collision-free goal, without any learning?

In the first setting, we train and test on the same dataset. Both the Neural Network and the GP perform drastically better than the no-learning baseline. We then change the situation slightly: first the clutter configurations change, then the target object position changes by approx. 20cm, followed by the starting configuration of the robot. In the last but one test, we change all these situation descriptors drastically, and the performance decreases significantly, although the learning algorithms still outperform the baseline. Finally, we show that more variety in the training set can lead to better generalization. When we increase the number of examples in the training set — we still train on very different situations, but we provide a wider range with more possible starting configurations and target poses — we notice that the performance again improves to about 8% for both regressors. The random choice baseline does of course not take into account this data and performs the same, around 62% degradation over the minimum cost.

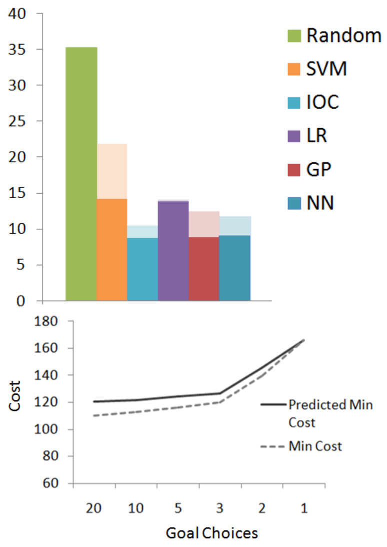


Figure 4.26: Top: Percentage loss over the best cost for all the methods. Solid bars are the data-efficient versions, and transparent bars are the vanilla algorithms, which perform worse. Bottom: The predicted minimum cost vs. the true minimum cost as function of the number of choices considered.

Main Experiment. We are also interested in a realistic evaluation of the day-to-day performance of our system, as well as establishing which learning approach is most suitable for our problem. Should the learner focus on just the optimal goal, or should it also focus on the sub-optimal goals and their performance?

We created a large set of training examples, comprising of 90 situations varying in the starting configuration, target object pose, and clutter distribution. In each situation, we ran the optimizer starting from the straight line trajectory to each of the collision-free goals in the discretized goal set (a total of 1154 examples) and recorded the final cost. We also created a test set of 108 situations (1377 examples) that differ in all three components from the training data.
Hypothesis. *Learning from experience improve the final result of the optimization.*

Figure 4.26(top) shows the percentage of cost degradation over the minimum, averaged across all testing situations, for the five learning approaches. The solid bars are the data-efficient versions of the algorithms: the regressors use thresholds established on a separate validation set, IOC uses the cost distance for the structured margin, and the classifier predicts goals close to the minimum as well.

Overall, using prior experience helps, even when using simple predictors.

In line with our hypothesis, all methods perform better than not using the experience. Furthermore, the vanilla versions of these methods, shown with transparent bars, always perform worse than their data-efficient counterparts.

The best performer is our version of data-efficient IOC — this algorithm focuses on predicting the expert rather than fitting cost, while taking into account the true cost and ensuring that non-expert predictions have low cost. Although both IOC and LR are linear, the advantage of IOC over LR is its expert prediction focus. The non-linear regressors have similar performance as IOC, and their advantage is a better fit of that data. The SVM is focusing on low-costs with a linear kernel, so its performance is, as expected, close to LR.

In these experiments, we had a fairly fine discretization of the goal set per scene. It makes sense to ask if we could get away with fewer choices. Figure 4.26(bottom) indicates that the answer is yes: with 5 goals, for example, we can predict the minimum cost better, and we this minimum is not a lot larger than the one considering, say, 20 goals.

In summary, attribute prediction can help trajectory optimization produce better solutions faster, even in cases where the attribute values describing a good basin of attraction are different from the attribute values that performed well before: the robot can more easily generalize and is not constrained by the successful trajectories that it has already encountered.

Limitations. Finding good attributes remains a challenge. Furthermore, our work can be augmented by work on contextual library optimization [19] in order to predict not one choice, but a sequence of choices that the optimizer should attempt.

4.4 *Chapter Summary*

Key to generating predictable or legible motion is the ability to generate motion that is *optimal*. Predictability and legibility instantiate optimality, for different choices of the cost functional to be optimized.

In this chapter, we started with an algorithm for trajectory optimization. The algorithm capitalizes on two observations: 1) gradients are often easy to compute, and provide useful information, and 2) non-Euclidean inner products lead to gradient steps that propagate local information globally to the trajectory. The optimizer iteratively follows the direction of the natural gradient of a cost functional combining an obstacle avoidance term with any differentiable prior (e.g., smoothness, efficiency, predictability, legibility).

Despite taking advantage of these two ideas, trajectory optimization is still challenging for robots with a high number of degrees of freedom. Obstacles induce non-convex constraints that, in turn, induce local optima. A local, gradient-based optimizer will sometime converge to high-cost local optima. On the other hand, global optimizers are not tractable.

exploit goal sets

↑

trajectory optimization

[gradient information]
[non-Euclidean inner product]

↓

learn to initialize

This chapter introduced two complementary ways to alleviate convergence to bad local optima: goal set constraints, and learning from experience.

The first was to exploit the existence of goal sets in typical manipulation tasks. We cast goal sets as an instance of trajectory-wide constraints, and showed how the derivation simplifies when the constraint only affects the end point of the trajectory. Our results suggest that exploiting goal sets does significantly improve optimization in tasks like reaching, placing, or handing off.

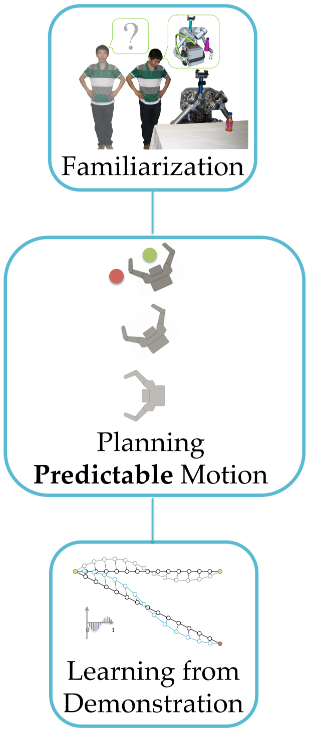
The second was to exploit the previous experience that the robot has: different motion planning problems it has attempted with different initializations. The robot learns from both failures and successes to predict, for a new problem, low dimensional attributes of the trajectory meant to place the initialization of the optimizer in a good basin of attraction. Our results show the utility of this for the choice of a goal as an attribute, but there is an opportunity for future work to explore more complex and possibly learned attributes.

Armed with an optimizer, the robot now has a necessary tool in place to generate predictable (Chapter 5) and legible (Chapter 6) motion.

[19] Debadeepta Dey, Tian Y Liu, Boris Sofman, and Drew Bagnell. Efficient optimization of control libraries. Technical report, DTIC Document, 2011

5

Generating Predictable Motion



Predictable motion matches the observer's expectation, given a known goal for the robot. In Chapter 3, we formalized predictability in terms of the cost functional C that the observer expects the robot to optimize, and introduced one example for C — the integral over squared velocities in the configuration space (Eq. 3.7).

Here, we derive the gradient for this example C (Section 5.1), which the robot can directly use in the optimizer from Chapter 4. However, as the experiment in Section 3.4 suggests, directly using this C is often not enough: for a robot like HERB, different participants had different notions of what C should be.

In the remainder of the chapter, we discuss two complementary ways of alleviating this issue: learning predictable motion from user demonstrations (Section 5.2), and familiarizing the observer though example motions to the robot's C (Section 5.3).

5.1 The Predictability Gradient

In Chapter 4, we derived an optimizer that takes natural gradient steps in the space of trajectories, with the update rule outlined in Eq. 4.12. This update rule depends on the Euclidean gradient $\nabla_{\xi_i}\mathcal{U}$:

$$\nabla_{\xi_i}\mathcal{U} = \nabla_{\xi_i}\mathcal{U}_{prior} + \alpha\nabla_{\xi_i}\mathcal{U}_{obs}$$

Eq. 4.3 shows the formula for $\nabla_{\xi_i}\mathcal{U}_{obs}$. To generate predictable motion, we need to derive $\nabla_{\xi_i}\mathcal{U}_{prior} = \nabla_{\xi_i}C$.

Our example C from Eq. 3.7 is the integral over squared velocities:

$$C[\xi] = \int F(\xi(t), \dot{\xi}(t), t)dt$$

$F(\xi(t), \dot{\xi}(t), t) = \frac{1}{2}||\dot{\xi}(t)||^2$

We can find its Euclidean gradient using Euler-Lagrange as

$$\nabla_{\xi}C(t) = \frac{\partial F}{\partial \xi(t)} - \frac{d}{dt}\frac{\partial F}{\partial \dot{\xi}(t)} = 0 - \frac{d}{dt}\dot{\xi}(t) = -\ddot{\xi}(t)$$

Therefore, our missing gradient piece is

$$\nabla_{\xi_i}\mathcal{U}_{prior} = -\ddot{\xi}_i$$

The most predictable motion is attained when C reaches its minimum, which happens when the gradient is 0:

$$-\ddot{\xi}(t) = 0 \Rightarrow \dot{\xi}(t) = k_1 \Rightarrow \xi(t) = k_1 t + k_2$$

k_1 and k_2 are determined by the end point constraints, $\xi(0) = s$ and $\xi(1) = g$.

With our example C, our observer thinks that the most predictable trajectory is the straight line to the goal, at constant velocity.

5.2 *Learning from Demonstration*

The C from the previous section is a good starting point, and can possibly be used as a common denominator across multiple users. However, our study from Section 3.4 suggests that different users have different expectations of how the same robot would move.

On the other hand, we are assuming a non-adversarial context, where the user directly benefits from the robot being more predictable. Therefore, users might be willing to *train* the robot's motion planner by giving the robot *demonstrated trajectories* for how it should move in different situations.

PROBLEM STATEMENT. Given a demonstration ξ_D (like the gray trajectory in Fig. 5.2) from a start s to a goal g (or a set of such demonstrations), the robot needs to *generalize* it to new situations. A very common type of generalization that the robot will face — the main one we focus in this section — is adaptation to new end-points: a new start $\hat{s}$ and/or a new goal $\hat{g}$.

Given a demonstration ξ_D from s to g, how should the robot adapt it to new end-point configurations $\hat{s}$ and $\hat{g}$? We discuss at the end of the section how to handle changes in the obstacles in the environment as well.

THERE ARE TWO SCHOOLS OF THOUGHT on addressing this problem: a model-based approach and a model-free approach. The former is to recover a cost function (typically as a weighted combination of features, $U[\xi] = w^T f_\xi$), which "explains" ξ_D. This is referred to as Inverse Reinforcement Learning (IRL) [2, 238, 183]. In the deterministic case, this means that ξ_D has the lowest cost by a margin [183]:

$$\text{Find } w \text{ s.t. } w^T f_{\xi_D} \leq w^T f_\xi + \zeta, \forall \xi \in \Xi_s^g$$

ζ is a slack variable

In the nosy case, it means that ξ_D has high probability (applying again the principle of maximum entropy) [238]:

$$\max_w P(\xi_D | s, g, w) = \frac{1}{Z}\exp\left(-w^T f_{\xi_D}\right)$$

Z is the normalizer

IRL imposes a model on the problem — that the demonstration can be explained by trajectory optimization as the (approximate) minimum of some cost, typically as a linear combination of features. It can transfer far from the demonstration, but it is not tractable in high-dimensional spaces because it relies on globally solving the forward problem — given a cost, find its optimum in an environment, or evaluate the expected features it will produce in that environment.

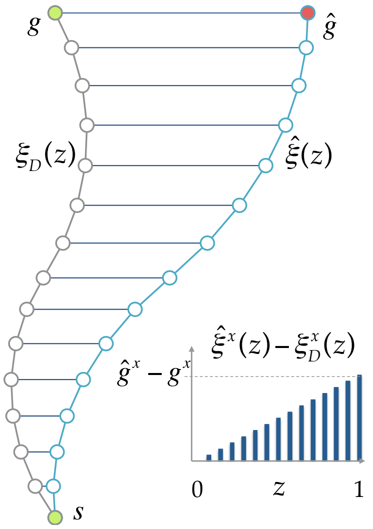


Figure 5.1: Using a norm M for adaptation propagates the change in the start and goal, from $\{s,g\}$ to $\{\hat{s}, \hat{g}\}$, to the rest of the trajectory, changing ξ_D into $\hat{\xi}$. The difference between the two as a function of time is plotted in blue.

In contrast, model-free approaches do not attempt to explain the demonstration, but use various processes to morph it to the new situation. These techniques do not necessarily transfer as far from the demonstration, but remain tractable and even efficient in high-dimensional spaces. *Due to their computational efficiency, they can be useful in generalizing predictable motion.*

Among several such methods [33, 234, 193, 13], a commonly used one is a Dynamic Movement Primitive (DMP) [105, 104]. DMPs have seen wide application across a variety of domains, including biped locomotion [166], grasping [174], placing and pouring [172], dart throwing [127], ball paddling [128], pancake flipping [133], playing pool [173], and handing over an object [179].

DMPs represent a demonstration as a dynamical system *tracking a moving target configuration*, and adapt it to new start and goal constraints by simply changing the start and goal parameters in the equation of the moving target. The adaptation process is the same, regardless of the task and of the user, and is merely one instance of a larger problem.

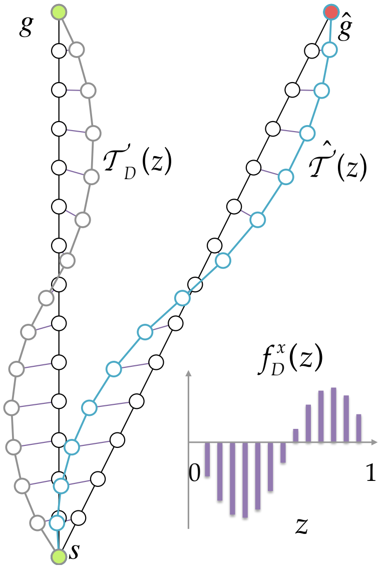


Figure 5.2: In contrast, DMPs represent the demonstration as a spring damper system tracking a moving target trajectory $\mathcal{T}_D$, compute differences f_D (purple) between $\mathcal{T}_D$ and the straight line trajectory, and apply the same differences to the new straight line trajectory between the new endpoints. This results in a new target trajectory $\hat{\mathcal{T}}$ for the dynamical system to track. When $M = A$, the velocity norm from Eq. 4.7, the two adaptations are equivalent. In general, different norms M would lead to different adaptions.

Our work focuses on the second family of methods due to their computational efficiency, and connects them to trajectory optimization. We introduce a generalization of this adaptation process — we provide a variational characterization of the problem by formalizing the adaptation of a demonstrated trajectory to new endpoints as an optimization over a Hilbert space of trajectories (Section 5.2.1). We find the *closest* trajectory to the demonstration, in the linear subspace induced by the new endpoint constraints (Fig. 5.3). Distance (the notion of "closer") is measured by the norm induced by the inner product in the space.

Using this formalism, different choices for the inner product lead to different adaptation processes. We prove that DMPs implement this optimization in the way they adapt trajectories, for a *particular* choice of a norm (Section 5.2.2). We do so by proving that when updating the endpoints, the moving target tracked by the dynamical system adapts (as in Fig. 5.2) using the very same norm A from Chapter 4, Eq. 4.7. We then show that this also implies that the adaptation in the trajectory space, obtained by then tracking the adapted target, is also the result of optimizing a norm based on A.

Beyond providing a deeper understanding of DMPs and what criteria they are inherently optimizing when adapting demonstrations, our generalization frees the robot from a fixed adaptation process by enabling it to use *any* inner product (or norm). Because computing the minimum norm adaptation is near-instant, any such adaption process can be used in the DMP to obtain the new moving target trajectory.

Thus, we can select a more appropriate norm based on the task at hand (Section 5.2.3). What is more, if the user is willing to provide a few examples of how to adapt the trajectory as well, then the robot can *learn* the desired norm (also Section 5.2.3): the robot can learn, from the user, not only the trajectory, but also *how to adapt the trajectory* to new situations.

We conduct an experimental analysis of the benefit of learning a norm both with synthetic data where we have ground truth, as well as with kinesthetic demonstrations on a robot arm. Our results show a significant improvement in how well the norm that the robot learns is able to reconstruct a holdout set of demonstrations, compared to the default DMP norm.

By learning how to adapt the demonstrated trajectories, the robot can produce trajectories for new situations that better match what the user would demonstrate, i.e., more predictable motions.

IN SUMMARY, we contribute a deeper theoretical understanding of DMPs that relates them to trajectory optimization, and also leads to practical benefits for learning from demonstration. In particular, robots can more predictable motion by adapting trajectories in a manner that more closely matches what the user would demonstrate (and therefore expect) in a new situation.

5.2.1 Hilbert Norm Minimization

In this subsection, we formalize trajectory adaptation as a Hilbert norm minimization problem. We then derive the solution to this problem, and study the case in which translating trajectories carries no penalty. This is the case for the norm DMPs use in their adaptation process.

Given a demonstrated trajectory ξ_D, we propose to adapt it to a new start $\hat{s}$ (the robot's starting configuration) and a new goal $\hat{g}$ by solving:

$$\begin{aligned}\hat{\xi} = \arg\min_{\xi\in\Xi} &\, ||\xi_D - \xi||^2_M \\ s.t.\ \xi(0) &= \hat{s} \\ \xi(1) &= \hat{g}\end{aligned} \tag{5.1}$$

M is the norm defined by the inner product in the Hilbert space of trajectories, as in Eq. 4.6.

Fig. 5.3 illustrates this problem. Different inner products lead to

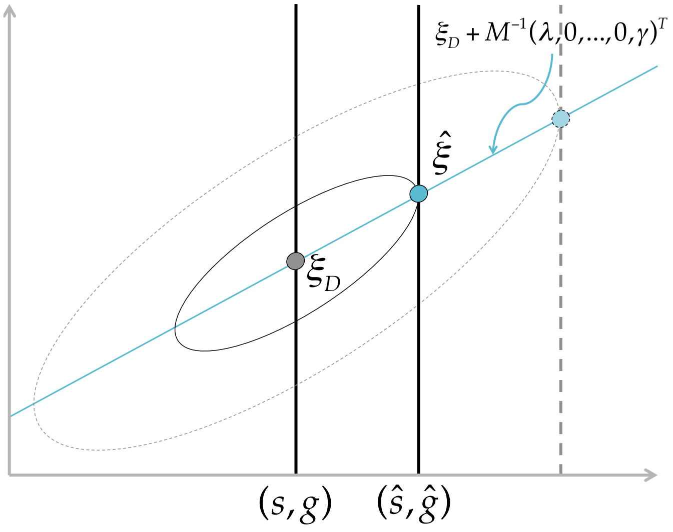


Figure 5.3: We adapt ξ_D by finding the closest trajectory to it that satisfies the new end point constraints. The x axis is the start-goal tuple, and the y axis is the rest of the trajectory. M warps the space, transforming (hyper)spheres into (hyper)ellipsoids. The space of all adaptations of ξ_D is a linear subspace of Ξ.

different Ms, which in turn lead to different adaptations.

SOLUTION. The Lagrangian of Eq. 5.1 is

$$\mathcal{L} = (\xi_D - \xi)^T M(\xi_D - \xi) + \lambda^T(\xi(0) - \hat{s}) + \gamma^T(\xi(1) - \hat{g}) \tag{5.2}$$

Taking the gradient w.r.t. ξ, λ, and γ:

$$\nabla_\xi \mathcal{L} = M(\xi_D - \xi) + (\lambda, 0, ..0)^T + (0, .., 0, \gamma)^T \tag{5.3}$$

$$\nabla_\lambda \mathcal{L} = \xi(0) - \hat{s}, \ \nabla_\gamma \mathcal{L} = \xi(1) - \hat{g} \tag{5.4}$$

Thus, the solution is:

$$\hat{\xi} = \xi_D + M^{-1}(\lambda, 0, .., 0, \gamma)^T \tag{5.5}$$

where the vectors λ and γ are set by Eq. 5.4.

This has an intuitive interpretation: *correct the start and the goal, and propagate the differences across the trajectory* in a manner dictated by the norm M (Fig. 5.2).[1] Fig. 5.3 depicts the geometry of the space.

FREE TRANSLATIONS. Often times, we are interested in being able to translate trajectories at no cost, i.e., if $\hat{\xi} = \xi + \xi_k$, with $\xi_k(t) = k, \forall t$ (a constant valued trajectory), then $||\hat{\xi} - \xi||_M = 0, \forall k$. However, that makes M a *semi-norm*, as $\langle \xi_k, \xi_k \rangle = 0, \forall k$, which makes the problem ill posed.

Our optimizer from Chapter 4 bypasses the semi-norm problem because at least one of the trajectory endpoints is constant (both in Section 4.1, only the start in Section 4.2). Similarly, the key to free

[1] It is a simplified version of the update from the goal set optimization in Section 4.2: think of ξ_D as the trajectory obtained after an unconstrained step, and of the goal manifold as comprising of just $\hat{g}$.

translations while maintaining a full norm is fixing one of the endpoints, e.g., the starting configuration: one can adapt the trajectory's goal in a restricted space of trajectories that all have the same (constant) start, and then translate the result to the new starting configuration.

Let $\Xi_{s=k}$ be the subspace of trajectories s.t. the starting configuration is a constant k: $\xi(0) = k$, $\xi \in \Xi_{s=k} \subset \Xi$. M is a full norm in $\Xi_{s=k}$, as no translations are allowed.

Let $\sigma_k : \Xi_{s=k} \rightarrow \Xi_{s=0}$, $\sigma_k(\xi) = \xi - \xi_k$ be the function that translates trajectories from $\Xi_{s=k}$ to start at $s = 0$. This function is bijective, $\sigma_k^{-1}(\xi) = \xi + \xi_k$.

We can reformulate Eq. 5.1 to finding the closest trajectory within $\Xi_{s=0}$ that ends at $\hat{g} - \hat{s}$, and translating this trajectory to the new start $\hat{s}$, thereby obtaining a trajectory from $\hat{s}$ to $\hat{g} - \hat{s} + \hat{s} = \hat{g}$:

$$\hat{\xi} = \sigma_{\hat{s}}^{-1}\left(\arg\min_{\xi \in \Xi_{s=0}} ||\sigma_s(\xi_D) - \xi||_M^2\right) \qquad s.t.\ \xi(1) = \hat{g} - \hat{s} \tag{5.6}$$

Perform the adaptation in a space with a constant starting configuration 0, then translate to the correct start.

The solution to this, following an analogous derivation to the constrained optimization problem in Eq. 5.1, is to take the demonstration translated to 0, correct the goal to $\hat{g} - \hat{s}$, propagate this change to the rest of the trajectory via M, and then translate the result to the new start:

$$\hat{\xi} = \sigma_{\hat{s}}^{-1}\left(\sigma_s(\xi_D) + M^{-1}(0, .., 0, \gamma)^T\right) \tag{5.7}$$

with γ s.t. $\hat{\xi(1)} = \hat{g}$. For a norm M with no coupling between joints, and m the last entry in M, this becomes:

$$\hat{\xi} = \sigma_{\hat{s}}^{-1}\left(\sigma_s(\xi_D) + \frac{1}{m}M^{-1}(0, .., 0, (\hat{g} - \hat{s}) - (g - s))^T\right) \tag{5.8}$$

This corrects the goal in $\Xi_{s=0}$ from $g - s$ to $\hat{g} - \hat{s}$, effectively changing the goal in Ξ from g to $\hat{g}$.[2]

5.2.2 *DMP Adaptation as a Special Case of Hilbert Norm Minimization*

In this subsection, we summarize a commonly used version of DMPs, and write it as a target tracker with a moving target. Next, we show that the adaptation of the tracked target to a new start and goal is an instance of Hilbert norm minimization (Theorem 1). Finally, we show that this induces an adaption in trajectory space that is an instance of norm minimization (Theorem 2).

DMPs. A commonly used version [172, 173, 179, 174] of a DMP is a second order linear dynamical system which is stimulated with a

[2] Note that here we are overloading M. In Eq. 5.6, we are measuring norms in a space of trajectories with constant start 0, which is a lower dimensional space of trajectories $\tilde{\xi} : (0, 1] \rightarrow \mathcal{Q}$ that do not contain the starting configuration (which is not a variable). In this space, we can define a norm $\tilde{M}$ by $||\tilde{\xi}||_{\tilde{M}} = ||\xi||_M$, with $\xi(0) = 0$ and $\xi(z) = \tilde{\xi}(z) \forall z \in (0, 1]$. $\tilde{M}$ is then of dimensionality one less than M and full rank, and what we actually use in Eq. 5.8.

non-linear forcing term:

$$\tau^2\ddot{\xi}(t) = K(g - \xi(t)) - D\tau\dot{\xi}(t) - K(g-s)u + K\bar{f}(u) \tag{5.9}$$

where $K(g - \xi(t))$ is an attractor towards the goal, $K(g-s)u$ avoids jumps at the beginning of the movement, $D\dot{\xi}(t)$ is a damper, and $K\bar{f}(u)$ is a nonlinear forcing term. u is a phase variable generated by the dynamical system

$$\tau\dot{u} = -\alpha u$$

Thus, u maps time from 1 to (almost) 0:

$$u(t) = e^{-\frac{\alpha}{\tau}t} \tag{5.10}$$

DMP Adaptation as Tracked Target Adaptation. Let $z = 1 - u$. We can reformulate a DMP as a target tracker with a moving target, $\mathcal{T}(z)$:

$$\tau^2\ddot{\xi}(t) = K(\mathcal{T}(z) - \xi(t)) - D\tau\dot{\xi}(t) \tag{5.11}$$

with $\mathcal{T}(z)$ moving from s to g as a function of z on a straight line constant speed in z plus a deviation f as a function of z, $f(z) = \bar{f}(u)$:

What was previously $\xi(t)$, here becomes $\mathcal{T}(z)$. $\xi(t)$ is now the original trajectory, as a function of time going from 0 to T. We first show that DMPs minimize a norm in the tracked target space $\mathcal{T}(z)$, and then use that to show that there is a norm being minimized in the original trajectory space $\xi(t)$.

$$\mathcal{T}(z) = s + z(g-s) + f(z) \tag{5.12}$$

Given a demonstration ξ_D, one forms a DMP by computing $f_D(z)$ from Eq. 5.11.[3] To generalize to a new $\hat{s}$ and $\hat{g}$, the target changes from Eq. 5.13 to Eq. 5.14:

$$\mathcal{T}_D(z) = s + z(g-s) + f_D(z) \tag{5.13}$$

$$\hat{\mathcal{T}}(z) = \hat{s} + z(\hat{g}-\hat{s}) + f_D(z) \tag{5.14}$$

The linear function from s to g is adapted to the new endpoints, becoming $\hat{s} + z(\hat{g} - \hat{s})$ (black trajectories in Fig. 5.2), and the deviation f_D remains fixed (purple deviations in Fig. 5.2).

Relation to Hilbert Norm Minimization. We prove the following:

> *The adaptation of the target being tracked by the DMP, from $\mathcal{T}_D$ to $\hat{\mathcal{T}}$, is a special case of the Hilbert norm adaptation from ξ_D to $\hat{\xi}$, when the norm $M = A$ from Eq. 4.7.*

To prove this, we show the equivalence between the DMP adapted trajectory $\hat{\mathcal{T}}$ and the outcome of the Hilbert norm minimization $\hat{\xi}$ from Eq. 5.8, for $\mathcal{T} = \xi$.

We do this in two steps. Since $\hat{\mathcal{T}}$ is the sum of a straight line trajectory (as a function of z) and a fixed deviation, we first show that the Eq. 5.6 will adapt a straight line trajectory to another straight line when M is the norm A. Next we show that when adding a nonzero deviation to the initial trajectory, the same deviation is added by Eq. 5.6 to the adapted trajectory.

---

[3] Typically, there is a smoothing step before adaptation where $\bar{f}_D$ is fitted by some basis functions, $\bar{f}_D(u) = \frac{\sum \psi_i(u)\theta_i u}{\sum \psi_i(u)}$. The same smoothing can be applied to a trajectory before performing Hilbert norm minimization.

Therefore, we first focus on the case when $f_D = 0$. In this case, the targets are straight lines from the start to the goal, moving at constant speed: $\mathcal{T}(z) = \xi_{straight}(z) = (g - s)z + s$, and $\hat{\mathcal{T}}(z) = \hat{\xi}_{straight}(z) = (\hat{g} - \hat{s})z + \hat{s}$.

In Lemma 3, we show that the adaptation of $\xi_{straight}$ to a new start $\hat{s}$ and a new goal $\hat{g}$ with respect to the norm A matches $\hat{\xi}_{straight}$. We build to this via two other lemmas, where the key is to represent straight lines in terms of the norm A. We first prove that $\xi_{straight}$ minimizes $\xi^T A \xi$ (Lemma 1). This enables us to write out $\xi_{straight}$ in terms of A (Lemma 2).

We then generalize this to non-zero f_D using that f_D in not actually changed by the norm M in Eq. 5.8.

Lemma 1: $\xi_{straight}$ is the solution to minimizing Eq. 4.7.

Constant speed straight line trajectories have minimum norm under A.

Proof: We show this by showing that the solution to Eq. 4.7 is a straight line with constant velocity, just like $\xi_{straight}$. The gradient of C is

$$\nabla_{\xi} C = -\ddot{\xi}$$

and setting this to 0 results in $\xi = az + b$. $\xi(0) = s \Rightarrow b = s$, and $\xi(1) = g \Rightarrow a = g - s$. Thus, $\xi = (g - s)z + s = \xi_{straight}$. ■

Lemma 2: $\sigma_s(\xi_{straight}) = \frac{1}{m} A^{-1}(0, .., 0, g)^T$ with m the last entry of A as in Eq. 5.8.

We can write constant speed straight line trajectories in closed form in terms of A.

Proof: From Lemma 1 and from $C[\xi] = \xi^T A \xi$, we infer that $\sigma_s(\xi_{straight})$, which is the straight line from 0 to $g - s$, is the solution to

$$\begin{aligned} &\min_{\xi \in \Xi_{s=0}} \xi^T A \xi \\ &s.t.\ \xi(1) = g - s \end{aligned} \tag{5.15}$$

Writing the Lagrangian and taking the gradient like before, we get that $\sigma_s(\xi_{straight}) = \frac{1}{m} A^{-1}(0, .., 0, g - s)^T$: this term is the straight line from 0 to $g - s$. ■

Lemma 3: $\hat{\xi}_{straight}$ is the solution to Eq. 5.6 for $\xi_D = \xi_{straight}$.

Constant speed straight lines get adapted by A to constant speed straight lines.

Proof: From Lemma 2, the term $\frac{1}{m} M^{-1}(0, .., 0, (\hat{g} - \hat{s}) - g)^T$ from Eq. 5.8 is the straight line from 0 to $(\hat{g} - \hat{s}) - (g - s)$, i.e., $((\hat{g} - \hat{s}) - g)t$.

Thus, Eq. 5.8 becomes

$$\begin{aligned}
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\Big(\sigma_s(\xi_{straight})+ \\
&\quad + \frac{1}{m}M^{-1}(0,..,0,(\hat{g}-\hat{s})-(g-s))^T\Big) \Rightarrow \\
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\left((g-s)z+((\hat{g}-\hat{s})-(g-s))z\right) \Rightarrow \\
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\left((\hat{g}-\hat{s})z\right) \Rightarrow \\
\hat{\xi} &= (\hat{g}-\hat{s})z+\hat{s} \Rightarrow \\
\hat{\xi} &= \hat{\xi}_{straight}
\end{aligned}$$

■

Theorem 1: $\hat{\mathcal{T}}$ is the solution to Eq. 5.6 for $\xi_D = \mathcal{T}$: *straight lines plus deviations get adapted by A to straight lines plus the same deviations, like the target trajectories in DMPs.*

Proof: When $f_D = 0$, $\mathcal{T} = \xi_{straight}$, and $\hat{\mathcal{T}} = \hat{\xi}_{straight}$. The theorem follows from Lemma 3.

When $f_D \neq 0$, the demonstrated target is $\mathcal{T}_D = \xi_{straight} + f_D$, and the adapted target is $\hat{\mathcal{T}} = \hat{\xi}_{straight} + f_D$. This adapted target still matches the solution in Eq. 5.8:

$$\begin{aligned}
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\Big(\sigma_s(\xi_{straight}+f_D)+ \\
&\quad + \frac{1}{m}M^{-1}(0,..,0,(\hat{g}-\hat{s})-(g-s))^T\Big) \Rightarrow \\
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\Big(\sigma_s(\xi_{straight})+f_D+ \\
&\quad + \frac{1}{m}M^{-1}(0,..,0,(\hat{g}-\hat{s})-(g-s))^T\Big) \Rightarrow \\
\hat{\xi} &= \sigma_{\hat{s}}^{-1}\left(f_D+(\hat{g}-\hat{s})z\right) \Rightarrow \\
\hat{\xi} &= f_D+(\hat{g}-\hat{s})z+\hat{s} \Rightarrow \\
\hat{\xi} &= \hat{\mathcal{T}}
\end{aligned}$$

■

Therefore, the target adaptation that the DMP does, from $\mathcal{T}$ to $\hat{\mathcal{T}}$, is none other than the Hilbert norm minimization from Eq. 5.1, with the same norm as the one often used in trajectory optimization algorithms like CHOMP.

Norm Minimization Directly in the Trajectory Space. Because the tracked target adaption from $\mathcal{T}$ to $\hat{\mathcal{T}}$ is a Hilbert norm minimization, then the corresponding adaptation in the space of trajectories, which adapts ξ_D into $\hat{\xi}$ by tracking $\hat{\mathcal{T}}$, is also the result of a Hilbert norm minimization.

To see this, let $\beta : \xi \mapsto \mathcal{T}$ be the function mapping a demonstrated trajectory to the corresponding tracked target like in Eq. 5.11. Given a particular spring damper system, β is a bijection: every demonstrated trajectory maps to a unique tracked target, and every tracked target maps to a unique trajectory when tracked by the spring damper. Furthermore, β is linear, due to Eq. 5.11 and additivity and homogeneity of differentiation.

Because β is bijective and linear, the norm A in the tracked target spaces induces a norm P in the trajectory space: $||\xi||_P = ||\beta(\xi)||_A$.

Theorem 2: The final trajectory obtained by tracking the adapted target $\hat{\mathcal{T}}$, $\hat{\xi} = \beta^{-1}(\hat{\mathcal{T}})$, is the closest trajectory to ξ_D that satisfies the new endpoint constraints with respect to the norm P: *the final trajectory in a DMP is the result of Eq. 5.1 for* $M = P$.

Proof: Assume $\exists \xi$ with endpoints $\hat{s}$ and $\hat{g}$ s.t. $||\xi_D - \xi||_P < ||\xi_D - \hat{\xi}||_P$, i.e., ξ is closer to ξ_D than $\hat{\xi}$ is. Then $||\beta(\xi_D - \xi)||_A < ||\beta(\xi_D - \hat{\xi})||_A \Rightarrow ||\beta(\xi_D) - \beta(\xi)||_A < ||\beta(\xi_D) - \beta(\hat{\xi})||_A \Rightarrow ||\mathcal{T}_D - \beta(\xi)||_A < ||\mathcal{T}_D - \hat{\mathcal{T}}||_A$, which contradicts Theorem 1: we know that $\hat{\mathcal{T}}$ is the closest to $\mathcal{T}_D$ w.r.t. the norm A given the endpoint constraints, thus $\beta(\xi)$ cannot be closer. ■

Therefore, DMPs adapt trajectories by minimizing a norm that depends on both A (the norm used to adapt the tracked target), as well as the particulars of the dynamical system (represented here by the function β).

5.2.3 *Implications*

Theoretical Implications. Our work connects DMPs to trajectory optimization, providing an understanding of what objective the DMP adaptation process is inherently optimizing.

Our work also opens the door for handling obstacle avoidance via planning. Currently with DMPs, obstacles that appear as part of new situations influence the adapted trajectory in a reactive manner, akin to a potential field. Certain more difficult situations, however, require using a motion planner for successful obstacle avoidance, which reasons about the entire trajectory and not just the current configuration. Using our generalization, a trajectory optimizer akin to CHOMP can search for a trajectory that minimizes the adaptation norm (as opposed to the trajectory norm, as in CHOMP) while avoiding collisions.

Practical Implications. First, the generalization frees us from the default A norm, and enables us to select more appropriate norms for each task. We discuss this benefit below.

Second, the generalization gives the robot the opportunity to *learn* how to adapt trajectories from the user. If the user is willing to provide not only a demonstration, but also a few adaptations of that

A is not always an appropriate choice for the Hilbert norm. Each of the plots below compares adapting a different original trajectory ξ_D (gray) using A ($\hat{\xi}_A$, blue) vs. using a better norm ($\hat{\xi}_M$, orange). The norms we used, much like A, do not allow free rotations, but free rotations could be obtained similarly to free translations.

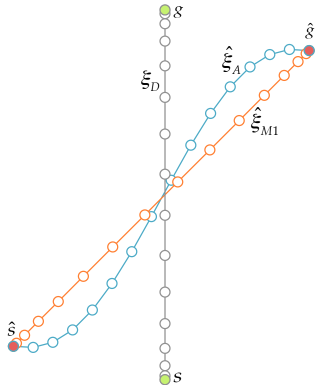


Figure 5.4: Minimum jerk.

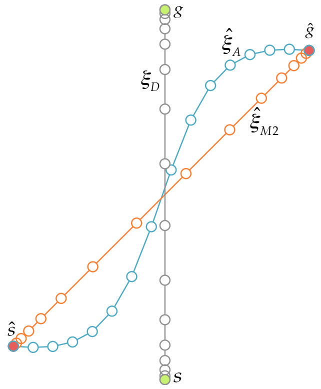


Figure 5.5: Reweighing time.

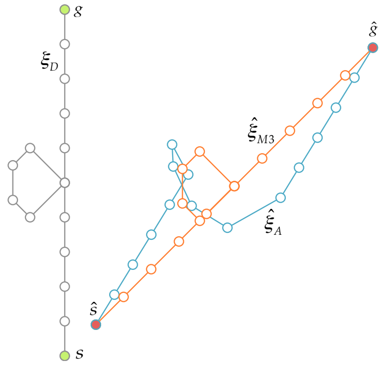


Figure 5.6: Coupling timepoints.

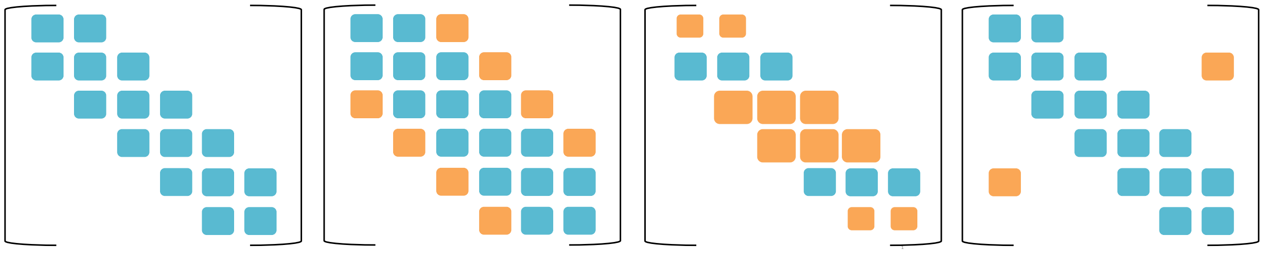

(a) Minimum Velocity (A) (b) Minimum Jerk ($M1$) (c) Reweighing Velocities ($M2$) (d) Coupling Timepoints ($M3$)

Figure 5.7: The different changes to the norm structure result in different adaptation effects.

demonstration to different start and goal configurations, then the robot can use this set of trajectories to learn the desired norm M. We describe an algorithm for doing so below.

Aside 1 — Computation. The adaptation in a DMP happens instantly, by instantiating the start and goal variables with new values. Hilbert norm minimization has an analytical solution, with computational complexity in the discrete case dominated by a single matrix multiplication. This means any DMP can adapt its moving target using norm minimization.

SELECTING A BETTER NORM. The norm A can lead to good adaptations (see Fig. 4.17), but it is not always the most suitable norm. Figures 5.4 through 5.6 show three cases where a different norm leads to better adaptations. In all three cases, the better norm is a modification of the matrix structure of A (as shown in Fig. 5.7).

The first case, Fig. 5.4, uses a demonstrated trajectory that minimizes jerk. Therefore, using a norm that stems from jerk as opposed to velocities, results in the correct adaptation — the minimum jerk trajectory (orange). This norm is band diagonal, like A, but has a winder band because computing the jerk requires terms further away from the current trajectory point than computing velocities (Fig. 5.7(b)).

Aside 2 — Using a Spring Damper. DMPs first cast the trajectory as a moving target tracked by a spring damper, and adapt the moving target trajectory. Hilbert norm minimization can be used to adapt trajectories both for the moving target, as well as for the demonstrated trajectory itself. The decision to use a spring damper is independent from the adaptation process.

The second case, Fig. 5.5, uses a demonstrated trajectory that moves faster in the middle than it does in the beginning and end. Therefore, a norm that weighs velocities in middle of the trajectory less than velocities at the endpoints (unlike A, for which the velocities at every time point matter equally), results in the adaption in orange: the trajectory remains a straight line, and follows a similar velocity profile as the demonstration. This norm is a reweighing of the rows of A (Fig. 5.7(c)).

The third case, Fig. 5.6, uses a loop as the demonstrated trajectory. The demonstration itself is not necessarily minimizing any L^2 norm. However, a more appropriate norm for adapting this demonstration couples waypoints that are distant in time but close in space: instead of only minimizing velocities, it also minimizes the distance between the two points that begin and end the loop. Unlike A, which is band diagonal, this norm also has entries far from the diagonal, depending on how far apart in time these two waypoints are (Fig. 5.7(d)).

LEARNING A BETTER NORM. As we saw in the previous section, different norms result in different ways of adapting a demonstrated trajectory. If the user providing the demonstration is willing to also provide example adaptations to new endpoints, then the robot can learn the norm M from these examples: *instead of adapting trajectories in a pre-defined way, the robot can learn from the user how it should adapt trajectories.*

Let $\mathcal{D} = \{\xi_i\}$ be the set of user demonstrations, each of them corresponding to a different tuple of endpoints $(\xi_i(0), \xi_i(1))$. The robot needs to find a norm M such that for each pair of trajectories $(\xi_i, \xi_j) \in \mathcal{D} \times \mathcal{D}$, ξ_j is the closest trajectory to ξ_i out of all trajectories between the new endpoints, $\xi_j(0)$ and $\xi_j(1)$, i.e., find a norm that explains why the user adapted ξ_i into ξ_j and *not* into any other trajectory:

$$||\xi_i - \xi_j||_M \leq ||\xi_i - \xi||_M, \ \forall \xi \in \Xi_{s=\xi_j(0)}^{g=\xi_j(1)} \tag{5.16}$$

Equivalently:

$$\begin{aligned} ||\xi_i - \xi_j||^2_M \leq \min_{\xi \in \Xi} & ||\xi_i - \xi||^2_M \\ s.t.\ \xi(0) &= \xi_j(0) \\ \xi(1) &= \xi_j(1) \end{aligned} \tag{5.17}$$

One way to find an M under these constraints is to follow Maximum Margin Planning [4]. We find M by minimizing the following expression:

$$\begin{aligned} \min_M \sum_{i,j} ||\xi_i - \xi_j||^2_M - \min_{\xi \in \Xi} & [||\xi_i - \xi||^2_M - \mathcal{L}(\xi, \xi_j)] \\ s.t.\ \xi(0) &= \xi_j(0) \\ \xi(1) &= \xi_j(1) \\ s.t.\ M \succ 0 & \end{aligned} \tag{5.18}$$

with $\mathcal{L}$ a loss function, e.g., a function evaluating to 0 when the trajectory matches ξ_j and to 1 otherwise, and $M \succ 0$ the positive-definiteness constraint.

If ξ^*_{ij} is the optimal solution to the inner minimization problem, then the gradient update is:

$$M = M - \alpha \sum_{i,j} [(\xi_i - \xi_j)(\xi_i - \xi_j)^T - (\xi_i - \xi^*_{ij})(\xi_i - \xi^*_{ij})^T] \tag{5.19}$$

followed by a projection onto the space of positive definite matrices.

Aside 3 — Geometry. An M that satisfies all the constraints only exists if the demonstrations in $\mathcal{D}$ lie in a linear subspace of Ξ of dimensionality $2d$, with d the number of degrees of freedom: the adaptation induces a foliation of the space, with each linear subspace of a demonstration and all its adaptations to new endpoints forming a plaque of the foliation. Fig. 5.3 depicts such a linear subspace, obtained by adapting ξ_D.

This follows from Eq. 5.5: the space of all adaptations of a trajectory is parametrized by the vectors λ and γ. Similarly, when we allow free translations, the linear subspace has dimensionality d (Eq. 5.7). Note that there are many norms that satisfy the constraints

---

[4] N. Ratliff, J. A. Bagnell, and M. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning (ICML)*, 2006

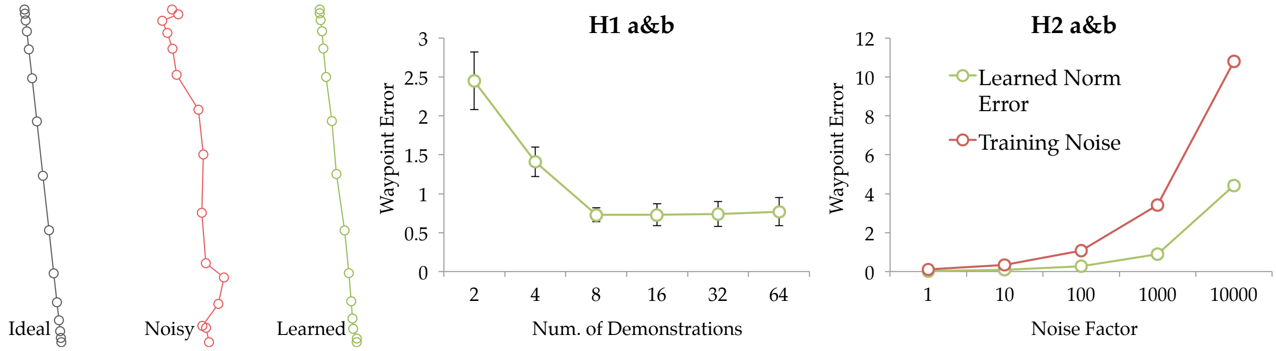


Figure 5.8: Left: an ideal adapted trajectory (gray), a noisy adapted trajectory (red) that we use for training, and the reproduction using the learned norm (green), with a 6-fold average reduction in noise. Center: the error on a test set as a function of the number of training examples. Right: the error on a test set as a function of the amount of noise, compared to the magnitude of the noise (red). Error bars show standard error on the mean — when not visible, the error bars are smaller than the marker size.

in this case, because only a subset of the rows of M^{-1} are used in the adaptation.

When the demonstrations do not form such a linear subspace, the algorithm will find an approximate M that minimizes the criterion in Eq. 5.18. We study the effects of noise in the next section. Other techniques for finding an approximate M, such as least squares or PCA, would also apply, but they would minimize different criteria, e.g., the difference between the trajectories themselves ($\sum ||\xi_j - \xi^*_{ij}||^2$), and not the difference between the norms.

5.2.4 *Experimental Analysis*

We divide our experiments in two parts. The first experiment analyzes the ability to learn a norm from only a few demonstrations, under different noise conditions. We do this on synthetically generated data so that we can manipulate the noise and compare the results to ground truth. We assume an underlying norm, generate noisy demonstrations based on it, and test the learner's ability to recover the norm. The second experiment tests the benefit of learning the norm with real kinesthetic demonstrations on a robot arm.

Synthetic Data. To analyze the dependency of learning the norm on the number of demonstrations, we generate demonstrations for different endpoints using a given norm M and some arbitrary initial trajectory. We then use the training data to learn a norm $\tilde{M}$. For simplicity, we focus on norms that allow free translations, and that do not couple different joints (similar to A).

Dependent Measures. We test the quality of a learned norm $\tilde{M}$ using two measures (which significantly correlate, see Analysis): one is about the norm itself, and the other is about the effect it has on adaptations.

Waypoint Error: This measure captures deviations of the behavior induced by the learned norm from desired behavior. We generate a test set of 1200 new start and goal configuration tuples for testing, leading to 1200 adapted trajectories using M as ground truth. We then adapt the demonstrated trajectory to each tuple using the learned norm $\tilde{M}$. For each obtained trajectory, we measure the mean waypoint deviation from the ground truth trajectories, and combine these into an average across the entire set.
Norm Error: This measure captures deviations in the learned norm itself (between M and $\tilde{M}$). Because only the last row of M^{-1} (which we denote M_N^{-1}) affects the resulting adaptation, we compute the norm of the component of the normalized $\tilde{M}_N^{-1}$ that is orthogonal to the true normalized M_N^{-1}.

Ideal Demonstrations. We first test learning from ideal demonstrations, meaning perfectly adapted using M, without any noise.

Because of the structure that M imposes on the optimal adaptations (a linear subspace of dimensionality $2d$ in general, d for free translations), only a few ideal demonstrations are necessary to perfectly retrieve M: 3 in the general case, and 2 in the case of free translations.

As a sanity check, we ran an experiment in which we chose the starting trajectory from Fig. 4.17 and generated 100 random norms. For each norm, we computed the two measures above. The resulting error was exactly 0 in each case: the learning algorithm perfectly retrieved the underlying norm.

Tolerance to Noise. Real demonstrations will not be perfect adaptations — they will be noisy. With noise comes the necessity for more than the minimal number of demonstrations, and the questions of how many demonstrations are needed and how robust the learning is to the amount of noise.
Manipulated Variables. In this experiment, we study these questions by manipulating two factors: (1) the **number of demonstrations**, and (2) the **amount of noise** we add to the adaptations in the training data.

We added Gaussian noise to the ideal adaptations using a covariance matrix that adds more noise to the middle of the trajectory than the endpoints (since the endpoints are fixed when requesting an adaptation).

For the first factor — number of demonstrations — we started at 2 (the minimum number required), and chose exponentially increasing levels (2, 4, 8, 16, 32, 64) to get an idea for what the scale of the number of demonstration should be. For the second factor, we scaled the covariance matrix (by 1, 10, 100, 1000, 10000) up to the point where the average noise for a trajectory waypoint was 50% of the average distance from start to goal (which we considered an extreme amount that exceeds by far levels we expect to see in practice). This resulted in 30 total conditions, and we ran the experiment with 30 different random seeds for each condition.

Hypotheses:

H1a. *The number of demonstration positively affects the learned norm quality.*

Sanity Check

H1b. *There is a point beyond which increasing the number of examples results in practically equivalent norm quality.*

We Only Need a Small Number of Examples

H2a. *The amount of noise negatively affects norm quality.*

Sanity Check

H2b. *The waypoint error is significantly lower than the noise on the training examples.*

Learning is Tolerant to Noise

Analysis. The waypoint error and norm error measures were indeed significantly correlated (standardized Crohnbach's $\alpha = 0.95$), suggesting that the waypoint error also captures the deviation from the real norm.

A factorial least squares regression revealed that, in line with H1a and H2a, both factors were significant: as the number of demonstrations increased, the error did decrease ($F(1, 867) = 24.07$, $p < .0001$), and as the amount of noise increased, the error did increase ($F(1, 867) = 628.35$, $p < .0001$).

Fig. 5.8 plots these two effects. In support of H1b, the error stops decreasing after 8 demonstrations (it takes a difference threshold of 0.3 for an equivalence test between the error at 8 and the error at 16 to reject the hypothesis that they are practically the same with $p = .04$). This suggests that learning the norm can happen from relatively few demonstrations.

In support of H2b, the error was significantly lower than the noise in the training trajectories ($t(899) = 19.35$, $p < .0001$): on average, the error was lower by a factor of 6.71, and this factor increased significantly with the number of demonstrations ($F(1, 869) = 869.01$, $p < .0001$).

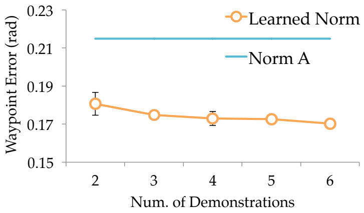
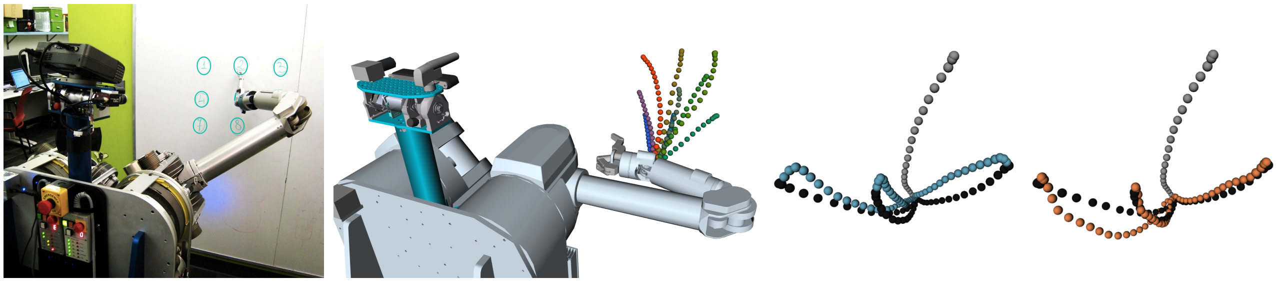


Figure 5.9: The average waypoint error on a holdout set of pointing gesture demonstrations on the HERB robot, for the adaptations obtained using the learned norm, compared to error when using the default A.

REAL DATA. Our simulation study compared the learned norm to ground truth. Next, we were interested in studying the benefits of learning the norm with real kinesthetic demonstrations on a robot arm.

We collected 9 expert demonstrations of pointing gestures on the HERB robot, where the task was to point to a particular location on a board, as in Fig. 5.10(a). We chose pointing as a task because the shape of the adapted trajectories is important for such gestures. We used up to 6 of these trajectories for training, and held out 3 for



(a) Robot Setup (b) Demonstrations (c) Norm A Adaptations (d) Learned Norm Adaptations

Figure 5.10: A comparison between adapting trajectories with the default A metric (c) and adapting using a learned metric (d) on a holdout set of demonstrated pointing gestures (shown in black). The trajectory ξ_D used for adaptation is in gray. Note that the adaption happens in the full configuration space of the robot, but here we plot the end effector traces for visualization. The learned norm more closely reproduces two of the trajectories, and has higher error in the third. Overall, the error decreases significantly (see Fig. 5.9).

testing.

Dependent Measures. We use the waypoint error measure from before, this time from the noisy holdout set as opposed to ground truth. We cannot use the norm error since we no longer have access to the true norm M.

Manipulated Variables. We used both the learned norm, as well as the default A norm from Eq. 4.7, to generate adaptations of the same original demonstration (its end effector trace is shown in gray in Fig. 5.10(c and d)). Note that even though the learned norm has access to more than the original demonstration, we used this demonstration only when testing the adaptation, to remain fair to the default norm. In practice, if the user provides multiple demonstrations, the one corresponding the situation closest to the test situation could be used for adaptation.

We also manipulated how many of the 6 demonstrations the learning algorithm used.

Hypotheses:

Data Improves Performance

H3. *As before, we expect that the number of demonstrations positively affects performance of the learned norm, i.e., error in reproducing the holdout trajectories decreases as the number of demonstrations increases.*

Learned Norm $>$ Default A

H4. *The learned norm has smaller error in reproducing the holdout demonstrations than the default A norm.*

Analysis. Fig. 5.10 qualitatively compares the learned and the default norm, and Fig. 5.9 plots our results.

Overall, the performance did tend to improve with the number of demonstrations, but the effect was not significant ($F(4,26) = 1.31$, $p = .29$). In support of H4, the error was significantly lower overall when learning the norm than when using the DMP default ($t(30) = 31.96$, $p < .0001$), suggesting that for real kinesthetic demonstrations, there is indeed a practical benefit to the generalization we propose in this paper.

IN SUMMARY, by learning the norm, the robot can produce trajectories in new situations that better match the desired shape, thus making the motion more predictable than when using a default adaptation procedure. The computation is instantaneous, and obstacles can be handle in the same way we do in trajectory optimization — by adding $\mathcal{U}_{obs}$ to the objective.

Limitations. Even a learned adaptation forces all the adaptation of a particular demonstration to lie in the same hyperplane. Norms that are richer than the L_2 norm would make this adaptation process even more flexible, though they would require more demonstrations.

5.3 *Familiarization to Robot Motion*

The previous section discussed one way for the robot to become more predictable: having the user train the robot through demonstrations — the user acts as the teacher, and provides demonstrations to the robot, which adapts its motion planner based on these examples.

In this section, we invert the teacher-learner relationship. Rather than focusing on the robot learning from the user's demonstrations (where it is difficult to obtain demonstrations [6], which, as we saw in the previous section, are then difficult to generalize), we explore the idea of the user learning from the robot's demonstrations, via *familiarization*:

Definition 5.3.1 *Familiarization to robot motion is the process of exposing the observer to how the robot moves in different situations.*

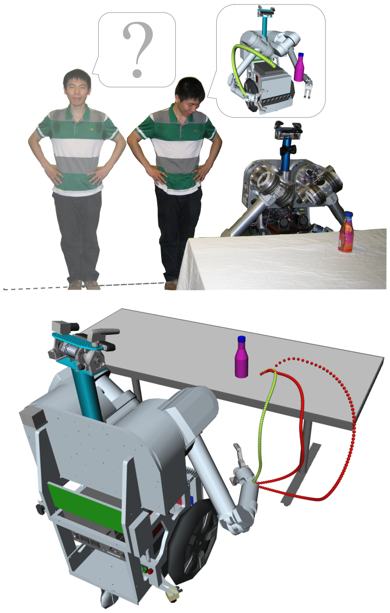


Figure 5.11: (Top) One of our users getting more comfortable with working/standing next to the robot after familiarization, as he can better predict how the robot will move. (Bottom) Users identify the robot's actual trajectory (we plot here its end effector trace only, in green, but show users the robot actually moving along it) as the one they expect more often after familiarization.

Many times, we take for granted that familiarization works. Familiarization is often used in studies prior to experimental conditions [121], under the assumption that it will adapt the user's mental model of the robot. Studies on sensemaking [227] support this assumption [198, 164, 14, 131, 208], as does the remarkable adaptability of humans: we learn new languages [162], adapt to new ways of communicating [209], and even remap existing sensors like our tongues to new senses, like vision [220].

Here, we study the effects of familiarization to *motion* on *predictability*. On the one hand, the breadth of human adaptability suggests that with familiarization, if the robot's motion is *consistent*, it will become significantly more predictable. On the other hand, the same obstacles robots face when learning motion — the high-dimensionality and complexity of the space — might induce similar limitations in humans.

We ran a series of three experiments investigating the effect of familiarization to two different types of motion, on the predictability of the motion. We also tested whether increased predictability matters by testing the users' comfort with the robot.

Our first experiment (Section 5.3.2) analyzed familiarization to consistent motion produced by our optimizer from Chapter 4. We evaluated predictability before and after familiarization by testing whether users identify the actual robot motion as the one they expect it to execute (from a set of different motions, see Fig. 8.12, bottom), as well as asking users to rate the motion on a subjective predictability scale.

We first tested whether consistent motion that is somewhat predictable to begin with becomes more predictable after familiarization.

Our results do support the utility of familiarization — *the motion became significantly more predictable*. However, we came across unexpected limitations of familiarization. We found that despite improving predictability, familiarization can fail to make the motion fully predictable, and can fail to generalize to new situations.

Next, we tested familiarization on a different type of motion. The initial study indicated that the optimizer-generated motion was moderately natural:

We tested how familiarization depends on how predictable the motion starts out to begin with, and what happens as the number of examples increases.

Definition 5.3.2 *Natural motion is motion that is predictable without (or prior to) familiarization.*

This finding raised an interesting question: would familiarizations still have an effect when the motion is less natural? In a second experiment (Section 5.3.3), we found that some unnatural motion may *never* reach a high predictability level, even when exposed to over twice the number of motions, suggesting that familiarization saturates.

Finally, we tested the practical effects of increasing predictability (Section 5.3.4) — does the user comfort with working or standing next to the robot also increase? We found a significant effect of familiarization on comfort. However, a lot of users over-trusted the robot, moving closer to it than would be safe. This has a surprising implication: less predictable motion might actually be safer in some situations, as it might prevent over-trust.

Our last study tests the effects of familiarization more practically, on the users' comfort with the robot

In summary, familiarization is an essential aspect of human-robot interactions, and it is important to study it and understand its limitations — sometimes, we cannot rely solely on human adaptability. Our data suggests that familiarization to motion helps, but *cannot be used exclusively for generating predictable motion*. The robot still has the burden of producing motion that is not too unnatural — motion *with which it is easy to familiarize*. However, given such motion, familiarization shows great promise for significantly improving predictability

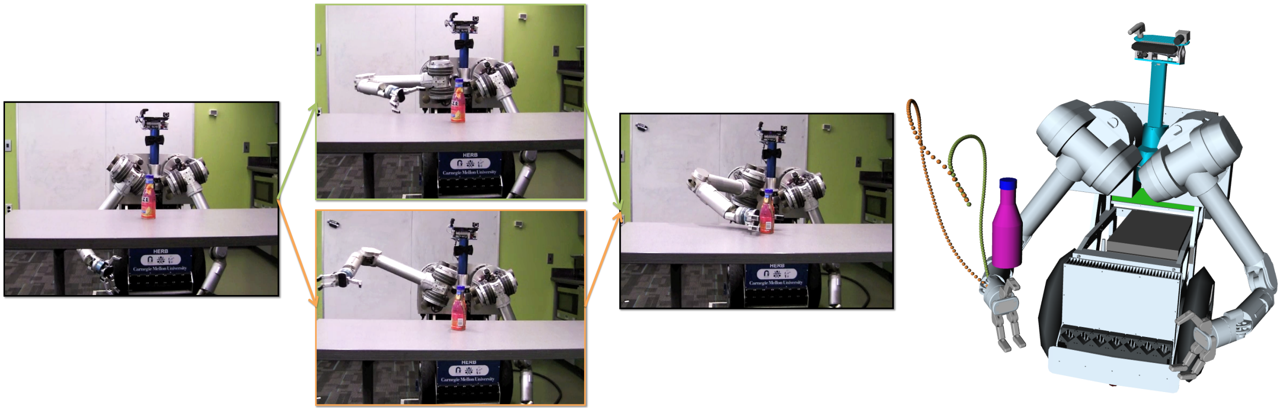


Figure 5.12: For the same situation, the trajectories for the more natural motion in Section 5.3.2 (top, green), and for the less natural motion in Section 5.3.3 (bottom, orange).

and ultimately enabling better human-robot collaboration.

5.3.1 *Generating Motion*

We generate motion using our optimizer from Chapter 4, using two different costs. We use our example cost C from Eq. 3.7 to generate *consistent* and relatively *natural* motion.

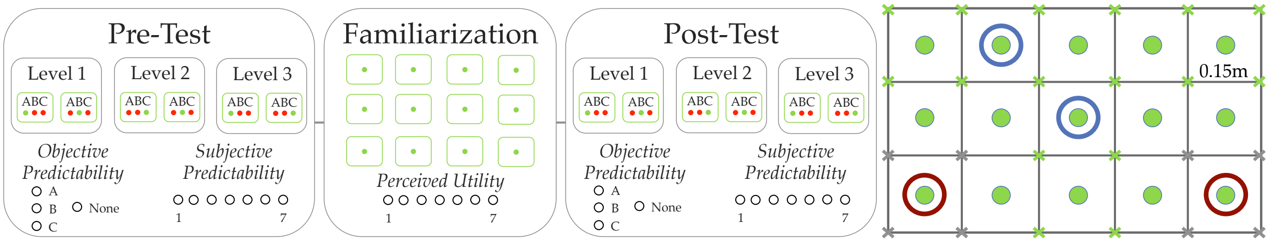
We use this cost in Section 5.3.2, when we test how useful familiarization is for state-of-the art generated motion. Fig. 5.12 (top, green) shows one of the example motions. We find that they are moderately natural, i.e., have good levels of predictability even before familiarization, and that familiarization increases their predictability further. This prompts us to test familiarization for less natural motion in Section 5.3.3 — would it still work?

To TEST FAMILIARIZATION ON LESS NATURAL MOTION, we changed the cost function. Rather than using our C, which uses the same weight on each of the robot's degrees of freedom, we weigh different degrees of freedom differently:

$$C_W[\xi] = \int ||\dot{\xi}(t)||_W^2 dt = \int \dot{\xi}(t)^T W \dot{\xi}(t) dt \tag{5.20}$$

By choosing a W with lower values for the shoulder joints and higher values for the wrist joints, the robot starts penalizing motion in the wrist, and starts moving it less at the expense of moving the shoulder more. This is contrary to what human motion does in reaching tasks [138], which suggests it will also make the robot's motion less natural. Our results in Section 5.3.3 support this.

Fig. 5.12 shows a comparison between the original cost function and this modified version (bottom, orange).



(a) (b) (c) (d)

Figure 5.13: The overall experimental procedure, consisting of a familiarization phase (b), and a pre- and post-test for predictability (a and c). The tests involve three types of examples (`Levels 1-3`), each with two instances to aid robustness. For each example, we show users three trajectories and ask them to identify which one they expect the robot to perform, as well as rate each on a predictability scale. The grid in (d) depicts target object placements on the table (shown in Fig. 8.12 and Fig. 5.12) to produce the familiarization examples. The ones we re-use for testing (`Level 1`) are highlighted in blue, and the ones we set aside for testing-only (`Level 3`) are highlighted in brown. The crosses represent additional example locations we use in the follow up study with more examples.

5.3.2 *Does Familiarization Work?*

We designed a user study to test the utility of familiarization to robot motion. Does exposing the users to how the robot moves *help* them form the right expectations in the future? And if so, *how good* do users get at predicting the robot's motion?

METHODS. We exposed users to examples of the robot's motion (Fig. 5.13, b), and measured improvement in predictability by administering a pre- (Fig. 5.13, a) and post-test (Fig. 5.13, c), using both objective and subjective measures. We detail our procedure below.

Design Decisions. The complexity and high-dimensionality of robot motion are the key obstacle to the utility of familiarization. *We designed our experiment to alleviate this issue: we focus on familiarization by training.*

We made familiarization a targeted learning experience, rather than treating it as exposure to the robot "in the wild". We chose a narrowly scoped task, structured the examples users see by parametrizing the task, and presented users with many examples comprising a good task discretization.

Robot Task. Rather than showing users a snippet of a daily activity, we chose to show them structured examples that better support learning. To do so, we narrowed the scope of our study to a single type of task, and extracted examples by parameterizing the task and discretizing the parameter space.

Of all possible tasks, we focus on *reaching motions*. Reaching for an object (and grasping it) is one of the most common manipulation tasks state-of-the art robots perform (along with placing): we see it in manufacturing environments [102] as well as in personal [204, 26] and assistive robotics [161].

We designed a typical reaching task, where HERB uses its right arm to reach for a target object on the table (see Fig. 5.12). We parametrize the task by a starting configuration for the arm, a goal configuration where the robot can grasp the target object, and obstacles in the environment which the robot's motion must not collide with.

We selected these parameters by replicating a scenario in which HERB drives up to the table and reaches for the bottle: we selected HERB's typical driving configuration as the start, and kept the table in place as the obstacle.

Example Number and Order. We generated examples by varying the goal parameter. We varied the location of the target object on the tabletop, as depicted in Fig. 5.13 (d). To aid familiarization, we discretized this space *finely*, forming a 5×3 grid with 0.15m resolution for where the bottle can be placed. This creates a space of 15 possible examples, 2 of which we kept aside for our pre- and post- test `Level 3`.

We followed human teaching patterns and presented the examples to the users in the order from most simple to most complex [214]. Here, we defined simplicity based on how efficient each trajectory was relative to the distance between the starting configuration and the goal.

User Instructions. We decided to specifically instruct the users to actively try to learn how the robot moves, in line with our decision of making this a learning task rather than a passive observation task.

Design Overview. We outline below the dependent and independent variables, as well as our subject allocation.

Manipulated Factors. We manipulate two factors: `familiarization` and `difference level`.

We manipulate familiarization by measuring predictability before and after the robot gives the user examples

We manipulate `familiarization` by testing the predictability of motion both `before` and `after` exposing the users to the examples. We use recordings of HERB executing the CHOMP-generated motions.

We manipulate difference level by having test situations that are more and more different from the training examples that users see.

With `difference level`, we look at test situations that relate in different ways to the examples.

We select two of the 13 possible scenarios the user will see during training and identify these as `Level 1` situations. Next, we select one of the two and change the start configuration or add another obstacle, and identify these as `Level 2` situations. Finally, the user is shown the two scenarios that will not be shown as part of the training set. These scenarios are `Level 3` situations.

Fig. 5.14 shows an example situation for each level. Since there is no clear ordering in terms of difficulty between levels 2 and 3, we keep this variable as nominal (as opposed to ordinal) in our analysis below.

We use two situations for each difference level (as opposed to only one) in order to alleviate the risk of introducing confounds in the manipulation. This leads to a total of 6 test situations, which we

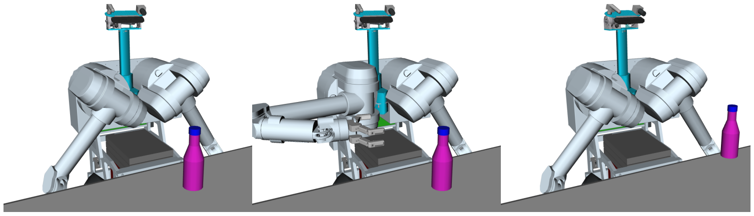

(a) Level 1 (b) Level 2 (c) Level 3

Figure 5.14: Example of the three distance levels.

present to the users in a randomized order both before and after familiarization.

Dependent Measures. We measured the predictability of the robot's motion in the 6 test situations using both an *objective*, as well as a *subjective* metric.

Objective Predictability. For our objective metric, we measured the accuracy with which users can identify the robot's actual motion from a set of different motions. This is a way of of objectively measuring whether users expect the motion that the robot would execute.

We measure predictability objectively by testing whether the participant can identify the robot's consistent motion from a set of trajectories.

For each test situation, we first presented the users with an image of the robot in the starting configuration, with the bottle placed in the corresponding location. We asked them to spend a minute imagining how they expect the robot to move his arm. To make sure they think the task through, we asked them to describe the motion.

Next, we showed them video recordings of HERB executing three motion trajectories (in randomized order). One of these is the actual trajectory (represented by a green dot in Fig. 5.13 (a) and (c)) produced according to the procedure outlined in Section 5.3.1.

We selected the other two motions (by varying the goal configuration) such that they are spatially similar either to the actual trajectory from the same situation, or to the actual trajectory from one of the example situations.

We imposed a minimum distance requirement on the test motions: they have to achieve a minimum distance (either at the end effector or at the elbow) from one another. We choose a threshold (of 0.2m) to signify "practical difference": if the users cannot distinguish among motions that are too similar, this has no practical side effect — at the limit, differences among motions will not even be observable to the naked eye; on the other had, if users mistake the motions for one in which the robot's arm reaches a different part of the space, this can have severe practical consequences when working next to the robot.

These two motions are represented as red dots in Fig. 5.13 (a) and (c). Fig. 8.12 (bottom) shows the end effector traces for the three candidate trajectories in one of the test situations.

After seeing the three trajectories, we asked the users a multiple-choice question: "Which of the trajectories matched the one you expected?". The choices were trajectories 1-3. as well as a "None" option (which, despite the strong wording in the question of having "matched" the expected trajectory, was only used in 12% of the cases).

Subjective Predictability. For our subjective metric, we designed a scale for predictability, comprised of three 1-7 Likert scale statements shown in Table 5.1.

We measure predictability subjectively by asking participants to rate the motion.

Table 5.1: The predictability scale.

Trajectory 'x' matched what I expected.
Trajectory 'x' is predictable.
I would be surprised if the robot executed Trajectory 'x' in this situation.

We asked users, after seeing all three trajectories, to indicate their level of agreement with each statement, for each trajectory (in order to not give away which trajectory HERB would actually execute). In our analysis below, we show that the scale has internal reliability, and combine the ratings for HERB's actual trajectory (with the third statement reverse-coded) into our subjective metric.

Aside from measuring the motion's predictability before and after familiarization, we were also interested in two additional measures: whether the users thought that familiarization helped, and what they thought of the robot's motion.

Perceived Utility. After we showed them the motion examples, users did Likert self reports on utility (whether seeing how HERB moves helps them predict how HERB would move in a new situation), on improvement (whether they are better now at predicting how HERB would move than they were originally), and on confidence (whether they are confident they can predict how HERB would move).

Motion Attributes. We also asked them about the motions that they saw. We were interested in whether the CHOMP motions made sense to them, whether they were more fluid or more machine-like than they originally expected, and whether they would be comfortable working next to the robot if it moved in the way they saw.

Subject Allocation. We opted for a within-subjects design. We explicitly wanted to measure predictability for the same user before and after `familiarization` in order to avoid additional variance. Furthermore, users never get to see what the right answer to the test situations are. This enables us to treat `difference level` as a within-subjects factor as well.

We recruited 25 users (11 female and 14 male, with ages between 19 and 56, $M = 34.68$, $SD = 10.29$, and only 5 reporting having a technical background) via Amazon Mechanical Turk. They performed the study in an average of 50 minutes. To avoid rushed re-

sponses, we prevented users from advancing in the task without watching all videos and answering all questions, and we asked control questions at the end to verify attention.
Hypothesis. *Familiarization significantly improves the predictability of motion, as reflected by both the objective accuracy metric, as well as the subjective user ratings.*

ANALYSIS. We analyze predictability through both the objective and subjective measures.
Accuracy (objective). Supporting our hypothesis, `familiarization` had a significant effect on the users' accuracy in recognizing HERB's actual motion, as indicated by a logistic regression using `familiarization` and `difference level` as factors ($\chi^2(1,300) = 8.53$, $p = .0035$). There was no main effect for `difference level`, and no interaction effect. A factorial repeated-measures ANOVA treating accuracy as a $0 - 1$ continuous variable ($F(1,270)$, $p = .0039$) confirmed the significance of `familiarization`. This test has the advantage of allowing a treatment of the user ID as a random variable, and is considered to be robust to dichotomous data [48].

Familiarization can make motion more predictable.

Prior to familiarization, users already had a 62% accuracy (significantly higher than the 33% random choice, Pearson $\chi^2(1,150) = 55.47$, $p < .0001$), suggesting that the CHOMP-generated motions were *moderately natural*.

Familiarization did significantly increase accuracy, but, surprisingly, only to 77%.

Although familiarization helps make the CHOMP motion more predictable, our data suggest that it has important limitations: despite the test situations coming from the same task as the training ones, and despite the fine discretization of the task space, users were not able to always identify the correct trajectory from other (spatially different) trajectories. For distance `Level 1`, i.e., testing situations that were also present in the training examples, the accuracy was highest, at 84%.

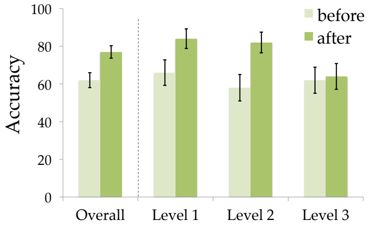


Figure 5.15: Overall, familiarization significantly improves the accuracy in recognizing the robot's motion (left). Different test situations, however, show different improvements (right). Error bars show standard error.

Fig. 5.15 shows the accuracy improvement after familiarization, both across tasks as well as split by the `difference level`: accuracy is highest on testing on situations that were also used as training examples (`Level 1`), as well as on situations with the same target location as a training example, but different starting/obstacle locations (`Level 2`). The test situations that were at the limits of the task (`Level 3`) did not see an improvement with familiarization (Fig. 5.15, right), suggesting that familiarization can have limited generalization.

Familiarization does not always make the motion fully predictable, and can have limited generalization ability.

Predictability Rating (subjective). Our scale for predictability comprised of ratings for expectedness, predictability, and surprise (reverse-coded) showed internal reliability (Chronbach's $\alpha = 0.91$), leading to a combined score for predictability based on the three ratings. This score is correlated with the accuracy (Pearson's $r(288) = .73$, $p < .0001$).

To test the effects of `familiarization` and `difference level` on this score, we ran a factorial repeated-measures ANOVA. This showed a significant main effect for `familiarization` ($F(1, 270) = 10.17$, $p = .0016$), but not for `difference level` (and no interaction effect). These results are consistent with our findings for accuracy, and strengthen the utility of familiarization to robot motion.

Table 5.2: The utility of familiarization ratings.

Utility Type	M	SD	$t(24)$ for $M \neq 4$	p
example helpfulness for prediction	5.76	1.09	8.06	<.0001
improvement in prediction capability	5.6	1.32	6.04	<.0001
confidence in prediction capability	5.76	0.97	9.07	<.0001

Perceived Utility. Table 5.2 shows the responses for the perceived utility of familiarization. Participants thought that seeing the videos helps them predict how HERB will move in a new situation, that they are better at predicting how HERB will move in a new situation than they were before seeing the videos, and were confident they can make this prediction accurately. These ratings are significantly different from the neutral stance of 4 (1-7 scale), even after Bonferroni corrections for multiple comparisons.

Table 5.3: The motion ratings.

Motion Attribute	M	SD	$t(24)$ for $M \neq 4$	p
makes sense	6.56	0.71	17.98	<.0001
more fluid than expected	5.52	1.66	4.57	<.0001
more machine-like than expected	2.32	1.62	-5.17	<.0001
comfort for collaboration	5.8	1.15	7.79	<.0001

Motion Attributes. Table 5.3 shows the responses for the motion attribute questions, together with the results of a t-test against the neutral mean of 4. Participants strongly agreed that HERB's motion made sense. They also agreed that the motions are more fluent than they originally expected, and disagreed that the motions were more machine like. All participants but one reported that they would be comfortable collaborating with HERB on a close-proximity task if it moved in the way they saw. The means are significantly different from the neutral stance, and remain significant after Bonferroni corrections.

These findings, together with the initial accuracy on CHOMP motions, suggest that CHOMP makes a good starting choice for familiarization. The next section will put familiarization to a more difficult test. It will study the effect of familiarization for less natural motions — does it still work, and how predictable do these motions become?

5.3.3 *Familiarization to Unnatural Motion*

Our results showed an improvement with familiarization when the motions are moderately natural. This led us to wonder: what if the robot moved in an unnatural way? Would familiarization still increase predictability?

METHODS. To investigate the effect of familiarization on less natural motion, we ran the same study, replacing the type of motion performed by the robot with the less natural version from Section 5.3.1, also depicted in Fig. 5.12 (bottom, orange).

For the testing situations, we were interested in whether familiarization would change the users' model and make them select the actual trajectory against the more natural CHOMP one. Thus, we selected the original CHOMP trajectory as one of the alternatives whenever possible, i.e., whenever it was practically different (using our definition of having a difference in the end effector or elbow locations of above 0.2m).

We recruited 25 new users via Amazon Mechanical Turk, and eliminated 1 for failing to answer the control questions correctly, leading to 11 male and 13 female users, with ages between 19 and 45 ($M = 29.16$, $SD = 7.12$).

Hypothesis. *Familiarization significantly improves the predictability of the less natural motion. Furthermore, it brings the less natural motion to the same predictability level as the more natural motion.*

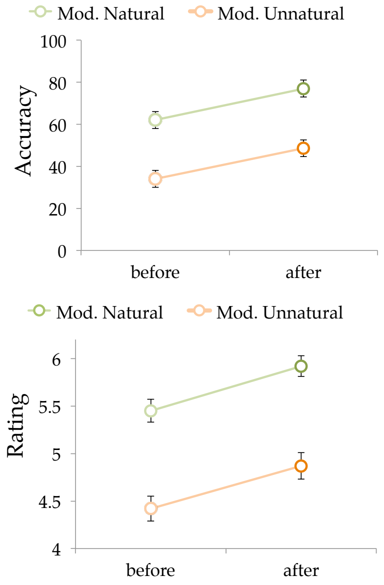


Figure 5.16: Results for familiarization to a less natural motion, as compared to the more natural CHOMP motion from Fig. 5.15. The error bars represent standard error on the mean. Familiarization does improve predictability, but not to the level of the more natural *C* motions.

ANALYSIS. We analyze the subjective and objective measures, provide a combined analysis with the previous experiment, and run a follow-up testing whether adding more examples helps.

Manipulation Check. The initial accuracy this time was only 34% (close to the random choice mark of 33%[5]). This confirms that the motions were less natural (less predictable before familiarization) than the CHOMP motions from the previous section ($\chi^2(1,588) = 47.38$, $p < .0001$). We call this type of motion *moderately unnatural*: low accuracy without going below the random choice threshold.

Accuracy and Rating Fig. 5.16(top) shows the accuracy before and after familiarization, as compared to the data from the more natural motions in the previous section. Consistent with our previous findings, and with our hypothesis, `familiarization` has a significant positive effect on accuracy, as evidenced by a logistic regression with

[5] this is 25% if we take the "none" option into account

our two factors ($\chi^2 = 6.95$, $p = .0084$).

Despite this improvement, the accuracy after familiarization is merely 48% — familiarization fails to bring this motion to the same predictability level that it brings the CHOMP motion (i.e., 77% accuracy). This is also supported by the ratings on our predictability scale: although familiarization has a positive main effect on the score ($F(1, 286) = 5.09$, $p = .0248$), the score after familiarization is significantly lower than for the CHOMP motion, as seen in Fig. 5.16(bottom).

Furthermore, for the test situations where a CHOMP trajectory was one of the options, more users chose the CHOMP trajectory (48%) than the trajectory generated by the cost function with which they were familiarized (43% on these situations).

Given that the initial accuracy on these tests was 29%, familiarization did change the users' model of how the robot moves, *but was not enough to make the true model more likely in their view than the more natural CHOMP model*.

Combined Analysis. The difference between the moderately natural and the moderately unnatural motions is also reflected when looking at the data overall. A logistic regression with `naturalness` (low versus high), `familiarization`, and `difference level` as factors shows significant main effects for all three factors (`naturalness` $\chi^2(1, 588) = 49.71$, $p < .0001$; `familiarization` $\chi^2(1, 588) = 15.46$, $p < .0001$; `difference level` $\chi^2(2, 588) = 14.26$, $p = .0008$). It also shows an interaction effect between difference level and initial predictability ($\chi^2(2, 588) = 10.19$, $p = .0061$).

A factorial repeated-measures ANOVA yielded the same results, and the Tukey HSD post-hoc analysis on the interaction effect revealed that all conditions for the moderately natural motions had significantly better accuracy than all moderately unnatural conditions, with the exception of difference level 2. The tests in this level maintained high accuracy, possibly due to a similarity in the motion for the test situations in this difference level).

Overall, we see that lower naturalness of motion results in lower predictability even after familiarization.

Familiarization can fail to bring less natural motion to the same predictability levels it brings more natural motion.

FOLLOW-UP: DO WE JUST NEED MORE EXAMPLES? Upon finding this limitation, we wondered: could we bring predictability levels as high as for the more natural motion by simply increasing the number of examples? Is this limitation caused by the amount of familiarization?

Methods. We tested this in a follow-up study. We created more examples by discretizing the space further, as shown by the grid crosses in Fig. 5.13. After eliminating the ones close to the testing situations from `Level 3` (shown in gray in the figure), we obtained 16 new examples (leading to a possible total of $13 + 16 = 29$).

We replicated our previous study, manipulating one additional factor — the `number of examples` — with 3 levels: 13 (previous study), 21, and 29. We added the additional examples after the original ones, maintaining their order and thus avoiding the order of the examples as a confound.

The `number of examples` factor was between-subjects.

This was necessary in order to manage the different number of examples in the familiarization stage. We recruited 25 users per level of examples.

Analysis. Fig. 5.17 shows the accuracy before and after familiarization for each case.

Familiarization can saturate: final predictability does not increase with the number of examples.

Surprisingly, accuracy decreases in the last case, with the largest number of examples. This decrease is significant in a logistic regression over all example levels, which shows a main effect for `number of examples` ($\chi^2(2, 876) = 6.85$, $p = .0325$), and marginally significant in a factorial repeated-measures ANOVA ($F(2, 70) = 2.80$, $p = .0675$).

The accuracy after familiarization with 29 examples is consistently smaller than with 13 or 21, in particular for difference level 1, i.e., tests that appear in the training data.

This could imply that with more examples to learn from, users are more focused on a general model and less able to keep in mind particular cases. Rather than over fitting to the limited number of examples, users might be fitting a more general but less accurate model. There can also be something specific to the examples added that adds confusion. Further investigation is needed in order to understand this drop, and verify it is not produced by chance.

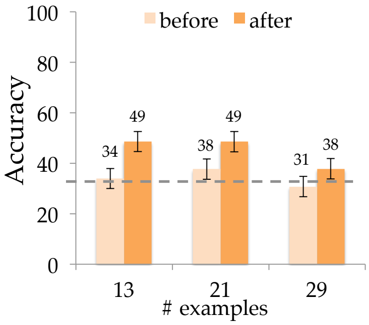


Figure 5.17: The limitation of familiarization on less natural motion is not due to the number of examples, since more examples fail to improve performance.

5.3.4 *Familiarization and Comfort*

In the previous sections, we found that familiarization increases the motion's predictability. Here, we are taking a first step towards analyzing the practical consequences of improved predictability to human-robot collaboration. In particular, does familiarization improve the users' comfort with working next to the robot?

METHODS. We designed an experiment where we evaluated user comfort before and after familiarization, using both an indirect, objective metric, as well as a direct, subjective metric.

Manipulated Factors. We manipulated two factors: `familiarization` and `naturalness` of the motion. We used the same familiarization procedure as before, and the two motions from Section 5.3.2 and Section 5.3.3.

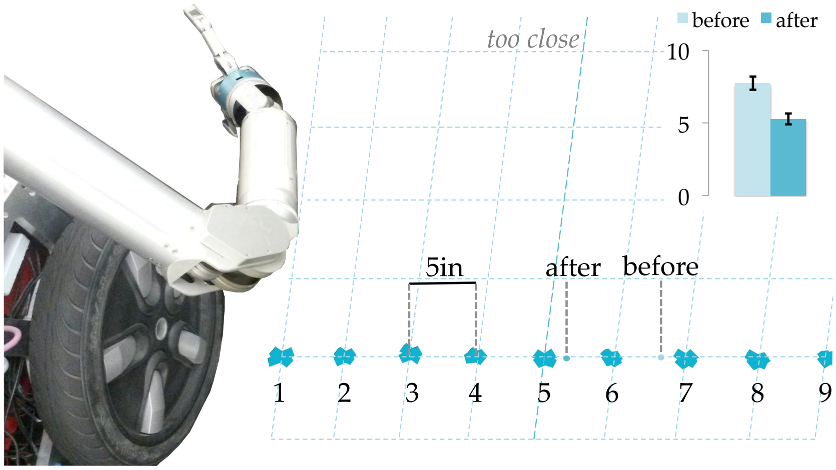


Figure 5.18: Markers measuring distance to the robot are spaced 5 inches apart. Familiarization brought users 7.35 inches closer to the robot.

We decided against manipulating the `difference level` factor in this study, and only used a `Level 1` situation. We could not use `Level 3`, as familiarization had no effect on the predictability of motion in situations from this level. Furthermore, our pilot for this study (with 6 users) showed no differences between `Level 1` and `Level 2`.

Dependent Measures. We evaluated comfort in two ways:

Objective Comfort. The robot was set in a `Level 1` situation, in the starting configuration. The experimenter told the users that the robot will move to reach for the target object, and asked them to stand side-by-side with the robot, as close as possible, but far enough away that they felt confident that the arm would not hit them as it moves during the reach (Fig. 8.12).

We marked the floor with 18 marks, starting right next to the robot and moving outward, placed every 5 inches (Fig. 5.18). We measured the distance (marker ID) from the user to the robot.

Although indirect, this metric is of high practical relevance for collaboration: we want users to be comfortable enough to get close to the robot as it is working, in oder to be able to do their own tasks simultaneously.

Subjective Comfort. We also directly asked users to indicate (on a Likert scale from 1 to 7), their level of agreement with the statement: "I would feel comfortable working side by side with the robot on a close-proximity task like cleaning up the dining room table." (which we augmented with "if it moved in the way I saw" after familiarization).

Subject Allocation. We used a mixed design. We kept `familiarization` within-subjects, measuring improvement in comfort before and after exposure to the robot's motion. However, `naturalness` was between-subjects, as each user could only familiarize with one type of motion (to avoid confusion and ordering effects).

We recruited 16 users from the local community (9 female and 7 male, with ages between 20 and 64, $M = 36.68$, $SD = 16.6$, with 7 reporting having a technical background).

Hypothesis. *Familiarization significantly improves comfort with working next to the robot, as indicated by both the objective and subjective metrics.*

ANALYSIS. We were very surprised by how comfortable users were with the robot to begin with: with no prior knowledge of how HERB moves, users stood only 33 inches from the robot's arm, while the arm could touch them even at 45 inches away. A particularly trusting user stood only 20 inches away, which makes it very difficult for the robot to avoid them even when it knows exactly where they are. Users also rated their comfort with the robot very highly ($M = 6.52$, $SD = 0.61$).

A factorial ANOVA showed a significant main effect for `familiarization` on our objective metric ($F(1, 14) = 12.68$, $p = .0031$): in line with our hypothesis, users were willing to stand closer to the robot after familiarization ($M = 5.28$, $SD = 1.49$) than they were initially ($M = 6.75$, $SD = 1.84$) — a difference of $1.47 \times 5 = 7.35$ inches.

Familiarization can increase comfort with working side-by-side with the robot.

We found no effect of `familiarization` on our subjective metric. The mean improved ever-so-slightly ($M = 6.56$, $SD = 0.51$).

Although there was no significant effect for `naturalness`, the means for the objective metric reveal that users did stand slightly further away in the unnatural condition. The means very closely matched the actual safe distances (5.06 for the natural case, and 5.5 for the unnatural case) — users were surprisingly good at estimating the correct spot on which stand, on average.

However, this has an interesting side-effect: familiarization made a lot of users *over-trust* the robot, in that it made them stand *too close* to it (5 out of 8 in the natural condition, and 3 out of 8 in the unnatural condition). Overall, `familiarization` had a marginally significant effect on whether users over-trusted the robot ($\chi^2(1, 32) = 3.56$, $p = .0592$), which could have a startling implication:

Less predictable motion might actually be safer in some cases, in that it might prevent over-trusting the robot.

This echoes findings in the trust literature: unreliable behavior increases trust [6]. However, when the robot needs to be conservative about safety (e.g., in the case of industrial arms), this can be a desired effect.

IN SUMMARY, we did find that motion becomes significantly more

[6] Munjal Desai, Mikhail Medvedev, Marynel Vázquez, Sean McSheehy, Sofia Gadea-Omelchenko, Christian Bruggeman, Aaron Steinfeld, and Holly Yanco. Effects of changing reliability on trust of robot systems. In *HRI*, 2012

predictable after familiarization, at least when the familiarization process is presented as a learning task. We also found that users' comfort level increases.

LIMITATIONS. On the other hand, we found that familiarization is not always enough to enable users to identify the robot's motion (despite choosing among spatially different trajectories), and that less natural motion reaches lower predictability levels. Our data suggests that this limitation can not (at least not always) be overcome by increasing the familiarization length: familiarization can saturate.

Furthermore, our experiments used a pre-test, which could prompt the users' learning toward test situations, and inflate the effect of familiarization. Predictability after familiarization could be even lower than our measurements indicate.

Of all the factors that could affect the utility of familiarization, our experiments touched upon two: the naturalness of motion, and the number of examples the robot gives the users. Many other factors could impact familiarization: the anthropomorphism of the robot (would users have a harder time with less anthropomorphic robot?), the dimensionality of the space (would they have an easier time with robots with fewer DOF?), the convexity of the cost function the robot optimizes (does non-convexity affect humans as it does machines?), the breadth of examples (one task vs. many), as well as the order or the examples.

5.4 Chapter Summary

In this chapter, after deriving the gradient descent update rule for predictable motion using a straw-man C, we explored two complementary ways of improving predictability: having the robot adapt to the human (learning from demonstration), and having the human adapt to the robot (familiarization).

$\text{human} \xrightarrow{\textit{examples}} \text{robot}$

$\uparrow$

gradient-based optimization for predictability

$\downarrow$

$\text{robot} \xrightarrow{\textit{examples}} \text{human}$

In learning from demonstration, we used a local trajectory adaptation approach, but cast it as a general trajectory optimization problem — by learning the adaptation norm, we we were able to bring an optimization and Inverse Reinforcement Learning perspective on these techniques.

With familiarization, we saw that familiarization does help. However, although our studies were controlled and focused, they revealed surprising limitations of familiarization. Given that we made optimistic choices for the factors we did not manipulate, aiding familiarizations, we expect to see similar limitations when performing familiarization "in-the-wild": familiarization improves predictability, but the robot still faces the challenge of producing good motion with which to familiarize.

6

Generating Legible Motion

Planning
Legible Motion

Chapter 3 introduced a mathematical measure for legibility — the LEGIBILITY score from Eq. 3.19. In this chapter, we go from the ability to *evaluate* how legible a trajectory is, to the ability to *generate* legible motion. This demands going beyond modeling the observer's goal inference, to creating motion that results in the correct goal being inferred, i.e., going from "I can tell that you believe I am grasping this.", to "I know how to *make* you believe I am grasping this".

We build on functional gradient descent (Chapter 4) in Section 6.1 to optimize the our legibility measure. Fig. 6.1 depicts this optimization process: by exaggerating the motion to the right, the robot makes the other goal option, G_O, far less likely to be inferred by the observer that the correct goal G_R.

The ability to optimize for legibility led us to a surprising observation: that there are cases in which the trajectory becomes too *unpredictable*. As we saw in Section 3.4, some unpredictability is sometimes necessary to convey intent — it is (like the outermost trajectory in Fig. 6.1) that confuses users and lowers their confidence in what the robot is doing, leading to an additional, "something else" hypothesis.

We address this fundamental limitation by prohibiting the optimizer to "travel to uncharted territory", i.e., go outside of the region in which its assumptions have support — we call this a "trust region" of predictability (Section 6.2). The trust region serves as an approximation to preventing the "something else" hypothesis from gaining too much probability mass. Our user studies indicate that indeed, there exists a size for this region in which legibility improves in practice, but outside of which the users' confidence in knowing the robot's goal drops.

6.1 *The Legibility Gradient*

We optimize for legibility by iteratively following Eq. 4.12, using the predictable motion as ξ_0. To instantiate the update rule, the robot

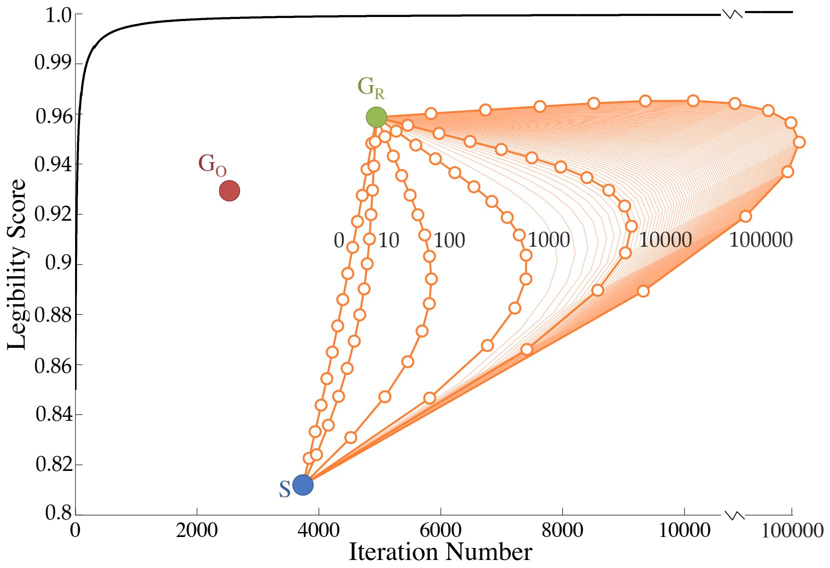


Figure 6.1: The legibility optimization process for a task with two candidate goals. By moving the trajectory to the right, the robot is more clear about its intent to reach the object on the right.

needs access to $\nabla\mathcal{U}$, and we use the negative Legibility (which we want to maximize rather than minimize, as before) as the prior term in $\mathcal{U}$.

Notation. Let $\mathcal{P}(\xi(t),t) = P(G_R|\xi_{S\rightarrow\xi(t)})f(t)$ and $K = \frac{1}{\int f(t)dt}$. The legibility score is then

$$\textsc{Legibility}[\xi] = K\int \mathcal{P}(\xi(t),t)dt \tag{6.1}$$

Set up Legibility for Euler-Lagrange.

Further, let

$$g = \exp\big(V_{G_R}(S) - V_{G_R}(Q)\big) \tag{6.2}$$

$$h = \sum_G \exp\big(V_G(S) - V_G(Q)\big)P(G) \tag{6.3}$$

The probability of a goal is

$$P(G_R|\xi_{S\rightarrow\xi(t)}) = \frac{g}{h} \tag{6.4}$$

and

$$\mathcal{P}(\xi(t),t) = \frac{g}{h}f(t) \tag{6.5}$$

Set up $\mathcal{P}$ for quotient rule.

Derivation. Analogous to Euler-Lagrange:

$$\nabla\textsc{Legibility} = K\left(\frac{\partial\mathcal{P}}{\partial\xi} - \frac{d}{dt}\frac{\partial\mathcal{P}}{\partial\xi'}\right) \tag{6.6}$$

$\mathcal{P}$ is not a function of ξ', thus $\frac{d}{dt}\frac{\delta \mathcal{P}}{\delta \xi'} = 0$.

$$\frac{\delta \mathcal{P}}{\delta \xi}(\xi(t), t) = \frac{g'h - h'g}{h^2} P(G_R) f(t) \tag{6.7}$$

which after a few simplifications becomes

$$\begin{aligned}
&\frac{\partial \mathcal{P}}{\partial \xi}(\xi(t), t) = \frac{\exp\big(V_{G_R}(S) - V_{G_R}(\xi(t))\big)}{\big(\sum_G \exp\big(V_G(S) - V_G(\xi(t))\big)P(G)\big)^2} P(G_R) \\
&\sum_G \left(\frac{\exp\big(-V_G(\xi(t))\big)P(G)}{\exp\big(-V_G(S)\big)} \big(V'_G(\xi(t)) - V'_{G_R}(\xi(t))\big) \right) f(t) \\
&= P(G_R | \xi_{S \to G_R}) \sum_G \Big(P(G | \xi_{S \to \xi(t)}) \big(V'_G(\xi(t)) - V'_{G_R}(\xi(t))\big) \Big) f(t)
\end{aligned} \tag{6.8}$$

Finally,

$$\nabla \text{Legibility}(t) = K \frac{\partial \mathcal{P}}{\partial \xi}(\xi(t), t) \tag{6.9}$$

with $\frac{\partial \mathcal{P}}{\partial \xi}(\xi(t), t)$ from (6.8).

The gradient has an intuitive direction: it pushes each point along the trajectory in the direction that minimizes the cost-to-go to the actual goal G_R $(-V'_{G_R})$, and away from the direction that minimizes the cost-to-go to the other goal, for every pair $(G_R,$ other goal$)$:

> *Legibility is about conveying that the robot's goal is the actual goal, but also conveying that **it is not any of the other candidate goals** that the observer might infer instead.*

Without obstacle avoidance, we follow Eq. 4.12 with only this gradient:

$$\xi_{i+1} = \xi_i + \frac{1}{\eta} A^{-1} \nabla \text{Legibility} \tag{6.10}$$

In the presence of obstacles, we use

$$\nabla \mathcal{U}_{prior} = -\nabla \text{Legibility} \tag{6.11}$$

EXAGGERATION EMERGES OUT OF LEGIBILITY OPTIMIZATION. Fig. 6.1 shows the optimizer at work in the center of the image, and also plots the score over iterations. Note that this uses a very small step size — in practice, with a larger $\frac{1}{\eta}$, only a few iterations are needed.

Fig. 6.2 shows the optimization for HERB for a reaching task, and Fig. 6.3 shows the initial trajectory along with an optimized one.

In both cases, the robots start with a straight trajectory to their goals, and autonomously start exaggerating the trajectory to the right so that the goal on the right becomes more clear.

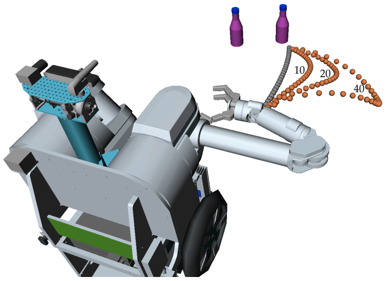


Figure 6.2: Legible trajectories on a robot manipulator assuming C, computed by optimizing Legibility in the full dimensional space. The figure shows trajectories after 0 (gray), 10, 20, and 40 iterations.

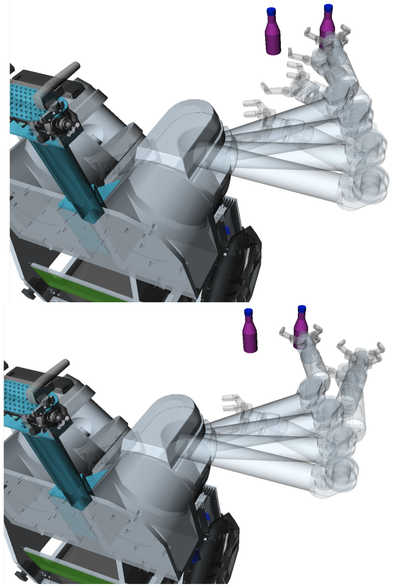


Figure 6.3: A full-arm depiction of the optimized trajectories at 0 and 20 iterations.

Exaggeration is one of the 12 Disney principles of animation, but nowhere did we have to encode exaggeration as a strategy: the robots figured out that they should exaggerate, as well as the details of that exaggeration:

Exaggeration naturally emerged out of the mathematics of legible motion.

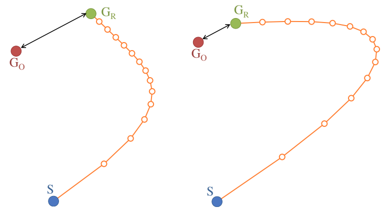


Figure 6.4: More ambiguity (right) leads to the need for greater departure from predictability.

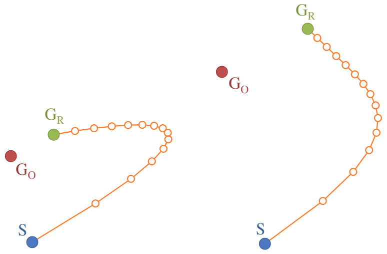


Figure 6.5: Smaller scales (left) lead to the need for greater departure from predictability.

Understanding Legible Trajectories Armed with a legible motion generator, we investigate legibility further, looking at factors that affect the final trajectories.

Ambiguity. Certain scenes are more ambiguous than others, in that the legibility of the predictable trajectory is lower. The more ambiguous a scene is, the greater the need to depart from predictability and exaggerate the motion. Fig. 6.4 compares two scenes, the one on the right being more ambiguous by having the candidate goals closer and thus making it more difficult to distinguish between them. This ambiguity is reflected in its equivalent legible trajectory (both trajectories are obtained after 1000 iterations).

Scale. The scale does affect legibility when the value functions V_G are affected by scale, as in our running example. Here, reaching somewhere closer raises the demand on legibility (Fig. 6.5). Intuitively, the robot could still reach for G_O and suffer little penalty compared to a larger scale, which puts an extra burden on its motion if it wants to institute the same confidence in its intent.

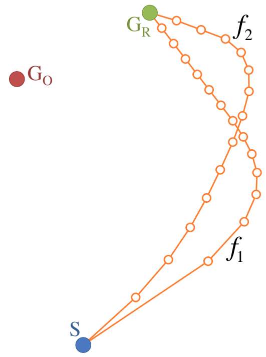


Figure 6.6: Effects of the weighting function $f(t)$.

Weighting in Time. The weighting function f from Eq. 3.19 qualitatively affects the shape of the trajectory by placing the emphasis (or exaggeration) earlier or later (Fig. 6.6). f can capture the need to convey the intent as early as possible, decaying as the trajectory progresses. An exponential decaying f is analogous to a discount factor in an MDP, discounting future reward (here, the probability of the correct goal). In the limit, f can cause the robot so "signal" in the very beginning, a strategy that our animator from Section 8.2 uses.

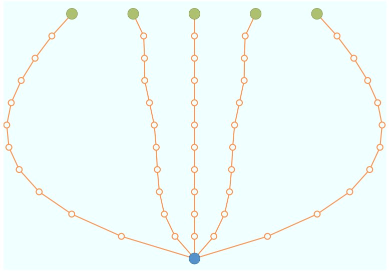


Figure 6.7: Legible trajectories for multiple goals.

Multiple Goals. Although for simplicity, our examples so far were focused on discriminating between two goals, legibility does apply in the context of multiple goals (Fig. 6.1). Notice that for the goal in the middle, the most legible trajectory coincides with the predictable one: any exaggeration would lead an observer to predict a different goal — *legibility is limited by the complexity in the scene.*

Obstacle Avoidance. We plot in Fig. 6.8 what we happens when C itself trades off between efficiency and obstacle avoidance, i.e., we use a new $C' = C + \mathcal{U}_{obs}$. Legibility in this case will move the predictable trajectory much closer to the obstacle in order to disambiguate between the two goals.

Local optima. There is no guarantee that Legibility is concave. This is clear for the case of a non-convex C, where we often see differ-

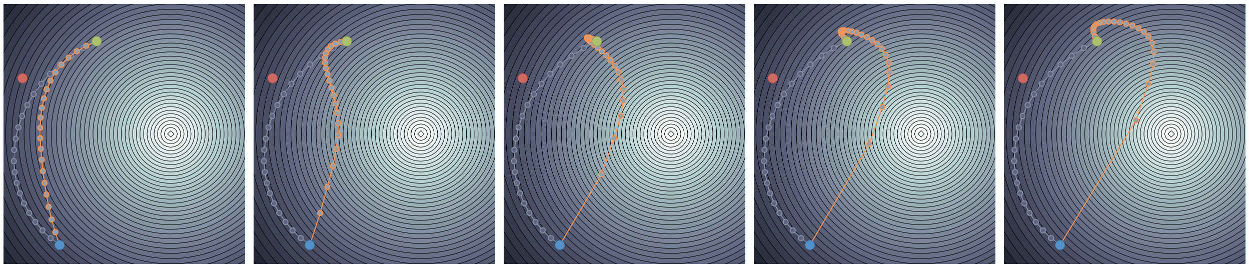


Figure 6.8: Legibility given a C that accounts for obstacle avoidance. The gray trajectory is the predictable trajectory (minimizing C), and the orange trajectories are obtained via legibility optimization for 10, 10^2, 10^3, 10^4, and 10^5 iterations.

ent initializations lead to different local maxima.

In fact, even for quadratic V_Gs, $P(G_R|\xi_{S\to Q})$ is — aside from scalar variations — a ratio of sums of Gaussian functions of the form $\exp\big(-V_G(\xi(t))\big)$. Convergence to local optima is thus possible even in this simple case.

As a side-effect, it is also possible that initializing the optimizer with the most predictable trajectory leads to convergence to a local maxima.

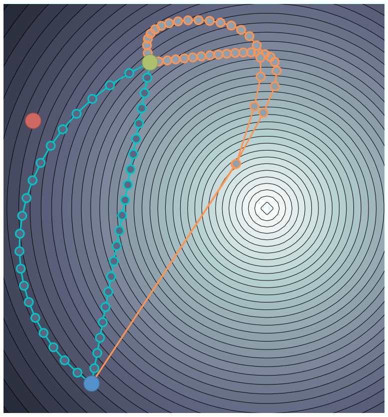


Figure 6.9: Legibility is dependent on the initialization.

6.2 Trust Region Constraint

Automating the generation of legible motion led us to a surprising observation: in some cases, *by optimizing the legibility functional, one can become arbitrarily unpredictable.*

Proof: Our gradient derivation in (6.8) enables us to construct cases in which this occurs. In a two-goal case like in Fig. 6.1, with our example C (Eq. 3.7), the gradient for each trajectory configuration points in the direction $G_R - G_O$ and has positive magnitude everywhere but at ∞, where $C[\xi] = \infty$. Fig. 6.10 (red) plots C across iterations. ■

THE REASON FOR THIS PECULIARITY is that the model for how observers make inferences in Eq. 3.11 *fails to capture how humans make inferences in highly unpredictable situations*. In reality, observers might get confused by the robot's behavior and stop reasoning about the robot's possible goals the way the model assumes they would — comparing the sub-optimality of its actions with respect to each of them. Instead, they might start believing that the robot is malfunctioning [1] or that it is not pursuing any of the goals — a "something else" hypothesis that is supported by our user studies in Section 6.3, which show that this belief significantly increases at higher C costs.

This complexity of action interpretation in humans, which is difficult to capture in a goal prediction model, can significantly affect the legibility of the generated trajectories in practice. Optimizing the

[1] E. Short, J. Hart, M. Vu, and B. Scassellati. No fair!! an interaction with a cheating robot. 2010

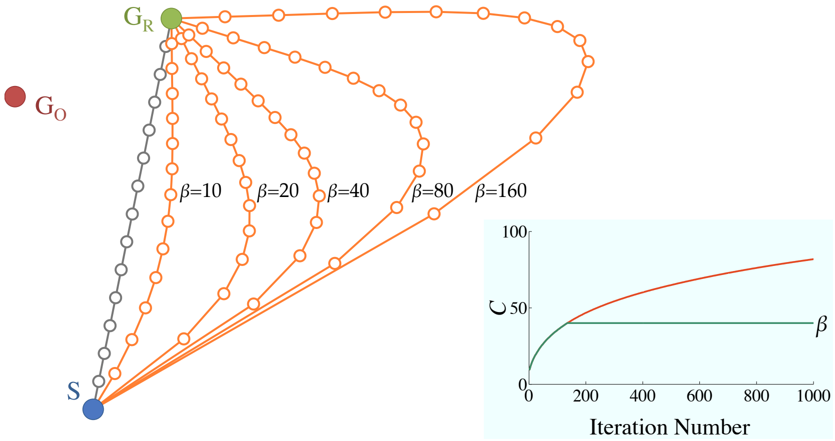


Figure 6.10: The expected (or predictable) trajectory in gray, and the legible trajectories for different trust region sizes in orange. On the right, the cost C over the iterations in the unconstrained case (red) and constrained case (green).

legibility score outside of a certain threshold for predictability can actually lower the legibility of the motion as measured with real users (as it does in our study in Section 6.3.2). Unpredictability above a certain level can also be detrimental to the collaboration process in general [8, 95, 167].

WE PROPOSE TO ADDRESS THESE ISSUES by only allowing optimization of legibility where the model holds, i.e., where predictability is sufficiently high. We call this a "trust region" of predictability — a constraint that bounds the domain of trajectories, but that does so w.r.t. the cost functional C, resulting in $C[\xi] \leq \beta$:

By constraining C, we constrain how large the probability of the "something else" hypothesis is allowed to become: a constraint on the trajectory imposes a constraint on any snippet of the ongoing trajectory, which, along with the prior value on "something else", induce a constraint on the probability mass of this hypothesis.

> *The legibility model can only be trusted inside this trust region.*

The parameter β, as our study will show, is identifiable by its effect on legibility as measured with users — the point at which further optimization of the legibility functional makes the trajectory less legible in practice.

We thus define a trust region of predictability, constraining the trajectory to stay below a maximum cost in C during the optimization:

$$\begin{aligned}\xi_{i+1} = \arg\max_{\xi}\ & \textsc{Legibility}[\xi_i] + \nabla\textsc{Legibility}^T(\xi - \xi_i) \\ & - \frac{\eta}{2}||\xi - \xi_i||^2_M \\ s.t.\ \ & C[\xi] \leq \beta \end{aligned} \tag{6.12}$$

This is the case of no obstacle avoidance. With obstacles, replace $\textsc{Legbility}$ with $\mathcal{U} = -\textsc{Legibility} + \alpha\mathcal{U}_{obs}$.

To solve this, we proceed analogously to Section 4.2: we first linearize the constraint, which now becomes $\nabla C^T(\xi - \xi_i) + C[\xi_i] \leq \beta$. The Lagrangian is

$$\begin{aligned}\mathcal{L}[\xi, \lambda] = & \textsc{Legibility}[\xi_i] + \bar{\nabla}\textsc{Legibility}^T(\xi - \xi_i) \\ & - \frac{\eta}{2}||\xi - \xi_i||^2_A + \lambda(\beta - \nabla C^T(\xi - \xi_i) - C[\xi_i])\end{aligned} \tag{6.13}$$

with the following KKT conditions:

$$\nabla\textsc{Legibility} - \eta A(\xi - \xi_i) - \bar{\nabla} C\lambda = 0 \tag{6.14}$$

$$\lambda(\beta - \nabla C^T(\xi - \xi_i) - C[\xi_i]) = 0 \tag{6.15}$$

$$\lambda \geq 0 \tag{6.16}$$

$$C[\xi] \leq \beta \tag{6.17}$$

Inactive constraint: $\lambda = 0$ and

$$\xi_{i+1} = \xi_i + \frac{1}{\eta} A^{-1} \nabla\textsc{Legibility} \tag{6.18}$$

Active constraint: The constraint becomes an equality constraint on the trajectory, for which the derivation for ξ_{i+1} is an instance of Eq. 4.16. From (6.14) we get

$$\xi_{i+1} = \xi_i + \frac{1}{\eta} A^{-1} \underbrace{(\nabla\textsc{Legibility} - \lambda\nabla C)}_{\nabla(\textsc{Legibility} - \lambda C)} \tag{6.19}$$

This is the functional gradient of Legibility with an additional (linear) *regularizer* λC *penalizing unpredictability*.

Substituting in (6.15) to get the value for λ and using (6.14) again, we obtain a new update rule:

$$\begin{aligned} \xi_{i+1} = \xi_i &+ \frac{1}{\eta} A^{-1} \nabla\textsc{Legibility} - \\ &\underbrace{\frac{1}{\eta} A^{-1} \nabla C (\nabla C^T A^{-1} \nabla C)^{-1} \nabla C^T A^{-1} \nabla\textsc{Legibility}}_{\text{projection on } \nabla C^T(\xi - \xi_i) = 0} - \\ &\underbrace{A^{-1} \nabla C (\nabla C^T A^{-1} \nabla C)^{-1} (C[\xi_i] - \beta)}_{\text{offset correction to } \nabla C^T(\xi - \xi_i) + C[\xi_i] = \beta} \end{aligned} \tag{6.20}$$

Fig. 6.10 shows the outcome of the optimization for various β values. In what follows, we discuss what effect β has on the legibility of the trajectory in practice, as measured through users observing the robot's motion.

6.3 *From Theory to Users*

Legibility is intrinsically a property that depends on the observer: a real user. In this section, we test our legibility motion planner, as well as our theoretical notion of a trust region, on users observing motion. If our assumptions are true, then by varying $\beta \in [\beta_{min}, \beta_{max}]$, we expect to find that an intermediate value β^* produces the most legible result: much lower than β^* and the trajectory does not depart predictability enough to convey intent, much higher and the trajectory becomes too unpredictable, confusing the users and thus actually having a negative impact on legibility.

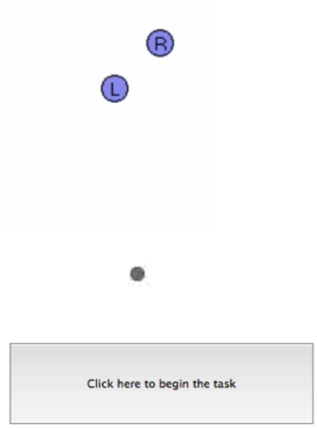
6.3.1 Main Experiment

Hypotheses.

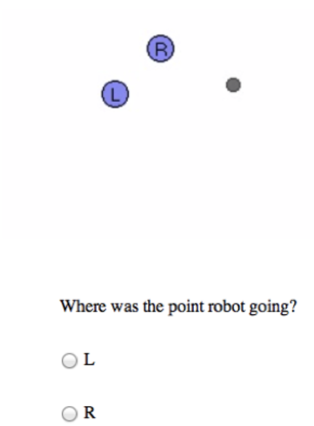
H1 *The size of the trust region, β, has a significant effect on legibility.*

H2 *Legibility will significantly increase with β at first, but start decreasing at some large enough β.*

Manipulated Variables. We manipulated β, selecting values that grow geometrically (with scalar 2) starting at 10 and ending at 320, a value we considered high enough to either support or contradict the expected effect. We also tested $\beta = \min_{\xi} C[\xi]$, which allows for no additional legibility and thus produces the predictable trajectory (we denote this as $\beta = 0$ for simplicity). We created optimal trajectories for each β in the scene from Fig. 6.11: a point robot reaching for one of two goals.



(a) Before



Where was the point robot going?

○ L

○ R

(b) After

Figure 6.11: We measure legibility by measuring at what time point along the trajectory users feel confident enough to provide a goal prediction, as well as whether the prediction is correct.

Dependent Measures. We measured the legibility of the seven trajectories. Our measurement method follows Section 3.4: we showed the users a video of the trajectory, and asked them to stop the video as soon as they felt confident in their prediction of which goal the robot is headed toward (Fig. 6.11). We recorded their goal prediction and the time from the start of the video to the point where they stopped it, and combined the two into a single metric based on the Guttman score [2]. Incorrect predictions received a score of 0, and correct ones received a linearly higher score when the response time was lower, i.e., when they became confident in the correct prediction earlier. We used slow videos (28s) to control for response time effects.

Subject Allocation. We chose a between-subjects design in order to not bias the users with trajectories from previous conditions. We recruited 320 participants through Amazon's Mechanical Turk service, and took several measures to ensure reliability of the results. All participants were located in the USA to avoid language barriers, and they all had an approval rate of over 95%. We asked all participants a control question that tested their attention to the task, and eliminated data associated with wrong answers to this question, as well as incomplete data, resulting in a total of 297 samples.

Analysis. An ANOVA using β as a factor supported H1, showing that the factor had a significant effect on legibility ($F(6, 290) = 12.57$, $p < 0.001$). Fig. 6.12(left) shows the means and standard errors for each condition.

An all-pairs post-hoc analysis with Tukey corrections for multiple comparisons revealed that all trajectories with $\beta \geq 20$ were significantly more legible than the predictable trajectory ($\beta = 0$), all with $p \leq 0.001$, the maximum being reached at $\beta = 40$ This supports the first part of H2, that legibility significantly increases with β at first: *there is no practical need to become more unpredictable beyond this point.*

Legibility did significantly increase with β at first, but there was no significant decrease after β^*.

---

[2] G.R. Bergersen, J.E. Hannay, D.I.K. Sjoberg, T. Dyba, and A. Karahasanovic. Inferring skill from tests of programming performance: Combining time and quality. In *ESEM*, 2011

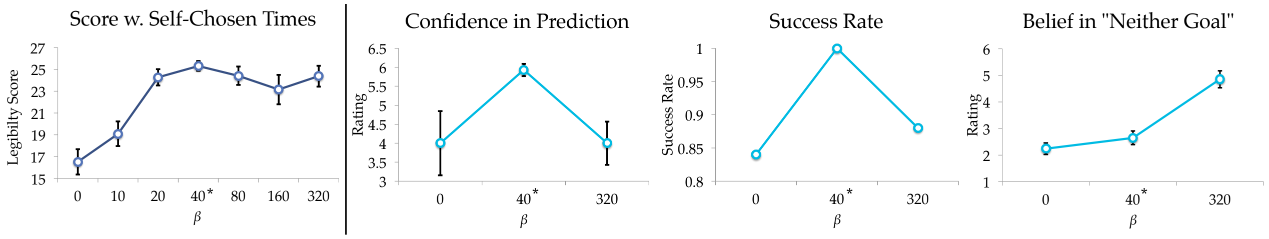


Figure 6.12: Left: The legibility score for all 7 conditions in our main experiment: as the trust region grows, the trajectory becomes more legible. However, beyond a certain trust region size ($\beta = 40$), we see no added benefit of legibility. Right: In a follow-up study, we showed users the entire first half of the trajectories, and asked them to predict the goal, rate their confidence, as well as their belief that the robot is heading towards neither goal. The results reinforce the need for a trust region.

The maximum mean legibility was the trajectory with $\beta = 40$. Beyond this value, the mean legibility stopped increasing. Contrary to our expectation, it did not significantly decrease. In fact, the difference in score between $\beta = 40$ and $\beta = 320$ is in fact significantly less than 2.81 ($t(84) = 1.67$, $p = 0.05$). At a first glance, the robot's overly unpredictable behavior seems to not have caused any confusion as to what its intent was.

Analyzing the score histograms (Fig. 6.13) for different β values, we observed that for the hight βs, users did not stop the trajectory in the middle: the guessed the goal in the beginning, or waited until the end. The consequence is that *our legibility measure failed to capture whether the mid-part of the trajectory becomes illegible*. Thus, we ran a follow-up study to verify that legibility in this region does decrease at $\beta = 320$ as compared to our $\beta^* = 40$.

6.3.2 *Follow-Up Study*

Our follow-up study was designed to investigate legibility during the middle of the trajectories. The setup was the same, but rather than allowing the users to set the time at which they provide an answer, we fixed the time and instead asked them for a prediction and a rating of their confidence on a Likert scale from 1 to 7. We hypothesize that in this case, the users' confidence (aggregated with success rate such that a wrong prediction with high confidence is treated negatively) will align with our H2: it will be higher for $\beta = 40$ than for $\beta = 320$.

We ran a follow-up study to test legibility at a fixed time point, rather than at the point where each user feels confident as we did before.

We conducted this study with 90 users. Fig. 6.12 plots the confidences and success rates, showing that they are higher for $\beta = 40$ than they are for both of the extremes, 0 and 320. An ANOVA confirmed that the confidence effect was significant ($F(2, 84) = 3.64$, $p = 0.03$). The post-hoc analysis confirmed that $\beta = 40$ had significantly higher confidence $t(57) = 2.43$, $p = 0.45$.

Legibility did decrease for $\beta > \beta^*$, with participants starting to infer a "something else" hypothesis.

We also asked the users to what extent they believed that the robot was going for neither of the goals depicted in the scene (also Fig. 6.12). In an analogous analysis, we found that users in the $\beta = 40$

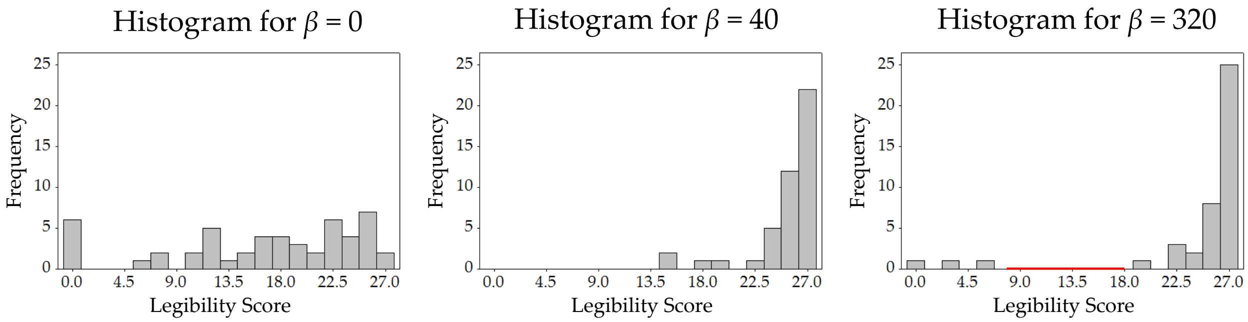


Figure 6.13: The distribution of scores for three of the conditions. With a very large trust region, even though the legibility score does not significantly decrease, the users either infer the goal very quickly, or they wait until the end of the trajectory, suggesting a legibility issue with the middle portion of the trajectory.

condition believed this significantly less than users in the $\beta = 320$ condition ($t(57) = 5.7$, $p < 0.001$).

In summary, the results support the existence of *a trust region of expectation within which legibility optimization can make trajectories significantly more legible* to novice users. Outside of this trust region, being more legible w.r.t. Legibility an impractical quest, because it no longer improves legibility in practice. Furthermore, the unpredictability of the trajectory can actually confuse the observer enough that they can no longer accurately and confidently predict the goal, and perhaps even doubt that they have the right understanding of how the robot behaves. They start believing in a "neither goal" option that is not present in the scene. *Indeed, the legibility formalism can only be trusted within this trust region.*

Limitations. The need for a trust region is limiting. First, it only approximates the true objective of drawing Bayesian inference on the “something else hypothesis”. Second, even if it were an exact inference, it would still depend on a parameter (β, or the prior on the actual goals in the scene) which needs to be tuned or learned.

Furthermore, there are limitations to how legible the robot can be. As scenes become more and more complex, optimizing for legibility starts having little advantage over being predictable. On the positive side, our formalism can quantify how legible the robot can be in any given task, and even enable sequencing the goals in the most legible way.

Additionally, as we saw in Section 5.3, an observer’s expectations change over time: C changes, which in turns changes Legibility. Further analysis is needed to understand these effects.

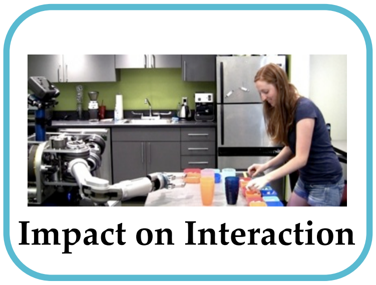
6.4 Chapter Summary

In this chapter, we introduced a functional gradient descent optimization algorithm for generating legible motion. Strategies from animation, like exaggeration, emerged out of the optimization without the need to pre-specify them. We also showed that the optimization can be unbounded, and that a trust region constraint is useful in practice for enabling robots to best take advantage of the legibility formalism.

legibility optimization
↓
exaggeration naturally emerges
↓
trust region constrains over-exaggeration
↓
user study

7

User Study on Physical Collaboration



So far, our user studies tested whether the robot can produce motion that is more predictable or more legible. Here, we put these planners, along with a functional motion planner, in the context of a real *physical* collaboration in order to test whether the predictability and legibility improvements *ultimately affect the collaboration fluency*.

We use a task that requires *coordinating* [156] with the robot (by inferring its goals and performing complementary actions), and study how the choice of a planner affects the fluency of the collaboration through both objective and subjective measures inspired by prior work on fluency [1].

We designed a study ($N = 18$) with objective measures, like the time it takes for participants to infer their action based on the robot's goal (coordination time), how efficient they are at the task (total task time), and how much they move while the robot is moving (concurrent motion), and subjective measures, like how participants perceive the collaboration in terms of fluency, comfort, trust, etc.

7.1 Motions

We plan predictable and legible motion as described in Sections 5.1 and 6.1. We plan functional motion using a bi-directional RRT [135].

Functional. Fig. 7.2 (left) shows the end effector trace of a functional motion plan to grasp the object on the right. Fig. 7.1 (left) shows a snapshot of the motion, along with a participant's reaction to it. The motion is not efficient, puts the robot in unnatural configurations, and can at times be deceptive about the robot's goal — it might seem like the goal is the one of the left until the very end of the motion.

Thus, we expect that people who collaborate with a robot that produces such motion will not be comfortable, and will not be able to coordinate with the robot because of the difficulty in inferring what

[1] G Hoffman. Evaluating fluency in human-robot collaboration. In *HRI Workshop on Human Robot Collaboration*, 2013

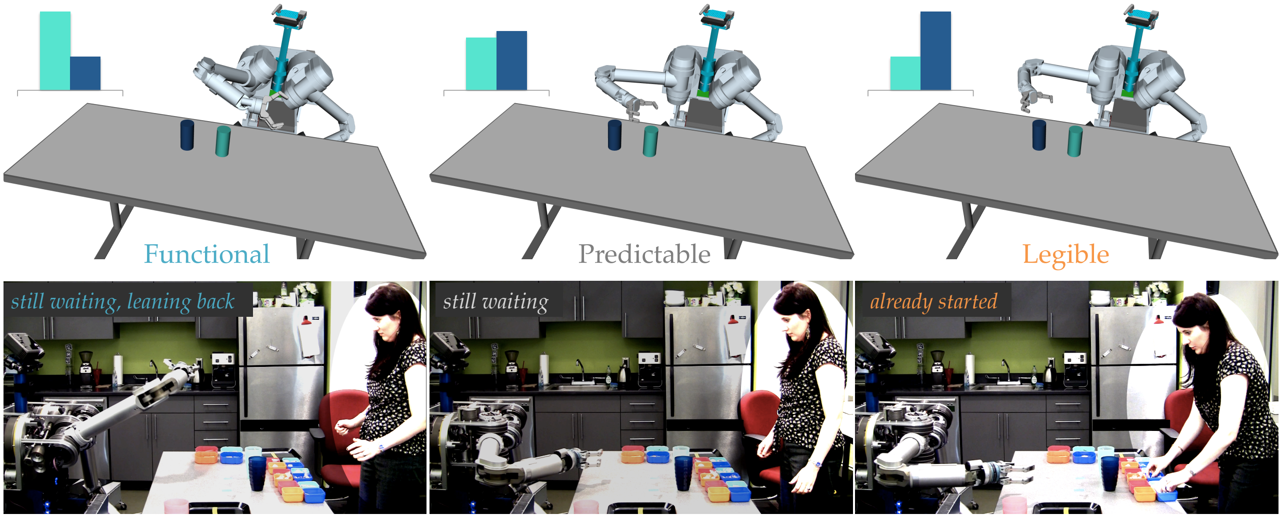


Figure 7.1: Snapshots from the three types of motion at the same time point along the trajectory. The robot is reaching for the dark blue cup. The functional motion is erratic and somewhat deceptive, and the participant leans back and waits before committing to a color. The predictable motion is efficient, but ambiguous, and the participant is still not willing to commit. The legible motion makes the intent more clear, and the participant is confident enough to start the task.

the robot is doing.

Predictable. Fig. 7.2 (center) shows the end effector trace of a predictable motion plan, a snapshot of which is in Fig. 7.1 (center). This motion is efficient, but it can be ambiguous about the robot's goal, making it difficult to infer its intent. This is especially true in the beginning of the motion, when the predictable trajectory to the goal on the right is very similar to what the predictable trajectory to the goal on the left would look like. The participant in Fig. 7.1 is still waiting to be confident about the robot's intent.

Because predictable motion matches what people expect, we anticipate that people who collaborate with a robot that produces predictable motion will be more comfortable than with functional motion, and better able to coordinate with the robot. However, we expect ambiguous situations to lead to difficulties in coordination, caused by the inability to quickly infer the robot's intent.

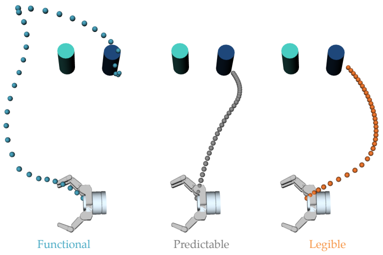


Figure 7.2: The end effector traces of the three types of motion for one part of the task.

Legible. Fig. 7.2 (right) shows the end effector trace of a legible motion plan, a snapshot of which is in Fig. 7.1 (right). This motion is less efficient than the predictable one (slightly more unpredictable), but, by exaggerating the motion to the right, it more clearly conveys that the actual goal is the one on the right. The participant in Fig. 7.1 already knows the robot's goal and has started her part of the task in response.

We expect that the benefit of clearly conveying intent will make legible motion better for collaboration than both predictable and functional motion. However, predictable motion is already much better at conveying intent than functional motion is. It is also more predictable (by definition) than legible motion. Together, this can imply a more subtle difference when going from predictability to legibility, than when going from functionality to predictability.

7.2 Hypotheses

As the predictions in the previous section suggest, we anticipate that the type of motion the robot plans will affect the collaboration both objectively and subjectively. We also expect it to affect participants' perceptions of how predictable and legible the motions are.

H1 - Objective Collaboration Metrics. *Motion type will positively affect the collaboration objectively, with legible motion being the best, and functional motion being the worst.*

objective effects

H2 - Perceptions of the Collaboration. *Motion type will positively affect the participants' perception of the collaboration, with legible motion being the best, and functional motion being the worst.*

subjective effects

H3 - Perceptions of Legibility and Predictability. *Participants will rate the legible motion as more legible than the predictable motion, and the predictable motion as more legible than the functional motion. In contrast, participants will rate the predictable motion as more predictable than the legible motion, and the legible motion as more predictable than the functional motion.*

perceptions about the motion itself

7.3 Experimental Design

To explore the effect of motion type on human-robot collaboration, we conducted a counterbalanced within-subjects study in which participants collaborated on a task with HERB.

7.3.1 Task

Challenges. Designing a human-robot collaborative task for comparing these types of motion was challenging for four reasons.

First, the success of a collaboration depends on more than the type of robot motion. Other errors during the collaboration can drastically affect the findings. Therefore, the task needs to emphasize the role of motion.

Challenge 1: restrict task to motion.

Second, since the study is not testing how the robot should respond to the human's motion, the human's action needs to depend on the robot's, but not vice-versa.

Challenge 2: robot cannot react to the user.

Third, the task must be repeatable: each participant must face the exact same motion planning situations. Different situations (e.g.,

Challenge 3: plan the same motions across participants.

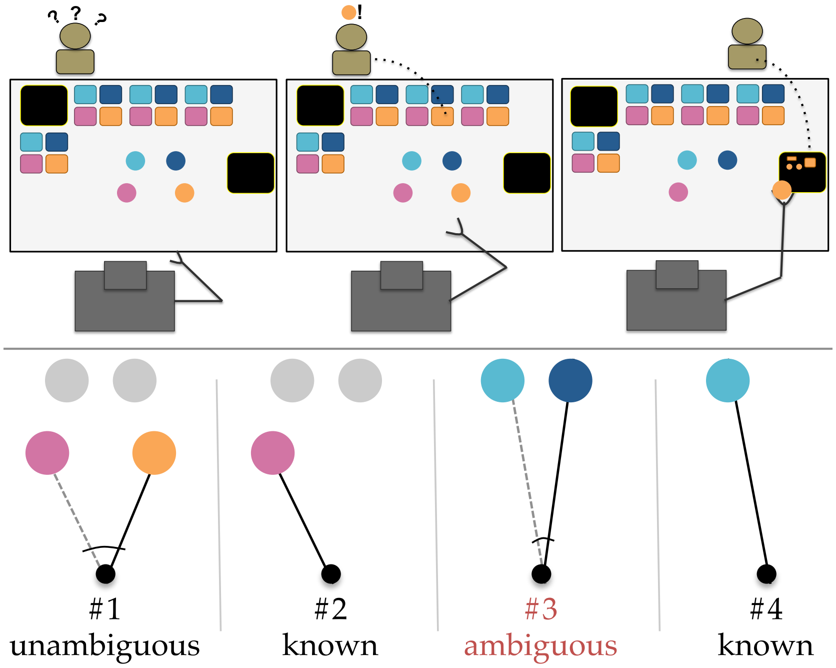


Figure 7.3: For each tea order, the robot starts reaching for one of the cups. The participant infers the robot's goal and starts gathering the corresponding ingredients. Both place their items on the tray, and move on to the next order. For order #3, the cups are further away from the robot, and closer to each other, making the situation ambiguous.

an object being at a slightly different location) can result in vastly different motions in the case of the functional planner, which could lead to a confound.

And fourth, the task should be as realistic as possible to the participants, and simulate a real world collaboration.

Challenge 4: quasi-realistic scenario.

To SATISFY THESE FOUR CONSTRAINTS, the task followed a coffeeshop scenario, in which participants work together with the robot to collaboratively fulfill tea orders. The robot retrieves the correct cup, and the participant gathers the ingredients. Key to this task was that *the selection of the ingredients depends on which cup the robot is retrieving*.

The human's task depends on what the robot is doing.

Fig. 7.3 shows a schematic of the task setup. There are four orders total, and four different-colored cups. For each order, the robot reaches for one of the cups, and the participant tries to infer the correct color and starts getting the corresponding ingredients from color-coded bins. This emphasizes the role of motion; it does not require that the robot respond to the human; and it leads to a repeatable task because the location of the cups and the order in which the robot picks them up can be predetermined.

The experiment required the participant to fulfill four orders consecutively instead of a single one because (1) this structure places participants in a longer interaction, and (2) it gives participants a chance to familiarize to the motion type. The four orders split into groups of two, as in Fig. 7.3: participants know that the the first two cups the robot reaches for are in the front, and the next two are in the back. Thus, participants do not know the robot's goal a-priori for the first and third order.

The cups are placed such that the situation corresponding to the first order is unambiguous — the cups are far enough apart that the predictable motion should be sufficient to convey the goal early on. The test situation is really the third order, which is ambiguous and thus the best at identifying the differences among the three planners. Furthermore, there is not a strong surprise factor, as each participant will have already seen the robot fulfill two orders.

7.3.2 Procedure

Participants entered the lab and following informed consent, were administered a pre-study questionnaire. Next, the experimenter explained the collaborative task and informed participants that three "programs" were being tested for the robot. They practiced the task once, after which they performed the task three times, one with each "program" (motion type). After each task, they took notes about the collaboration with the robot. At the end, they were administered a post-study questionnaire, and asked to describe the three programs they had experienced.

7.3.3 Manipulated Variables

We manipulated a single variable, *motion type*, to be functional, predictable, or legible. Since the functional planner is nondeterministic, committing to a particular trajectory for each situation is a nontrivial decision. We did so by generating a small set of trajectories and selecting the trajectory with the smallest legibility score. This emphasizes situations where functional motion accidentally leads to deceptive paths, which can harm coordination.

We controlled for by imposing the same duration for all trajectories.

7.3.4 Participant Assignment Method

We recruited a total of 18 participants (5 males, 13 females, aged $18 - 61$, $M = 29.17$, $SD = 12.50$) from the local community. Only five of the participants reported having a technical background.

The experiment used a within-subjects design because it enables participants to compare the three motions. Participants were told that there were three different robot "programs" to avoid biasing them towards explicitly looking for differences in the motion itself.

We fully counterbalanced the order of the conditions to control for order effects. We used a practice round to eliminate some of the variance introduced by the novelty effect. During the practice round, the robot moved predictably, helping to set the predictable motion as their expectation.

The three test rounds (with the three motion types) used the same ordering of the cups, while the practice round used a different ordering. This way, participants would know that the ordering is not set, while allowing for the ability to eliminate cup order as a confound. A single participant noticed the repeating pattern, as detailed in the Analysis section.

7.3.5 *Dependent Measures*

The measures capture the success of a collaboration in both *objective* and *subjective* ways, and are based on Hoffman's metrics for fluency in human-robot collaborations [98].

OBJECTIVE MEASURES include the *coordination time*, the *total task time*, and the *concurrent motion time* for the test order (order #3).

The coordination time is the amount of time from the moment the robot starts moving, until the participant infers the correct goal (either by declaring it aloud, which we ask participants to do, or by starting to reach for the correct ingredients, whichever comes first). The total task time is the amount of time, from the moment the robot starts moving, until the last ingredient touches the tray. Finally, the concurrent motion time is the amount of time when both the human and the robot are moving.

coordination time: time to infer the goal

task time: time to complete the human part of the task

concurrent motion: time when the human and robot are both moving

Table 7.1 shows the seven subjective scales that we used, together with a few forced-choice questions. The *fluency* and *trust* scales were used as-is from [98]. The *robot contribution* scale was shortened to avoid asking participants too many questions. A subset of questions were chosen related to *capability*, and extended questions were chosen related to *safety/comfort*. We added additional questions were added that were more appropriate to the physical setup (feeling safe next to the robot, and being confident that the robot can avoid collisions with them).

we used scales for fluency, trust, robot contribution, capability, and safety/confort, as well as closeness

The *closeness* to the robot question from [163] (not shown in the table) asked participants to select among five diagrams portraying different levels of mental proximity to the robot during the task.

Additionally, participants answered forced-choice questions at the end, about which program they were the fastest with, which program

Table 7.1: Subjective measures.

Fluency $\alpha = .91$
1.The human-robot team worked fluently together.
2.The robot contributed to the fluency of the team interaction .
Robot Contribution [shortened] $\alpha = .75$
1.I had to carry the weight to make the human-robot team better.(r)
2.The robot contributed equally to the team performance.
3.The robot's performance was an important contribution to the success of the team.
Trust $\alpha = .91$
1.I trusted the robot to do the right thing at the right time.
2.The robot was trustworthy.
3.The robot and I trust each other.
Safety/Comfort [extended] $\alpha = .83$
1.I feel uncomfortable with the robot.(r)
2.I believe the robot likes me.
3.I feel safe working next to the robot. [new]
4.I am confident the robot will not hit me as it is moving. [new]
Capability $\alpha = .72$
1.I am confident in the robot's ability to help me.
2.The robot is intelligent.
Predictability [re-phrased for clarity] $\alpha = .86$
1.If I were told what cup the robot was going to reach for ahead of time, I would be able to correctly anticipate the robot's reaching motion.
2.The robot's reaching motion matched what I would have expected given the cup it was reaching for.
3.The robot's reaching motion was surprising.(r)
Legibility [new] $\alpha = .95$
1.The robot can reason about how to make it easier for me to predict what it is reaching for.
2.It was easy to predict what the robot was reaching for.
3.The robot moved in a manner that made its intention clear.
4.The robot was trying to move in a way that helped me figure out what it was reaching for.
Forced-Choice Questions $\alpha = .91$
1.Which program were you the fastest with?
2.Which program was the easiest?
3.Which program do you prefer?

was easiest to work with, and which program they preferred.

The subjective measures also included perceived *predictability* and *legibility*. The predictability scale was adapted from Section 5.3. For this experiment, we added clarifications because the task was so focused on predicting goals that the word “predictable” was too easily misunderstood in this context.

We devised a legibility scale to capture both how easy inferring the goal is, as well as whether participants believe that the robot has the ability to reason about making this inference easy, and whether it was explicitly trying to do so.

In addition to these measures, we administered a pre-survey to participants, asking demographics questions, as well as the "Big-5" personality questionnaire, since personality type could potentially correlate with how they experience the collaboration.

Finally, we adapted the service orientation *attitude* scale [2], measuring whether participants have a relational or utilitarian orientation toward a food service provider. The questions were modified to refer to food preparation. We chose this measure because having a relational attitude could correlate with the way participants interpret legibility, in particular whether they think the robot is *purposefully* trying to help them infer the goal easier.

7.4 Analysis

Each of the 18 participants performed the task three times, with each task consisting of four orders (trials). This led to a total of 216 trials, out of which 54 were test trials (order #3), 54 were unambiguous trials (order #1) that still had a coordination time, and the rest were trials that did not need coordination.

7.4.1 H1 - Objective Measures

A repeated measures ANOVA on the *coordination time* ($R^2 = .67$) showed a significant effect for motion type ($F(2,51) = 52.06$, $p < .0001$), in line with **H1**.

Robot motion does affect collaboration.

A post-hoc analysis with Tukey HSD supported **H1**, showing that all three conditions were significantly different from each other, with functional taking significantly longer than predictable ($p < .0001$), and predictable taking significantly longer than legible ($p = .01$). Legible motion resulted in a 33% decrease in coordination time compared to predictable motion.[3]

Fig. 7.4 shows a scatter plot of the coordination time by the total task time. As expected, legible motion < predictable motion < functional motion in terms of coordination time, with functional motion being better separated as a cluster. These differences propagate to the total task time.

There is one outlier in the plot, for the functional motion (the blue circle in the center left). This was a participant who noticed a repeat-

---

[2] Min Kyung Lee, Sara Kiesler, Jodi Forlizzi, Siddhartha Srinivasa, and Paul Rybski. Gracefully mitigating breakdowns in robotic services. In *HRI*, 2010

[3] These results are for the test trials. There was no difference between legibility and predictability on the unambiguous trials (Fig. 7.3), since the predictable motion is sufficiently legible when there is little ambiguity.

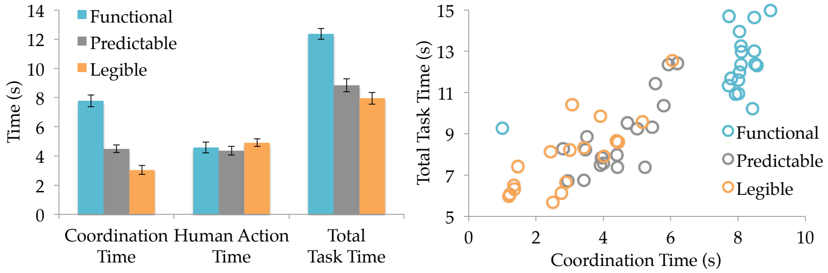


Figure 7.4: Findings for objective measures.

ing pattern in the ordering of the cups, and achieved minimal coordination time as a result during his third condition, which happened to be the functional condition.

A repeated measures ANOVA on the *total task time* ($R^2 = .56$) showed similar results. Motion type was significant ($F(2, 51) = 32.59$, $p < .0001$), and the post-hoc showed a significant difference between predictable and functional ($p < .0001$), partially supporting **H1**.

However, the difference between predictable and legible, although trending in the expected direction (Fig. 7.4 bottom center), was no longer significant ($p = .27$). Surprisingly, participants took slightly longer to gather the ingredients in the legible condition ("human action time", Fig. 7.4 bottom center). Analysis of the video recordings showed that even though some participants could infer the correct cup earlier, they would *hesitate* a bit during the task, looking back at the robot again to make sure they made the right prediction and thus slowing down.

Surprisingly, participants did not wait for the robot to finish moving in the functional condition, as we had anticipated. Instead, participants were comfortable enough to do the task while the robot was still moving. Since the robot took longer than the participants to achieve its part of the task, the concurrent motion time was equal to the human action time and did not provide any additional insight.

Participants' main complaint about the functional motion was that it was difficult to coordinate with the robot, and not that they felt unsafe. This could potentially be the result of placing participants in a lab setting, leading to them over-trusting the robot.

Some of the participants did lean back more, as if to avoid the robot arm, and also took a curved path to place the ingredients on the tray (see Fig. 7.5 for an example). Many participants looked surprised when the robot started moving. However, there were some who remained completely unphased by the motion.

Because of the delay in inferring the correct cup, a participant exclaimed "Wait for me!" as she was hurrying to catch up because

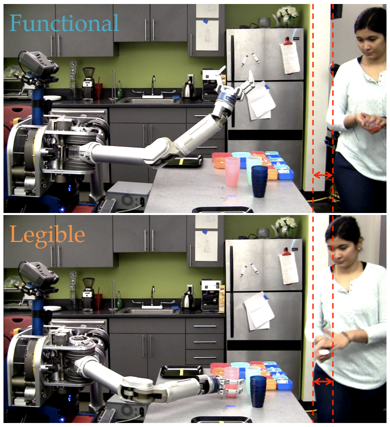


Figure 7.5: Some of the participants kept a larger distance to the robot during the functional condition. However, most participants were surprisingly comfortable with the robot during this condition.

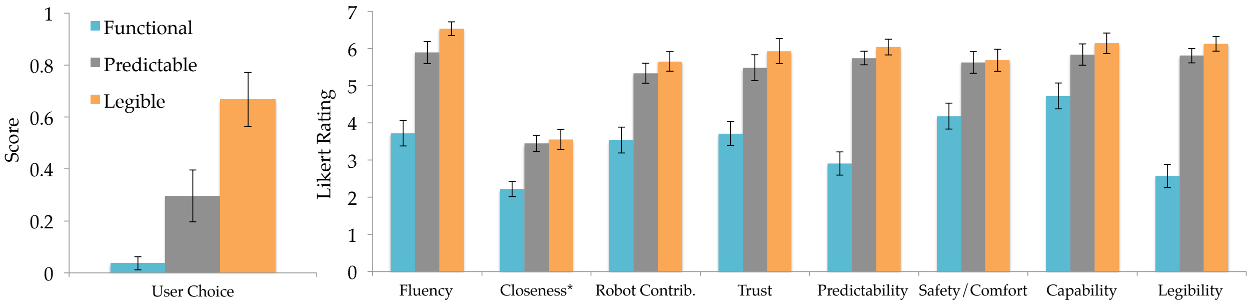


Figure 7.6: Findings for subjective measures. Closeness was on a 5-point scale.

of the long coordination time. Some of the participants would speed up in gathering the ingredients in the functional condition, as if they were trying to catch up to the robot and still finish the task before. This was not the case in general, with some of the participants having a longer action time than in the predictable condition, stopping more to watch the robot, and hesitating in gathering the ingredients.

None of the participants complained about the robot being much slower than them. This could be due to the bias of participating in a lab experiment. However, as the "Wait for me!" complaint suggests, participants seemed to actually mind the robot finishing its part of the task before they finished theirs, emphasizing the importance of synchronization in collaboration tasks.

OVERALL, supporting **H1**, legible motion had significantly lower coordination time than predictable, which had significantly lower coordination time than legible. 17 out of 18 participants had lower coordination time with the legible motion compared to predictable, and 15 had a lower total task time. As expected, the difference between legibility and predictability was more subtle than that between predictability and pure functionality. Surprisingly, the robot moving functionally did not affect concurrent motion time, and participants were comfortable enough to move at the same time as the robot even with functional motion.

7.4.2 H2 - Perceptions of the Collaboration

Table 7.1, which lists the subjective scales, also shows the internal consistency of each scale, reported via Cronbach's α. Most scales had good to excellent consistency, the exceptions being *capability* and *robot contribution*, which were acceptable. Scale items were combined into a score and analyzed with repeated-measures ANOVAs. Fig. 7.6 plots the results.

The score produced by the overall forced-choice questions was significantly affected by the motion type ($F(2,51) = 13.59$, $p < .0001$), with the post-hoc revealing that legible motion had a significantly higher score than predictable motion ($p < .01$), but predictable motion was only marginally better than functional motion ($p = .08$). 12 out of the 18 participants preferred the legible motion.

All the Likert ratings showed a significant effect for motion type as well, with post-hocs revealing that functional motion was significantly lower rated than predictable and legible motion in every case (with $p < .0001$, except for *capability*, details below). The legible motion tended to be rated higher than predictable, but those differences were not significant. Fig. 7.6 summarizes these findings.

The biggest difference between predictability and legibility was in *fluency*. *Safety*, on the other hand, was the same for both — this is not surprising, given that legible motion is better at conveying intent, but this does not necessarily lead to an increased feeling of safety.

Capability was high with the functional motion as well, though still significantly lower than with predictable motion ($p = .03$).

With respect to additional participant measures, unsurprisingly, being extroverted significantly correlated to having a relation attitude towards a food preparation partner ($r(16) = .51$, $p = .03$). Additionally, extroversion inversely correlated with preferring the legible motion over the other two motion types ($r(16) = -.49$, $p = .04$). However, extroversion did not correlate with whether or not the legible motion worked objectively, i.e., achieved lower coordination time. More research is needed to verify this result and understand why introverts might be more likely to appreciate a legible robot.

Overall, participants significantly preferred the legible motion over the predictable motion, and tended to prefer the predictable motion over the functional. However, as with the objective measures, their ratings of the collaboration suggest that legibility is a more subtle improvement over predictability, compared to the improvement of predictability over functionality.

7.4.3 *H3 - Perceptions of Predictability and Legibility: Rationalization of the Motion*

Perceptions of Legibility. As predicted by **H3**, motion type significantly affected the legibility rating ($F(2,51) = 67.56$, $p < .0001$). The post-hoc analysis did show a significant difference between functional and predictable motion ($p < .0001$), but not between predictable and legible motion.

The biggest difference between predictable and legible motion was in how easy participants thought it was to predict the robot's goal (question 2) (mean 6 vs. 6.61). Participants thought the legible motion made goal inference easier. In contrast, participants did not think that the robot was more capable of higher-order reasoning. Question 1 yielded almost no difference between predictability and legibility, and had a lower overall mean (5.11 vs. 5.27).

PARTICIPANTS' COMMENTS matched their ratings of legibility of motion. Three participants described the functional motion as "exaggerated", with one of them commenting that "the arm motions were so exaggerated that it was hard to see which cup he was going to choose until just before". Many of the participants referred to it as less intent-expressive, commenting that "it made it almost impossible to guess" or that it was "trickier".

One participant said that the functional motion made her less confident about the intent even for the orders where the cup was predetermined (2nd and 4th): "even when I knew the cup it would grab, I was still less confident than with the other programs". Indeed, we noticed some participants hesitate more during the functional motion condition on these orders, while others remained completely focused and ignored the erratic nature of the motion.

Interestingly, some participants attributed *agency* to the random nature of the functional motion: "he was picking a cup at random", "the robot appeared to be searching before selecting a cup", "makes me think that it's playing on purpose", "it appeared that the robot had a mind of its own, along with its own agenda", the robot "tricked me". One participant actually rated the functional program as the one they prefer overall, and a couple rated it as the most intelligent of the three, possibly because of this attribution of agency.

BECAUSE THE PREDICTABLE AND LEGIBLE MOTIONS ARE MORE SIMILAR to each other than they are to the functional motion, participants tended to contrast the two in their descriptions of the three programs.

Most participants described the predictable motion as somewhat less intent-expressive than the legible: "slightly harder to recognize", "the direction it's going in isn't as clear as the (legible motion)", "slight uncertainty about the cup choice", "not very clear as the (legible motion)", "not as easy as (the legible motion); I had to wait a bit after his hand moved to realize the cup he was going for", "it was had to determine which he'd pick", "it was not as clear".

In contrast, the descriptions for the legible motion referred to it as "easier to predict [the cup]" and "very straightforward", noting that one "could clearly see the trajectory of its hand to the cup". Some of the participants recognized that the robot was altering the motion in order to better convey intent. They thought that "the wide movements made it easy to identify [the cup]", "the angle was such that you could discern", and that "he starts out clearly moving towards one direction".

One of the participants even associated the beginning of the robot's legible motion to a communicative gesture: "it was almost like the robot was pointing at the cup he was going for right before, while he was moving his arm".

Perceptions of Predictability. Motion type significantly affected the predictability rating as well ($F = 50.48$, $p < .0001$). Counter to **H3**, however, participants actually tended to rate the legible motion higher, and the ratings for predictability and legibility significantly correlated ($r(52) = .91$, $p < .0001$).

It appears that when legibility works for someone and they can infer the goal easier, they tend to *rationalize* it as the "natural" motion, or even "direct" or "efficient". In contrast, some participants refer to the predictable motion as "inefficient", and even as "going towards the other cup initially", which is inaccurate.

This rationalization may happen because of the importance of inferring intent in the task. Legible motion is easier for collaboration, and that makes participants believe it is what they would have expected.

Note that, as shown by the second follow-up study in Section 3.4, it is not the case that users perceive the legible motion as more predictable when they do not collaborate, if they have seen an example of predictable motion a-priori. This was the case with our participants, who have seen a practice round of predictable motion. In such cases, 70% of users perceive the predictable motion as indeed more predictable when they do not collaborate with the robot and have no need for coordination.

In summary, the results do largely support our hypotheses, with the exception of how people perceive predictability: participants rationalized the legible motion as also being more predictable/efficient.

Limitations. This was a narrowly-scoped study, with a task chosen to emphasize the role of motion. There are certainly many other aspects of collaboration that are important, including (but not limited to) other channels of communication. Furthermore, to run well-controlled study, we had a contrived task that is not as realistic as we had hoped.

Our study also included a task with only four orders, whereas in real situations humans and robots will have prolonged interactions over many tasks. As a result, humans will adapt to robot motion and the need for legibility will decrease to a certain extent. On the other hand, even when motion is perfectly predictable, there are inherently ambiguous situations. An example of this is human motion: although a human's motion is perfectly predictable to another human, we still change the way we move and use exaggeration in collaborations [4].

7.5 Chapter Summary

We conducted a study that puts functional, predictable, and legible motions in the context of a real physical collaboration. We found that the legible motion was significantly better for collaboration than the predictable motion, and the predictable motion was significantly better than the functional motion. The difference between predictable and legible was more subtle than the difference between predictable and functional.

legible > predictable >> functional

users preferred the legible motion

users perceived the legible motion also as more predictable

The findings from this study suggest that functional motion is not enough for collaborative tasks that require coordination, and that the robot should take the collaborator's expectations into account when planning motion. Although this was a laboratory study with an artificial task, the findings lead to interesting conjectures about motion design for collaborative tasks.

One finding is that legibility is preferable to predictability in coordination tasks, as it decreases coordination time, collaborators prefer it overall, and rationalize it as more predictable despite it actually being less efficient (and them not being able to anticipate it a-priori). Furthermore, for quadratic costs C, legibility has no computational overhead compared to predictability in planning time.

Furthermore, functional motion might be enough for tasks that do not require coordination nor close proximity (such as repetitive tasks like those one might encounter on a factory floor, or tasks that have been carefully planned in advance, with separate and known roles). Participants were surprisingly willing to move at the same time as the robot, and mainly complained about not being able to coordinate.

Predictable motion seems to be best when coordination is not necessary (or the situations are not ambiguous, making the predictable motion legible enough), but when people work in close proximity to the robot and would be uncomfortable with surprising motion.

[4] Giovanni Pezzulo, Francesco Donnarumma, and Haris Dindo. Human sensorimotor communication: a theory of signaling in online social interactions. *PloS one*, 8(11):e79876, 2013

8

Generalizations of Legibility

This chapter presents generalizations of the legibility formalism from Chapter 3 to different situations, tasks, and channels of communication.

8.1 Viewpoint, Occlusion, Other DOFs

In this section, we focus on goal-directed legible motion beyond the situations from Chapter 6.

Effects of Observer Viewpoint. So far, we have seen the robot exaggerate the motion to the left or right to convey the goal on the left or on the right. This works very well when the observer is across from the robot, like in Fig. 8.1 - Viewpoint 1. But the same trajectory is no longer very legible when the observer is side by side with the robot (Viewpoint 2).

Project led by Stefanos Nikolaidis.

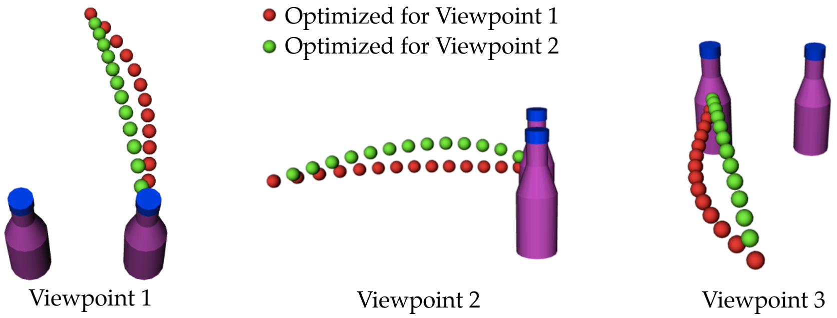


Figure 8.1: The red trajectory works in viewpoint 1, but is not as legible in viewpoint 2. The robot finds a different way to exaggerate when the observer has a different viewpoint (green trajectory). From viewpoint 2, it looks like the robot is exaggerating more, but that is not the case (see green trajectory in viewpoint 1). The two trajectories have the same cost C, but exaggerate in different directions (see viewpoint 3).

By defining the cost C in the observer's viewpoint (i.e., using $C(T(\xi))$ instead of $C(\xi)$, where T is a transformation that projects the trajectory onto the camera plane of the observer), the robot can generate a trajectory that is legible not to an omniscient observer, but to that particular observer. The robot exaggerates the trajectory in a different direction for Viewpoint 2.

Effects of Occlusion. Sometimes, there are occlusions which prevent the observer from some portions of the trajectory. Our formalism treats portions that cannot be observer the same as portions that happen in the future: we model the observer as integrating over all possible options, as we did in Eq. 3.13. At a waypoint that is occluded, the observer does not know the current configuration $\xi(t)$, leading to an additional integral over the occluded region.

Taking occlusions into account, the robot comes up with interesting strategies, like the one in Fig. 8.2: the robot "realizes" that it does not need to exaggerate the trajectory while occluded, and it can exaggerate more (given the same constraint β on C) when the observer can actually perceive the trajectory.

Project led by Stefanos Nikolaidis.

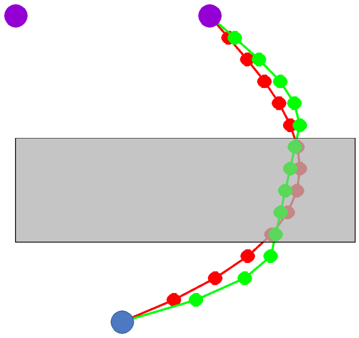


Figure 8.2: The robot does not exaggerate in the occluded region, so that it can exaggerate more outside of it.

Using other DOFs. Motion trajectories are not restricted to the arm degrees of freedom. Legibility optimization can also be applied over the hand DOFs, leading again to the robot moving its fingers inefficiently, but in a way that better conveys intent, as in Figures 8.3 and 8.4.

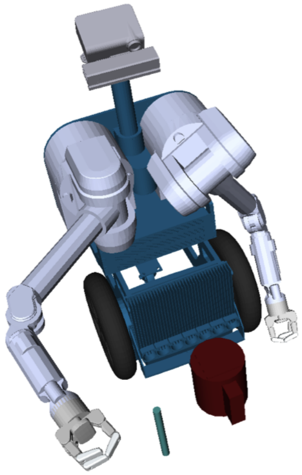


Figure 8.3: The robot uses a smaller than needed hand aperture to convey that it will grasp the smaller object.

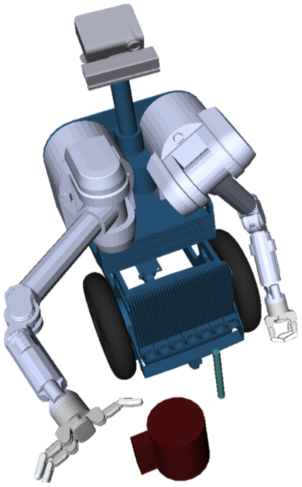


Figure 8.4: The robot uses a larger than needed hand aperture to convey that it will grasp the larger object.

We could have handcoded each of these strategies, but we did not need to. The same formalism can generate the different motions and strategies for all three contexts from above:

The mathematics of legibility leads to generalization.

8.2 Deception

The formalism for legible motion targets effective communication. But effective communication, which clearly conveys truthful information, has a natural counterpart: effective *deception*, which clearly conveys false information, or hides information altogether.

Robotic deception has obvious applications in the military [53], but its uses go far beyond. At its core, deception conveys *intentionality* [213], and that the robot has a *theory of mind* for the deceived [27] which it can use to manipulate their beliefs. It makes interactions with robots more engaging, particularly during game scenarios [223, 213, 197].

Deceptive motion is an integral part of being an opponent in most sports, like squash [74], soccer [201], or rugby [106]. It can also find uses outside of competitions, such as tricking patients into exerting more force during physical therapy [31].

Furthermore, a robot that can generate deceptive motion also has the ability to quantify an accidental leakage of deception and therefore avoid *deceiving accidentally.* As the study in Chapter 7 revealed, users sometimes think of the functional and even the predictable motion as being deceptive.

8.2.1 *Deceptive Motion Strategies*

We can use our legibility formalism to generate three different deception strategies, chosen based on our studies on how humans deceive [1]: exaggeration (decoy), switching, and ambiguity. The resulting motions are in Fig. 8.5.

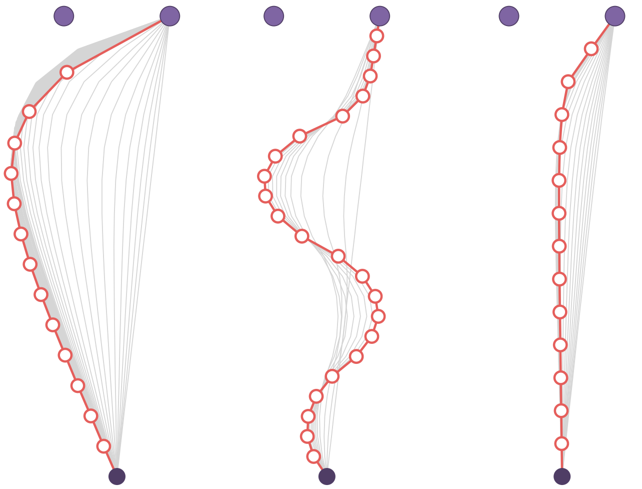

(a) Exaggerating (b) Switching (c) Ambiguous

Figure 8.5: Strategies replicated by the model: the typical exaggeration towards another goal, as well as the switching and ambiguous trajectories. The trajectories in gray show the optimization trace, starting from the predictable trajectory.

Exaggeration/Decoy. The *typical* strategy that users demonstrated in [58] is about selecting another goal, G_{decoy}, and conveying that through the motion. In our model, this translates to maximizing the probability of that goal:

$$\xi_{exaggerate} = \arg\max_{\xi} \int P(G_{decoy}|\xi_{S\to\xi(t)})dt \qquad (8.1)$$

Solving this optimization problem leads to the trajectory in Fig. 8.5a. This is the opposite of legibility: in a situation with two candidate goals, this strategy is equivalent to minimizing Legibility.

Switching. The switching strategy alternates between the goals. If $\sigma : [0,1] \to \mathcal{G}$ is a function mapping time to which goal to convey at that time, then the switching trajectory translates in our model to maximizing the probability of goal $\sigma(t)$ at every time point:

$$\xi_{switching} = \arg\max_{\xi} \int P(\sigma(t)|\xi_{S\to\xi(t)})dt \qquad (8.2)$$

---

[1] A.D. Dragan, R. Holladay, and S.S. Srinivasa. An analysis of deceptive robot motion. In *Robotics: Science and Systems (R:SS)*, 2014

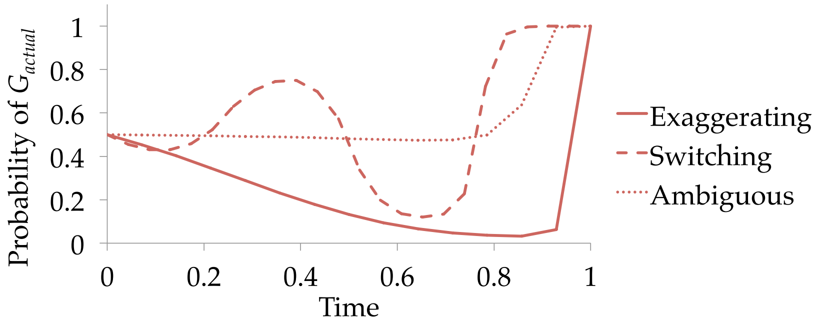


Figure 8.6: The probability of the actual goal along each model trajectory.

Unlike other strategies, this one depends on the choice of σ. Optimizing for a default choice of σ (a piece-wise function alternating between G_{other} and G_{actual}, $\sigma(t) = G_{other}$ for $t \in [0, .25) \cup [.5, .75)$ and $\sigma(t) = G_{actual}$ for $t \in [.25, .5) \cup [.75, 1]$) leads to the trajectory from Fig. 8.5b, which alternates between conveying the goal on the right and the one on the left.

Ambiguity. The ambiguous strategy keeps both goals as equally likely as possible along the way, which translates to minimizing the absolute difference between the probability of the top two goals:

$$\begin{aligned}\xi_{ambiguous} = &\arg\min_{\xi} \int |P(G_{actual}|\xi_{S\to\xi(t)}) \\ &- P(G_{other}|\xi_{S\to\xi(t)}))|dt \end{aligned} \tag{8.3}$$

Fig. 8.5c is the outcome of this optimization: it keeps both goals just as likely until the end, when it commits to one. An alternate way of reaching such a strategy is to maximize the *entropy* of the probability distribution over all goals in the scene.

8.2.2 *Comparing Strategies*

Using this model, we see that different strategies can be thought of as optimizing different objectives, which gives us insight into why exaggeration was the most popular in the user demonstrations from [58]: *it is the most effective at reducing the probability of the actual goal being inferred along the trajectory.*

Theoretical comparison. Fig. 8.6 plots the $P(G_{actual})$ along the way for each strategy: the lower this is, the more deceptive the strategy. While the ambiguous strategy keeps the probability distribution as close to $50 - 50$ as possible, and the switching strategy conveys the actual goal for parts of the trajectory, the exaggerate (or decoy)

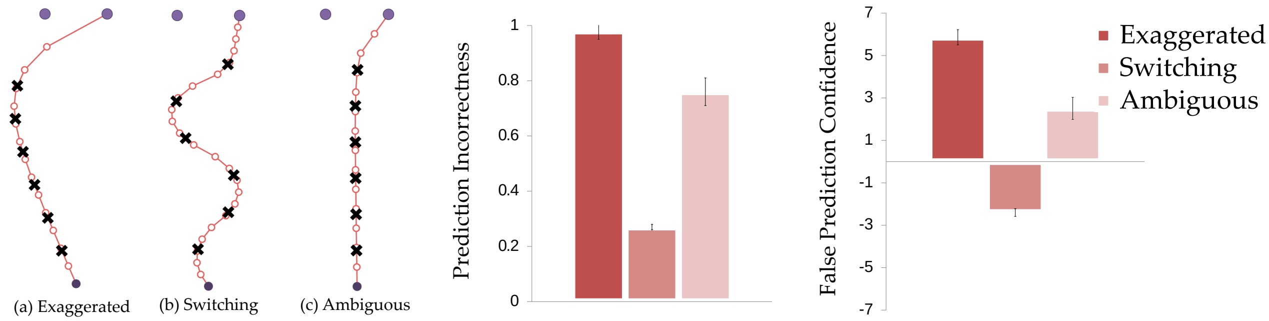

(a) Exaggerated (b) Switching (c) Ambiguous

Figure 8.7: A comparison among the three deception strategies: ambiguous, exaggerated and switching.

strategy biases the distribution toward the other goal as much as possible for the entire trajectory duration: the observer will not only be wrong, but will be *confidently* wrong.

User study comparison. We designed an online user study that compares the effectiveness of the three deception strategies from Fig. 8.5: exaggerating, switching and ambiguous. From Fig. 8.6, we predict that exaggerating is more deceptive than the other two:
Hypothesis. *The exaggerating deceptive trajectory is more deceptive then the switching and ambiguous strategies.*
Manipulated Factors. We manipulated the *deception strategy* used (with the 3 levels outlined above), and the *time point* at which the trajectory is evaluated (with 6 time points equally spaced throughout the trajectory). This yielded a total of 18 conditions.
Dependent Measures. We measured how deceptive the trajectories are by measuring which goal the users believe the robot is going toward as the trajectory is unfolding: the less correct the users are, the more deceptive the motion.

For each trajectory and time point, we generated a video of the robot (i.e., a disc on the screen) executing the trajectory up to that time point. We measured *incorrectness* and *confidence*. We asked the users to watch the video, predict which goal the robot is going towards, and rate their confidence in the prediction on a 7 point Likert scale. We treat the confidence as negative for correct predictions (meaning the trajectory failed to deceive).[2]
Participants. We used a between-subjects design again, and recruited a total of 360 users (20 per condition) on Amazon's Mechanical Turk. We eliminated users who failed to answer a control question correctly, leading to 313 users (191 male, 122 female, aged $18-65$).

Analysis. An ANOVA for *incorrectness* showed a significant main effect for *deception strategy* ($F(2,310) = 77.98$, $p < .0001$), with the

[2] This is analogous to our evaluation of legibility from the follow-up study in Section 6.3.

post-hoc revealing that all three strategies were significantly different from each other (all with $p < .0001$). An ANOVA for *false prediction confidence* yielded analogous findings.

As Fig. 8.7 shows, the exaggerating strategy was the most successful at deception, followed by the ambiguous strategy. This supports our hypothesis and the prediction of our model, since the exaggerating strategy assigns the lowest probability to the actual goal along the way (as shown in Fig. 8.6).

Fig. 8.8 shows the correctness rate over time for the three strategies. This experimental evaluation has similar results to the theoretical prediction from Fig. 8.6: the exaggerating strategy decreases correctness over time, the switching strategy oscillates, and the ambiguous strategy stays closer to .5.

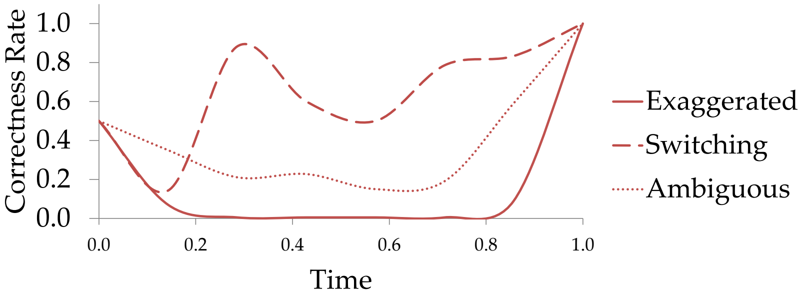


Figure 8.8: The correctness rate for the three strategies as evaluated with users.

However, we do observe differences from the predicted values. The exaggerating and ambiguous trajectories were more deceptive than expected, and the switching was less deceptive. In particular for switching, this could be an effect of the time point discretization we selected.

8.2.3 *Generalization to Arm Motion*

In this section, we put deception to the test beyond 2 degrees of freedom, by applying the model to HERB's 7DOF arm. Fig. 8.10 (top) shows the resulting deceptive trajectory, along with a comparison between its end effector trace and that of the predictable trajectory (bottom left).

Both trajectories are planned s.t. they minimize cost and avoid collisions. The difference is in the cost functional: the predictable trajectory minimizes C, while the deceptive one minimizes LEGIBILITY (implements Eq. 8.1).

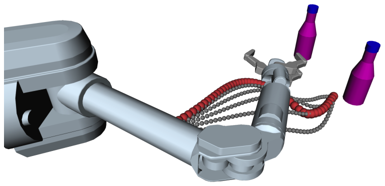


Figure 8.9: Optimization trace for deception.

Fig. 8.9 shows the optimization trace transforming the predictable into the deceptive trajectory. After a few iterations, the trajectory

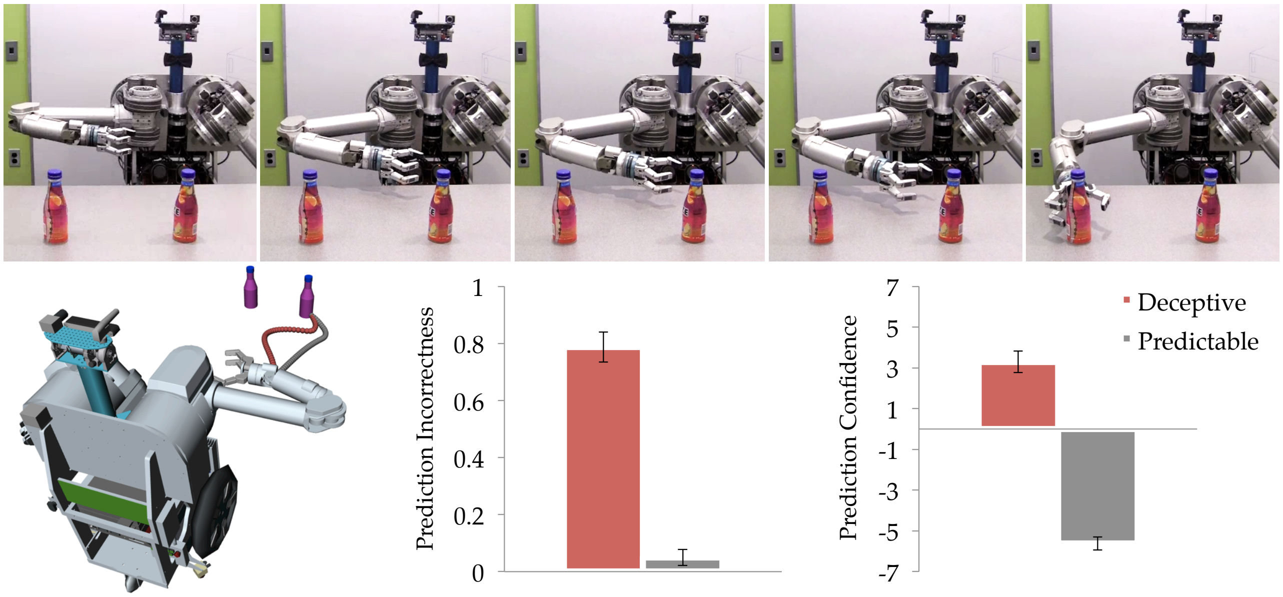


Figure 8.10: Top: The deceptive trajectory planned by the model. Bottom: a comparison between this trajectory and the predictable baseline.

shape starts bending to make progress in the objective, but remains on the constraint manifold imposed by the obstacle avoidance term.

To EVALUATE WHETHER THIS TRAJECTORY IS REALLY DECEPTIVE, we repeat our evaluation from the previous section, now with the physical robot.

Manipulated Factors and Dependent Measures. We again manipulate *trajectory* and *time-point*, this time with only two levels for the trajectory factor: the deceptive and predictable trajectories from Fig. 8.10. This results in 6 conditions. We use the same dependent measures as before.

Participants. For this study, we recruited 120 participants (20 per condition; 80 male, 40 female, aged $19-60$) on Amazon's Mechanical Turk.

Hypothesis. *The model deceptive trajectory is more deceptive than the predictable baseline.*

Analysis. In line with our hypothesis, a factorial ANOVA for *correctness* did reveal a significant main effect for *trajectory* ($F(1,117) = 150.81$, $p < .0001$). No other effects were significant. Fig. 8.10 plots the results.

The users who were deceived relied on the principle of rational action [3], commenting that the robot's initial motion towards the left "seemed like an inefficient motion if the robot were reaching for the other bottle".

When the robot's trajectory starts moving towards the other bottle,

[3] GyÃűrgy Gergely, Zoltan Nadasdy, Gergely Csibra, and Szilvia Biro. Taking the intentional stance at 12 months of age. *Cognition*, 56(2):165 – 193, 1995

the users find a way to rationalize it: “I think that jerking to my left was to adjust it arm to move right.”, or “It looks as if the robot is going for the bottle on my right and just trying to get the correct angle and hand opening”.

As for the features of the motion that people used to make their decision, the direction of the motion and the proximity to the target were by far the most prevalent, though one user quoted hand orientation as a feature as well.

Not all users were deceived, especially at the end. A few users guessed correctly from the very beginning, making (false) arguments about the robot’s kinematics, e.g., “he moved the arm forward enough so that if he swung it round he could reach the bottle”.

8.2.4 Implications of Deception for HRI

Our studies thus far test that the robot can generate deceptive motion. Our final study is about what effect this has on the perceptions and attitudes of people interacting with the robot.

Although no prior work has investigated deceptive *motion*, some studies have looked into deceptive robot *behavior* during games. A common pattern is that unless the behavior is very obviously deceptive, users tend to perceive being deceived as *unintentional*: an error on the side of the robot [197, 223, 113]. In a taxonomy of robot deception, Shim et al. [195] associate *physical* deception with unintentional, and *behavioral* deception with intentional. Deceptive motion could be thought of as either of the two, leading to our main question for this study:

> *Do people interpret deceptive motion as intentional?*

And, if so, what implications does this have on how they perceive the robot? Literature on the ethics of deception cautions about a drop in trust [94, 15], while work investigating games with cheating robots measures an increase in engagement [197, 223]. We use these as part of our dependent measures in the study.

We also measure perceived intelligence, because deception is also associated with the agent having a theory of mind about the deceived [27].

EXPERIMENTAL SETUP. We designed an experiment to test some of the implications of deception during a game.
Procedure. The participants play a game against the robot, in which they have to anticipate which bottle (of the two in front of them) the robot will grab, and steal it from the robot, like in Fig. 8.11. The faster they do this, the higher their score in the game.

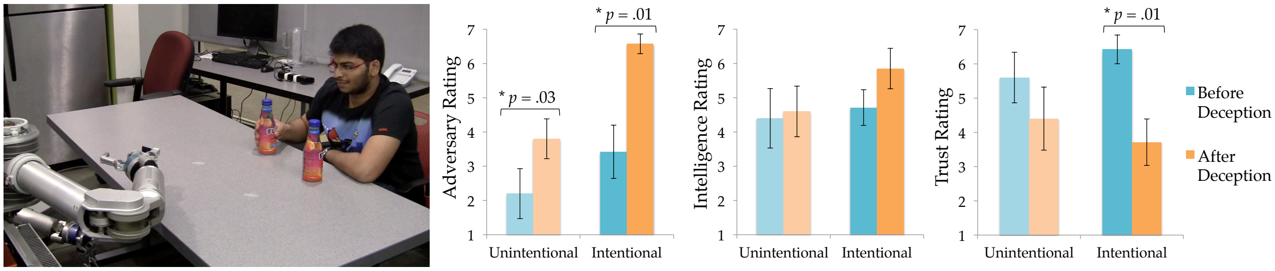


Figure 8.11: A snapshot of the deception game, along with the adversary and trust ratings: after deception, users rate the robot's skill as an adversary higher, and trust in the robot decreases. The difference is larger when they perceive the deception as intentional.

Before the actual game, in which the robot executes a deceptive trajectory, they play two practice rounds (one for each bottle) in which the robot moves predictably. These are meant to expose them to how the robot can move, and get them to form a first impression of the robot.

We chose to play two practice rounds instead of one for two reasons: (1) to avoid changing the participants' prior on what bottle is next, and (2) to show participants that the robot can move directly to either bottle, be it on the right or left. However, to still leave some suspicion about how the robot can move, we translate the bottles to a slightly different position for the deception round.

Dependent Measures. After the deception round, we first ask the participants whether the robot's motion made it seem (initially) like it was going to grab the other bottle. If they say yes, then we ask them whether they think that was intentional, and whether they think the robot is reasoning about what bottle they will think it would pick up (to test attribution of a theory of mind).

Both before and after the deception round, we ask participants to rate, on a 7 point Likert scale, how intelligent, trustworthy, engaging, and good at being an adversary the robot is.

Participants. We recruited 12 participants from the local community (9 male, 3 female, aged $20 - 44$).

Hypothesis. *The ratings for intelligence, engagement, and adversary increase after deception, but trust drops.*

Analysis. The users' interpretation was surprisingly mixed, indicating that deception in motion can be subtle enough to be interpreted as accidental.

Out of 12 users, 7 thought the robot was intentionally deceiving them, while 5 thought it was unintentional. Among those 5, 2 thought that the deceptive motion was hand-generated by a programmer, and not autonomously generated by the robot by reasoning about their inference. The other 3 attributed the way the motion looked to a necessity, rationalizing it based on how they thought the kinematics of the arm worked, e.g., "it went in that direction because it had to stretch its arm out".

Analyzing the data across all 12 users (Fig. 8.11), the rating of the robot as an adversary increased significantly (paired t-test, $t(11) = 4.60$, $p < .001$), and so did the rating on how engaging the robot is ($t(11) = 2.45$, $p = .032$), while the robot's trustworthiness dropped ($t(11) = -3.42$, $p < .01$). The intelligence rating had a positive trend (increased by .75 on the scale), but it was not significant ($p = .11$). With Bonferroni corrections for multiple comparisons, only adversary and trust remain significant, possibly because of our small sample size. Further studies with larger sample sizes would be needed to investigate the full extent of the effect of deceptive motion on the interaction.

We also analyzed the data split by whether deception was perceived as intentional — this leads to even smaller sample sizes, meaning these findings are very preliminary and should be interpreted as such. We see larger differences in all metrics in the intentional case compared to the unintentional. This is somewhat expected: if deception is attributed to an accident, it is not a reflection on the robot's qualities. The exception is the rating of the robot as an adversary: both ratings increase significantly (Fig. 8.11), perhaps because even when the deception was accidental, it was still effective at winning the game.

There was one user whose trust did not drop, despite finding deception intentional. He argued that the robot did nothing against the rules. Other users, however, commented that even though the robot played by the rules, they now know that it is capable of tricking them and thus trust it less.

IN SUMMARY, our legibility formalism can be used to generate deceptive robot motion. This motion is effective at deceiving, increases the robot's perceived capability, and lowers trust.

LIMITATIONS. Thus far, we have studied single-instance deception. Iterated deception raises a game-theoretical aspect of the problem, which we only began studying [4].

8.3 Pointing Gestures

The legibility formalism applies beyond goal-directed motion, to other channels of communication. Communication can entail explicit verbal statements [226, 90, 50] (which we discuss in Section 8.5), or nonverbal cues through gaze [165, 3] or gestures [83, 151, 190, 191].

Among these, here we focus on spacial deixis — on producing

[4] A.D. Dragan, R. Holladay, and S.S. Srinivasa. From legibility to deception. *Autonomous Robotics*, 2015

pointing gestures. Regardless of language and culture, we rely on pointing to refer to objects in daily interactions [124], be it at the grocery store, during a meeting, or at a restaurant.

Project led by Rachel Holladay.

Imagine pointing at one of the objects on a table. This pointing configuration has to accomplish two tasks: (1) it has to convey to the observer that you are pointing at the goal object, and (2) it has to convey that you are *not* pointing at any other object.

Myopically deciding on a pointing configuration that ignores this second task can lead to the situation in Fig. 8.12(top), where even though the robot's pointer is directly aligned with the further bottle, it is unclear to an observer which of two objects is the goal. It is the second task, of not conveying other goals, that ensures the clarity — or *legibility* — of the pointing gesture.

THE PROBLEM OF GENERATING pointing configurations has been studied in robotics as an inverse kinematics problem of aligning an axis with a target point [217], or a visually-guided alignment task [151]. Here, we explicitly focus on finding an axis that will make the target object most clear, analogously to work on legible motion [66, 67, 211, 83, 20, 8] or handovers [153].

Legible *pointing* has been a focus in the computer graphics community [230]. There, it is possible to go beyond the physical constraints of robots and augment a character's configuration with virtual features, such as extending the character's arm to the goal object [75], or visually highlighting the object [97]. Here, we focus on legible pointing while constrained by the physical world.

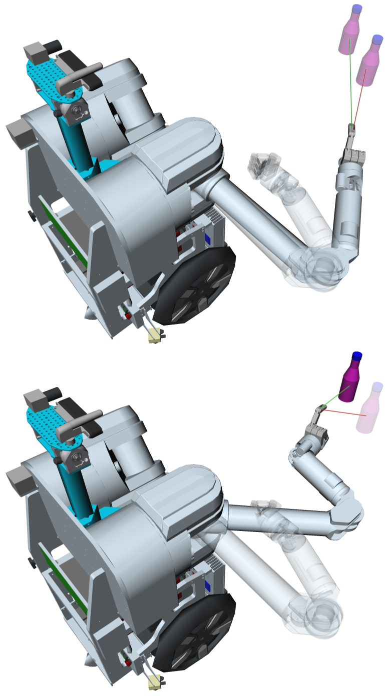


Figure 8.12: Top: An efficient pointing configuration that fails to clearly convey to an observer that the goal is the further bottle. Bottom: Its less efficient, but *more legible* counterpart, which makes the goal clear.

IF THE ROBOT IN FIG. 8.12 HAD A LASER RAY going out of its index finger and landing on the target object, then both configurations would be perfectly legible. In reality though, there is ambiguity about the pointing direction. We do not have the accuracy of laser pointers — not in forming pointing configurations, and definitely not in observing them. What we have is more akin to a torch light, shooting rays in a range of directions.

Starting with such a ray model, we introduce the cost function C for pointing, and show that applying the legibility formalism to this cost produces more legible pointing gestures.

8.3.1 The Cost C for Predictable Pointing

We begin by modeling pointing as the minimum of a cost function based on rays that shoot out from the pointer and intersect the goal objects, or get blocked by obstacles in the scene.

Formally, the robot is in a starting configuration, $S \in \mathcal{Q}$, and needs to point at the goal object G within a set of objects $\mathcal{G}$. The robot must find a pointing configuration $P \in \mathcal{Q}$. We model finding this pointer as an optimization problem.

The natural human end effector shape when pointing is to close all but the index finger [89, 39], which serves as the pointer. We assume the robot's end effector is in some equivalent shape, as in Fig. 8.12. Let $\phi(P)$ denote the transform of the pointer when the robot is in configuration P.

We expect a good pointing configuration to satisfy the following trivial properties: (1) the pointer must be directly oriented towards the goal object; (2) there should be no obstacles in between the pointer and the goal object.

We design a cost function for pointing such that the minima satisfy these properties, and deviating from them is more and more expensive. To this end, we propose a *ray model* as in Fig. 8.13, where ray vectors r shoot out from the pointer. Rays that do not contact the goal object are assigned no weight. Rays that contact the goal object are assigned a higher weight when they are more aligned with the pointer $\phi(P)$:

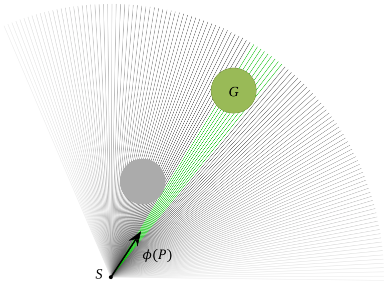


Figure 8.13: The ray model only takes into account rays that hit the object, weighing them more when they are more aligned with the pointer.

$$R_G(P) = \frac{\int \delta(P, r, G) w(r) dr}{\int w(r) dr} \tag{8.4}$$

with w increasing with the dot product between the pointer and the ray, and δ a function evaluating to 1 when the ray at angle r intersects the goal object, and 0 otherwise.

However, simply accounting for the ray intersections does not tell the whole story. As the pointer $\phi(P)$ moves closer the the goal G, more rays intersect and therefore R_G increases. This would imply that the best pointing position would be to be as close to the object as possible to the point of touching it.

In contrast, humans observing agents tend to apply the principle of rational action, expecting them to take efficient actions to achieve their goals [78]. In the case of pointing, this implies we expect robots to not deviate too much from their starting configuration. Thus, we model the cost of a pointing configuration as the trade-off between a high reward R_G and moving the minimal distance from the start:

$$C_G(P) = (1 - R_G(P)) + \frac{\lambda}{M} ||S - P||^2 \tag{8.5}$$

with

$$M = \max_{p \in \mathcal{Q}} ||S - p||^2 \tag{8.6}$$

Fig. 8.14 plots this cost for all positions in a 2D grid, assuming the direction of the pointer is aligned with the goal object (in green). There is a large increase in cost around the other object (in red),

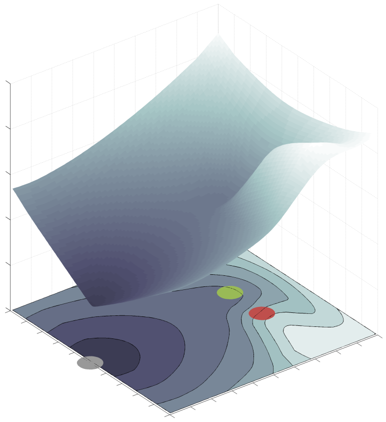


Figure 8.14: Surface plot for C_G.

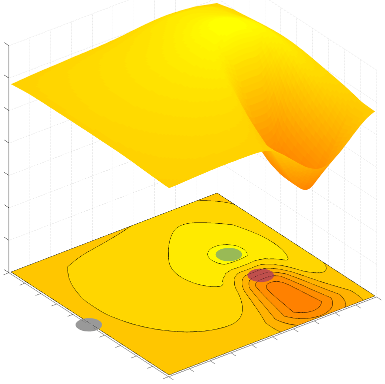


Figure 8.15: Surface plot for LEGIBILITY.

because this objects starts blocking the rays when the pointer is in those positions.

8.3.2 *Legible Pointing*

As with goal-directed motion, let

$$P(P|G) \propto e^{-C_G(P)} \tag{8.7}$$

and use that to compute

$$\textsc{Legibility}(P) = P(G|P) = \frac{e^{-C_G(P)}}{\sum_{g \in \mathcal{G}} e^{-C_g(P)}} \tag{8.8}$$

Very importantly, this normalizes over the set of candidate objects.

Fig. 8.15 shows this probability (Legibility).

Difference from C_G. A main implication of optimizing for legibility is that the distance from the starting configuration S becomes inconsequential: $P(G|P)$ does not depend on the distance to S.

Proof: $P(G|P) = \frac{e^{-C_G(P)}}{\sum_{g \in \mathcal{G}} e^{-C_g(P)}} \Rightarrow P(G|P) = \frac{e^{-(1-R_G(P))+\frac{\lambda}{M}||S-P||^2}}{\sum_{g \in \mathcal{G}} e^{-(1-R_g(P))+\frac{\lambda}{M}||S-P||^2}} \Rightarrow$
$P(G|P) = \frac{e^{-(1-R_G(P))}}{\sum_{g \in \mathcal{G}} e^{-(1-R_g(P))}}$ ■

This happens because the purpose of legibility is to find the absolute clearest pointing configuration, even if that requires more effort: legibility will spare no expense in making the goal object clear.

As a result, the optimal pointing configuration is different from the optimum with respect to C_G.

Difference from R_G. Because legibility incorporates the probability of the other potential goals in the scene, the resulting pointing configuration is also different from simply using the ray model only, R_G. We create an illustrative example in Fig. 8.16, where we constrain the position of the pointer to a fixed distance to the goal object.

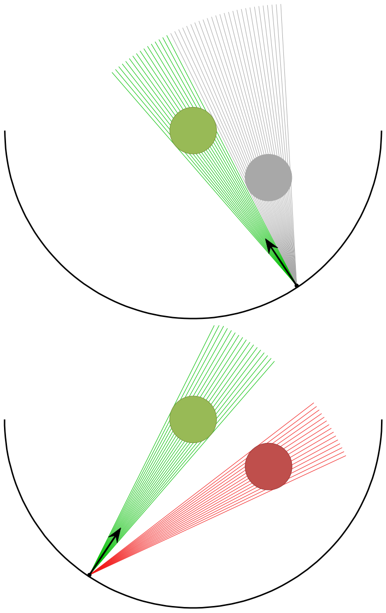


Figure 8.16: Legibility is different from the ray model because it accounts for the probability that will be assigned to the other objects. In this example, both pointers are equally good according to the ray model, because the other object does not occlude either pointer. However, the pointer in right the right image is more legible. We put this to the test in practice in our last experiment.

The figure shows two different pointers. They both have the same ray value R_G, because in both cases the other object does not block any rays that would normally hit the goal.

However, the pointer in the left image is much less legible because it does not account for the probability an observer would assign to the other object. In contrast, the pointer on the right is the result for optimizing L_G, and makes the intended goal much more clear.

8.3.3 *From Theory to Users*

Experimental Design. Our study compares how clearly a pointing configuration conveys its goal object for the cost and legibility optimizations, testing our model's prediction that maximizing legibility will be more effective than minimizing cost.

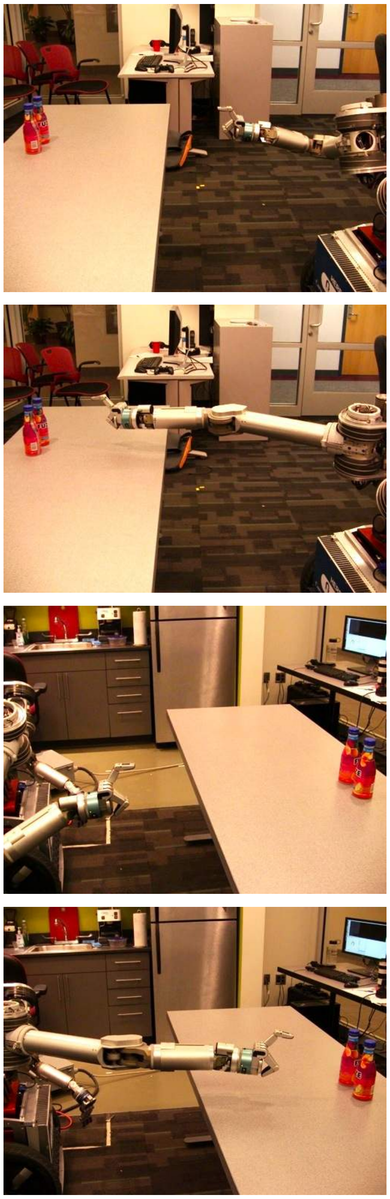


Figure 8.17: The four experimental conditions for our main study, which manipulates legibility and observer viewpoint. From top to bottom: Cost View 1, Legibility View 1, Cost View 2, and Legibility View 2.

Manipulated Factors: We manipulate *legibility* — whether the pointing is generated by minimizing the cost C_G from (8.5) or by maximizing the legibility score L_G from (8.8). For efficiency, we perform the optimization in a restricted space of pointers, where the pointer is constrained to point directly at the goal object (we explore effects of orientation exaggeration in a side study), and the optimization over position happens in the 2D plane, constrained by the robot's arm reachability.

We also manipulate the *viewpoint*. The point of view of the observer can change the perception of geometry. To control for this potential confound, we test two different opposite view points, one from the right of the robot and the other from the left.

We use a factorial design, leading to a total of four conditions, shown in Fig. 8.17.

Dependent Measures: We measure how clearly the pointing configuration expresses its goal object (as opposed to other objects in the scene).

We show the participants (an image of) the robot pointing, and ask them to 1) select which of the two objects on the table the robot is pointing at (the objects are labeled in the images) — we use this to measure *prediction correctness*, and 2) rate their confidence on a 7-point Likert scale — we use this to measure *correct prediction confidence* by computing a score equal to the confidence for correct predictions, and equal to the negative of the confidence for incorrect prediction (i.e., we penalize being confidently wrong).

We also ask participants to rate how *expected* or natural the robot's pointing configuration is, on a 7-point Likert scale, since the cost minimization was designed to better match the expectation of efficiency, while the legibility optimization was designed to be more clear about which object is conveyed.

Hypotheses:

H1. *Legibility positively affects prediction correctness and correct prediction confidence.*

H2. *Legibility negatively affects expectedness.*

Subject Allocation: We opted for a between-subjects design in order to avoid biasing the participants. This is especially important because all conditions have the same target object, and seeing one pointer affects the prior over what the robot is pointing at.

We recruited 20 participants per condition (leading to a total of 80 participants) using Amazon's Mechanical Turk. We imposed two selection criteria for the participants: a high acceptance rate on their previous work to avoid participants who are not carefully considering the task, and a US location to avoid language barriers.

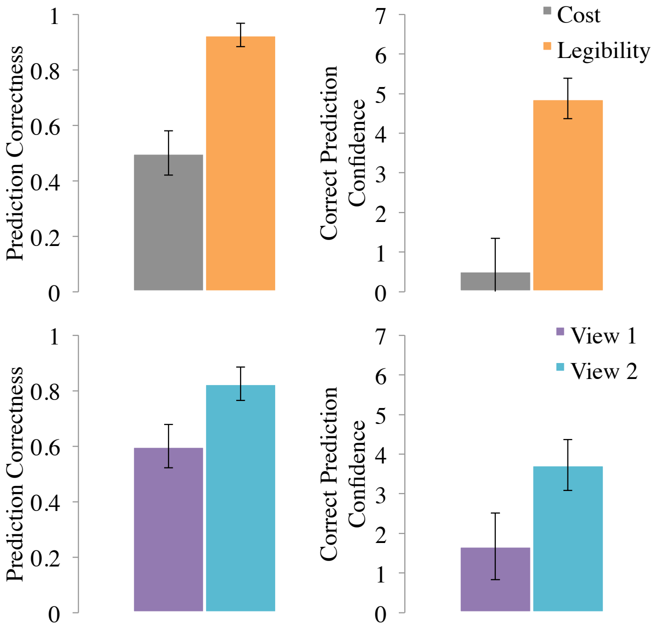


Figure 8.18: Effects of legibility (top) and viewpoint (bottom) on correctness of predictions (left), and correct prediction confidence (right).

ANALYSIS. In line with our first hypothesis, a logistic regression on *prediction correctness* with *legibility* and *viewpoint* as factors revealed a significant main effect for *legibility* (Wald $\chi^2(1,80) = 12.68$, $p < .001$): *legible pointing was indeed more legible* (or clear) than minimizing the pointing cost. The *viewpoint* factor was marginal ($\chi^2(1,80) = 2.86$, $p = .09$), with the first viewpoint leading to worse predictions.

With *correct prediction confidence*, the differences were all the more clear. A factorial ANOVA also showed a significant main effect for *legibility* ($F(1,76) = 21.86$, $p < .0001$), and also one for *viewpoint* ($F(1,76) = 4.85$, $p = 0.03$). The interaction effect was only marginal ($F(1,76) = 64.8$, $p = .057$).

Fig. 8.18 plots the two measures for each factor. We see that legibility increase both measures, but increases the confidence score more, and that it has a larger effect than the viewpoint. Our data also revealed that *legibility optimization is less susceptible to viewpoint changes than cost optimization*: for the legible pointing, the mean difference between viewpoints for confidence is only 0.25, compared to 3.85 for the cost minimization.

Looking at the rating for how expected or natural the pointing configuration is, we found that the second hypothesis was only supported for one of the easier viewpoints (view 2). An ANOVA revealed only a significant interaction effect ($F(1,76) = 12.8$, $p = .028$), with the Tukey HSD post-hoc analysis showing that for the second viewpoint (which led to large differences for the cost minimization configuration), the cost minimization configuration was significantly more expected than the legible configuration ($p = .0446$).

This was not true for the first viewpoint, where the cost minimization rating was much lower than for the first viewpoint, *despite the actual configurations being identical*. This shows the importance of viewpoints: an expected/natural configuration from one viewpoint can seem unnatural from a different viewpoint. Our conjecture is that this happens because certain viewpoints deem the cost minimization output too unclear.

In summary, we found that optimizing for legibility does make the goal object of the pointer more clear in practice.

Limitations. A key limitation is the use of images as opposed to in-person views of the robot: this was a logistical necessity for the between-subjects design, but future work should follow up with a smaller pool of in-person users, and include the full gesture from the starting configuration to the pointing one. However, even using images provided insight into the utility of legibility and the biases introduced by changes in the viewpoint.

A second limitation is that because of the ray model, the legibility gradient for pointing is not analytic, but requires numerical evaluation. A faster approximation would be needed for real-time legibility pointing optimization.

8.4 *Assistive Teleoperation*

Our legibility formalism includes a tractably-computable model of how humans infer goals from ongoing trajectories.[5] Here, we use the same algorithm to enable the robot to infer the user's goal from their ongoing trajectory.

We analyze the context of *assistive teleoperation*. In direct teleoperation, the user realizes their intent, for example grasping the bottle in Fig. 8.12, by controlling the robot via an interface. Direct teleoperation is limited by the inadequacies and noise of the interface, making tasks, especially complex manipulation tasks, often tedious and sometimes impossible to achieve. In assistive teleoperation, the robot attempts to predict the user's intent, and augments their input, thus simplifying the task. Here, the robot faces two challenges when assisting: 1) predicting what the user wants, which we do through the legibility formalism, and 2) deciding how to use this prediction to assist.

We contribute a principled analysis of assistive teleoperation. We

[5] Building on work in plan recognition [37], cognitive science [16], psychology [176], natural language understanding [90], and perception of human action [239].

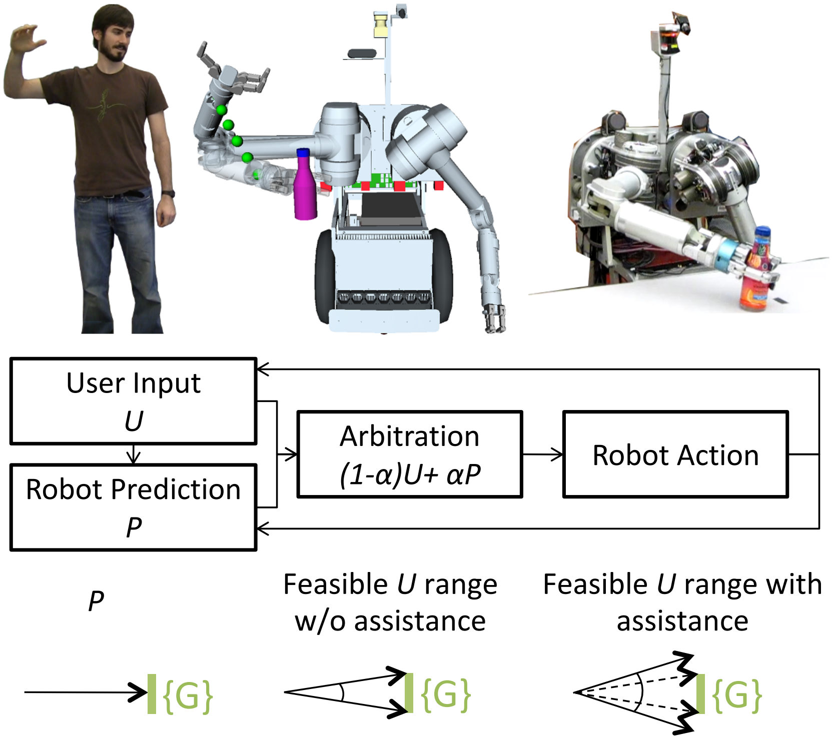


Figure 8.19: (Top) The user provides an input U. The robot predicts their intent, and assists them in achieving the task. (Middle) Policy blending arbitrates user input and robot prediction of user intent. (Bottom) Policy blending increases the range of feasible user inputs (here, $\alpha = 0.5$).

introduce *policy blending*, which formalizes assistance as an arbitration of two policies: the user's input and the robot's prediction of the user's intent. At any instant, given the input, U, and the prediction, P, the robot combines them using a state-dependent arbitration function $\alpha \in [0, 1]$ (Fig. 8.12(middle)). Policy blending with accurate prediction has a strong corrective effect on the user input (Fig. 8.12,bottom). Of course, the burden is on the robot to *predict* accurately and *arbitrate* appropriately.

Despite the diversity of methods proposed for assistance, from the robot completing the grasp when close to the goal [129], to virtual fixtures for following paths [1], to potential fields towards the goal [4], all methods can be seen as arbitrating user input and robot prediction. This common lens for assistance enables us to analyze the factors that affect its performance, and recommend design decisions for arbitration.

Prior work (detailed in Section 8.4.1) compared more manual vs. more autonomous assistance modes [155, 235, 122] with surprisingly conflicting results in terms of what users prefer. Rather than using autonomy as a factor, we introduce *aggressiveness*: arbitration should be moderated by the robot's confidence in the prediction, leading to a spectrum from very timid to very aggressive assistance, from small augmentation of user input even when confident to large augmentation even when unsure. Rather than analyzing the effect of aggressiveness (or autonomy) alone on the performance of assistance, we conduct a user study that analyzes how aggressiveness interacts with new factors, like *prediction correctness* and *task difficulty*, in order to help explain the seemingly contradictory findings from above.

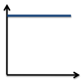
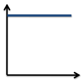
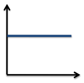
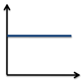
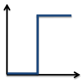
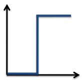
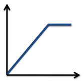
8.4.1 Prior Work as Policy Blending

In 1963, Goertz [88] proposed manipulators for handling radioactive material that are able to turn cranks based on imprecise operator inputs, introducing one of the first instances of assistive teleoperation. Since then, research on this topic has proposed a great variety of methods for assistance, ranging from the robot having full control over all or some aspect of the motion [187, 154, 49, 235, 122, 155, 51, 69], to taking control (or releasing it) at some trigger [129, 145, 194], to never fully taking control [45, 4, 235, 155, 1]. For example, Debus et al. [49] propose that the robot should be in full control of the orientation of a cylinder while the user is inserting it into a socket. In [129], the robot takes over to complete the grasp when close enough to the target. Crandal et al. [45] propose to mix the user input with a potential field in order to avoid obstacles.

Attempts to compare different modes of assistance are sometimes contradictory. For example, You and Hauser [235] found that for a complex motion planning problem in a simulated environment, users preferred a fully autonomous mode, where they only clicked on the desired goal, to more reactive modes of assistance. On the other hand, Kim et al. [122] found that users preferred a manual mode and not the autonomous one for manipulation tasks like object grasping.

POLICY BLENDING provides a unifying view of assistance, leading to an analysis which helps conciliate these differences. Table 8.1 shows how various methods proposed arbitrate user input and robot prediction (or simply robot policy, in cases where intent is assumed to be known). For example, potential field methods (e.g., [45, 4, 236]) that help the user avoid obstacles become blends of the user input with a policy obtained from the repulsive force field, under a constant arbitration function that establishes a trade-off. Virtual fixture-based methods (e.g., [155, 145, 1, 236]) that are commonly used to guide the user along a predefined path become blends of the user input with a policy that projects this input onto the path. The arbitration function dictates the intensity of the fixture at every step, corresponding to a normalized "stiffness/compliance" gain. However, the same framework also allows for the less studied case in which the robot is able

Table 8.1: Assistive teleoperation and intent prediction methods.

Method	Prediction	Arbitration
[187, 154, 49, 235, 122, 155, 144]	no	
[51, 69]	predefined paths/behaviors	
[45, 4, 235, 155]	no	
[236]	predefined paths/behaviors	
[129, 194, 148, 202]	no	
[145]	predefined paths/behaviors	
[1, 236]	predefined paths/behaviors	
[222]	fixed environment, goals (2D)	no
[239]	fully flexible (goal+policy) (2D)	no

to generate a full policy for completing the task on its own, rather than an attractive/repulsive force or a constraint (e.g., [129, 194]). In this case, the arbitration is usually a switch from autonomous to manual, although stages that trade off between the two (not fully taking control but still correcting the user's input) are also possible [12]. Arbitration as a linear blend has also been proposed for unmanned ground vehicles [12], and outside the teleoperation domain for mediating between two human input channels [86].

Analyzing assistance based on how arbitration is done, together new factors like prediction correctness and task difficulty, helps explain previously contradictory findings: our results show that aggressive assistance is preferable on hard tasks, like the ones from [235], where autonomy is significantly more efficient; opinions are split on easier tasks, like the ones from [122], where the autonomous and manual mode were comparable in terms of time to completion.

THE SAME TABLE shows how prior methods handle prediction of the user's intent. Aside from work that classifies which one of a *predefined* set of paths or behaviors the user is currently engaging [51, 69], most work assumes the robot has access to the user's intent, e.g., that it knows what object to grasp and how (except in [202], which deals with time delays in ball catching by projecting the input forward in time using a minimum-jerk model). Predicting or recognizing intent has received a lot of attention outside of the teleoperation domain, dating back to high-level plan recognition [192]. Predicting intended motion, however, is usually again limited to classifying behaviors, or is done in low-dimensional spaces [222, 239]. In the following section, which presents the building blocks of assistance, we present the general prediction problem, along with simplifying assumptions that make it tractable.

8.4.2 Arbitration

Given U and P, the robot must decide on what to do next. The arbitration function α, which makes this decision, can depend on a number of inputs, such as the distance to the goal or to the closest object, or even a binary switch operated by the user. We propose a simple principle: that arbitration must be moderated by how good the prediction is.

Timid vs. Aggressive. In trading off between not over-assisting (providing unwanted assistance) and not under-assisting (failing to provide needed assistance), the arbitration lies on a spectrum: On the one hand, the assistance could be very timid, with α taking small values even when the robot is confident in its prediction. On the other hand, it could be very aggressive: α could take large values even when the robot does not trust the predicted policy.

Inescapable Local Minima Do not Occur. In general, when arbitrating between two policies, we need to guarantee that inescapable local minima do not occur. In our case, these are states at which the arbitration results in the same state as at the previous time step, regardless of the user input.

Theorem. Let Q be the current robot configuration. Denote the prediction velocity as $p = P - Q$, and the user input velocity as $u = U - Q$. Arbitration never leads to inescapable local minima, unless $\forall u \neq 0$, $p = -ku$ for some $k \geq 0$, *and* $\alpha = \frac{1}{k+1}$ (i.e., the policy is always chosen to directly oppose the user's input, and the arbitration is computed adversarially, or $p = 0$ and $\alpha = 1$ for all user inputs).

Proof: Assume that at time t, a local minima occurs in the arbitration, i.e., $(1-\alpha)(Q+u) + \alpha(Q+p) = Q$. Further assume that this minima is inescapable, i.e., $(1-\alpha')(Q+u') + \alpha'(Q+p') = Q, \forall u'$, where p' and α' are the corresponding prediction and arbitration if u' is the next user input. $\Leftrightarrow (1-\alpha')u' + \alpha' p' = 0, \forall u'$.

Case 1: $\forall u' \neq 0$, the corresponding $\alpha' \neq 0 \Rightarrow p' = -\frac{1-\alpha}{\alpha} u', \forall u' \neq 0$ $\Rightarrow p' = -ku'$ and $\alpha = \frac{1}{k+1}$, with $k \geq 0$ (since $\alpha \in [0,1]$) $\forall u' \neq 0$. Contradiction with the problem statement.

Case 2: $\exists u' \neq 0$ s.t. the corresponding $\alpha' = 0 \Rightarrow (1-0)u' + 0p' = 0$

$\Rightarrow u' = 0$. Contradiction with $u' \neq 0$.

$\Rightarrow \exists u'$ s.t. $(1-\alpha')(Q+u') + \alpha'(Q+p') \neq Q$, ■

Therefore, with an adversarial exception, the user can always take a next action that escapes a local minimum.

Evaluating Confidence. Earlier, we had proposed that the arbitration should take into account how good the prediction is, i.e., a measure of the confidence in the prediction, c, that correlates to prediction correctness. One way to evaluate c is to assume that the closer the predicted goal gets, the more likely it becomes that it is the correct goal: $c = \max(0, 1 - \frac{d}{D})$, with d the distance to the goal and D some threshold past which the confidence is 0. Alternately, confidence can be defined as the probability assigned to the prediction. If a cost function is assumed, the match between the user's input and this cost should also factor in. If a classifier is used for prediction, then such a probability is obtained through calibration [6].

8.4.3 *A Study on Assistance*

Mathematically, arbitration can be any non-adversarial function of the robot's confidence in its prediction, from very timid to very aggressive. But assistive teleoperation is fundamentally a human-robot interaction task, and this interaction imposes additional requirements on arbitration: the robot must arbitrate in an efficient and user-preferred way. Therefore, we embarked upon a user study that analyzes the effect of the aggressiveness of arbitration on the performance of assistance — an analysis that we believe must incorporate other factors, like prediction correctness (users might not appreciate assistance if the robot is wrong) and task difficulty (users might appreciate assistance if the task is very hard for them).

EXPERIMENTAL DESIGN. We tasked 8 users with teleoperating the robot to grasp an object from a table, as in Fig. 8.19. There were always two graspable objects, and we gave the user, for every trial, the farther of the two as goal. We implemented a whole-body interface that tracks their skeleton (OpenNI, www.openni.org), yielding an arm configuration which serves as the user input U. The robot makes a prediction of the goal and the policy to it (that minimizes length in configuration-space), leading to P, and combines the two via the arbitration function α.

Hypotheses. We test the following two hypotheses:

1. **H1.** *Prediction correctness, task difficulty, and aggressiveness of assistance each has a significant effect on task performance.*

Main Effects

2. **H2.** *Aggressive assistance performs better on hard tasks if the robot is*

Interaction Effects

---

[6] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, 1999

right, while the timid assistance performs better on easy task if the robot is wrong.

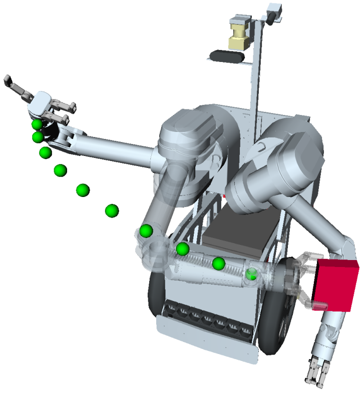


Figure 8.20: Hard and Right Task

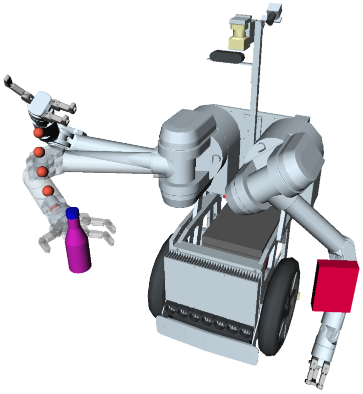


Figure 8.21: Hard and Wrong Task

Manipulated Variables. We manipulated prediction correctness by using a simple, easy to manipulate goal prediction method: the amnesic prediction based on workspace distance, which always selects the closest object. We setup wrong conditions at the limit of the robot being wrong yet rectifiable. We place the intended object further, guaranteeing wrong prediction until the user makes his preference clear by providing an input U closer to the correct goal. We setup right conditions by explicitly informing the robot of the user's intended goal.

We manipulated task difficulty by changing the location of the two objects and placing the target object in an easily reachable location (e.g., grasping the bottle in Fig. 8.21 makes an easy task) vs. a location at the limit of the interface's reachability (e.g., grasping the box in Fig. 8.21 is a hard task). This leads to four types of tasks: Easy&Right, Easy&Wrong, Hard&Right and Hard&Wrong.

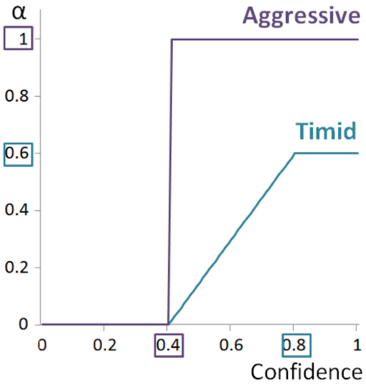


Figure 8.22: The arbitration function for the timid and the aggressive assistance modes. The aggressive mode reaches a higher maximum value earlier.

Finally, we manipulated the aggressiveness of the assistance by changing the arbitration function, and used the distance-based measure of confidence from Section 8.4.2. As the user makes progress towards the predicted object, the confidence increases. We had two assistance modes, shown in Fig. 8.22: the timid mode increases the assistance with the confidence, but plateaus at a maximum value, never fully taking charge. On the other hand, the aggressive mode eagerly takes charge as soon as the confidence exceeds a threshold.

Subject Allocation. We chose a within-subjects design, enabling us to ask users to compare the timid and aggressive mode on each task. Each of our 8 participants (all students, 4 males and 4 females) executed both modes on each of the four types of tasks. To avoid ordering effects, we used a balanced Latin square for the task order, and balanced the order of the modes within each task.

Dependent Measures. We measure the performance of assistance in two ways: the amount of time each user took to complete the task under each condition, and each user's preference for the timid vs. the aggressive mode on each task type (on a 7 point Likert scale where the two ends are the two choices). We expect the two measures to be correlated: if an assistance mode is faster on a task, then the users will also prefer it for that task. We also asked the users additional questions for each condition, about how helpful the robot was, how much its motion matched the intended motion, and how highly they would rate the robot as a teammate.

Covariates. We identified the following confounds: the users' initial teleoperation skill, their rating of the robot without assistance, and the learning effect. To control for these, users went though a *training* *phase*, teleoperating the robot without assistance. This partially eliminated the learning effect and gave us a baseline for their timing and ratings. We used these as covariates, together with number of tasks completed at any point — a measure of prior practice.

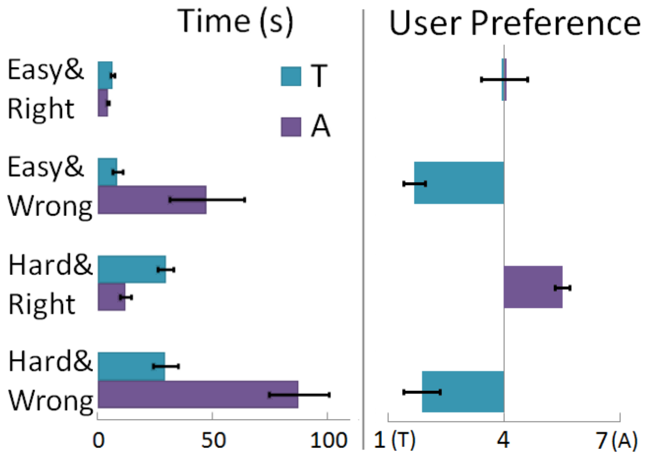


Figure 8.23: The results of the assistive teleoperation user study.

Analysis. We analyze both the objective and subjective measures. **Teleoperation Timing.** The average time per task was approximately 28s. We performed a factorial repeated-measures ANOVA with Bonferroni corrections for multiple comparisons and a significance threshold of $p = 0.05$, which resulted in a good fit of the data ($R^2 = 0.66$). In line with our first hypothesis, we found main effects for all three factors: hard tasks took 22.9s longer than easy ones ($F(1,53) = 18.45$, $p < .001$), tasks where the policy was wrong took 30.1s longer than when right ($F(1,53) = 31.88$, $p < .001$), and the aggressive mode took overall 19.4s longer than the timid ($F(1,53) = 13.2$, $p = .001$). We found a significant interaction effect between aggressiveness and correctness, showing that when wrong, being timid is significantly better than being aggressive. This is confirmed in Fig. 8.23, which compares the means and standard errors on each task: the timid mode is better on both Easy&Wrong and Hard&Wrong. The timid mode performed about the same on Easy&Right, and, as expected, worse on Hard&Right (the time taken for aggressive is smaller than for timid for every user). Surprisingly, the interaction effect among all factors was only marginally significant ($F(1,53) = 2.63$, $p = .11$). We believe that increasing our user pool would strengthen this effect.

To conclude based on this regression that the timid mode is overall better would be misleading, because it would assume that the robot is wrong in 50% of the tasks (in general, either by predicting he wrong goal, or by computing a motion that, for example, collides with an unseen obstacle). Our data indicates that the aggressive mode is overall more efficient if the robot is wrong in less than 16% of the cases. However, efficiency is only part of the story: as the next section points out, *some users are more negatively affected than others by a wrong robot policy*.

User Preferences. Fig. 8.23 also shows the users' preferences on each task, which indeed correlated to the timing results (Pearson's $r(30) = .66$, $p < .001$). The outliers were users with stronger preferences than the time difference would indicate. For example, some users strongly preferred the timid mode on Hard&Wrong tasks, despite the time difference not being as high as with other users. The opposite happened on Hard&Right tasks, on which some users strongly preferred the aggressive mode despite a small time difference, commenting that they appreciated the precision of the autonomy. On Easy&Right tasks, the opinions were split and some users preferred the timid mode despite a slightly longer time, motivating that they felt more in control of the robot. Despite the other measures (helpfulness, ranking as a teammate, etc.) strongly correlating to the preference rating ($r(30) > .85$, $p < .001$), they provided similar interesting nuances. For example, the users that preferred the aggressive mode on Easy&Right tasks because they liked having control of the robot were willing to admit that the aggressive mode was more helpful. On the other hand, we also encountered users that preferred the aggressive mode, and even users that followed the robot's motion while aggressive, not realizing that they were not in control and finding the motion of the robot to match their own very well (i.e., the predicted policy P matched what they intended, resulting in seamless teleoperation).

IN SUMMARY, although difference in timing is a good indicator of the preference, *it does not capture a user's experience in its entirety*. First, some users exaggerate the difference in preferences. Second, some users prefer the timid mode despite it being slightly less efficient. Third, *assistance shouldn't just be quick — it should also be intent-transparent*. Our users commented that "Assistance is good if you can tell that [the robot] is doing the right thing".

8.5 Relation to Language

Tellex et al. [7] used the same underlying legibility formalism to make natural language requests from a robot legible — easily understood by a human.

In language, motions become utterances, and goals become groundings. To speak legibly, the robot needs to maximize the probability

[7] Stefanie Tellex, Ross Knepper, Adrian Li, Daniela Rus, and Nicholas Roy. Asking for help using inverse semantics

that the listener will infer a desired grounding from the robot's utterance:

$$\max_{\Lambda} P(\Gamma|\Lambda,\phi)$$

with ϕ a correspondence vector mapping words to their groundings.

Motion	Language
trajectory ξ	utterance Λ
goal G	grounding Γ

The predictability inference, $P(\Lambda|\Gamma,\phi)$, captures efficiency in speech:

$$P(\Lambda|\Gamma,\phi) \propto P(\phi|\Lambda,\Gamma) = \prod_i P(\phi_i|\lambda_i, \gamma_{i_1}, .., \gamma_{i_k})$$

This ignores constant terms, and factorizes the probability using a graphical model structure called a *grounding graph* [212]. The product prevents adding unnecessary words.

Maximizing for legibility uses this probability, but normalizes it over the space of possible groundings as opposed to possible sentences, taking into consideration what the listener might infer and making sure to disambiguate the desired grounding from the rest:

$$\max_{\Lambda} \frac{P(\phi|\Lambda,\Gamma)}{\sum_{\Gamma'} P(\phi|\Lambda,\Gamma')}$$

The result is that the robot will say "Hand me the table leg" when there is only one such option, but it will add clarifying adjectives that best differentiate that leg from any other in the scene when necessary, e.g., "Hand me the while table leg", or "Hand me the table leg that is on the couch".

9
Final Words

The goal of this thesis was to integrate the notion of a human observer, and in particular the inferences that he or she makes, into motion planning. We focused on two complementary inferences that are fundamental to goal-directed motion: "action-to-goal" and "goal-to-action", and formalized predictability and legibility of motion based on them (Chapter 3).

We modeled predictability using the principle of rational action: we assumed that the observer expects the robot to take the most efficient motion to achieve its goal, and captured efficiency via a cost functional over the space of trajectories. We mainly worked with a simple assumption of what efficiency means to the observer, but Chapter 5 also introduced ways of learning predictable motion from demonstration, or familiarizing the observer with the robot's own notion of efficiency. However, customizing predictable motion to the observer and taking advantage of the co-adaptation that will occur is still an open area of research.

The cost function for predictability induced, though the principle of maximum entropy, a probability density function over trajectories. Bayesian inference starting with this density function (along with a Laplace approximation for tractability) gave the robot a model of how the observer infers its goal from its ongoing trajectory. This model echoes techniques and findings in plan recognition, cognitive science, natural language understanding, and perception of humans. The exact same model enabled the robot to make the same inference about a human during a collaborative task, so that it can then assist the human in achieving the task more efficiently (Section 8.4.

Armed with such a model, the robot could then use trajectory optimization (Chapter 4) to find motions that are *legible* — that make the observer infer the correct goal quickly and confidently (Chapter 6). Techniques from animation that we typically need to hand-code, such as exaggeration, naturally emerge out of this optimization.

Our user studies were two-fold. A first set of studies tested the model's ability to generate legible motion, supporting our hypothesis that legibility in practice increases within some trust region as the theoretical legibility score improves. But we also tested what impact legible motion has on collaborations.

When users collaborated with the robot on a physical task, predictable motion was significantly better than purely functional motion both objectively, as well as subjectively (as measured through multi-item Likert scales for the fluency of the collaboration). Legible motion was better than predictable motion, albeit the difference were more subtle.

Participants rationalized the legible motion as more efficient, suggesting that with legibility, robots can spend the same the same optimization effort (modulo convergence to local optima) as with predictable motion but improve efficiency and user perception. With functional motion, participants mainly complained about coordination with the robot being difficult, and not about feeling unsafe.

Overall, our studies suggest that legibility can be useful when robots collaborate, and that functional motion could be sufficient for tasks where no coordination is required and the human and robot don't share the workspace. One-time tasks are especially well served by legible motion, because the need for intent inference is highest. In particular, one-time tasks with ambiguous scenes require legibility for coordination. In contrast, repetitive tasks where the human would apriori know the goal are well served by predictable motion when the human and the robot interact in the same workspace, and by functional motion when the workspaces are different.

Finally, we have also seen that even though we designed the formalism for goal-directed motion, it is applicable across different tasks and channels of communication, including gestures and even language (Chapter 8).

HOWEVER, THESE FINAL WORDS ARE ANYTHING BUT FINAL. There are so many aspects of legible motion that remain open challenges, including handling continuous goal regions as opposed to a set of discrete goal configurations, handling multiple observers, analyzing the interaction between different channels of communication (and deciding on the best one to use for a give task), and studying how the communication should evolve and adapt over the course of repeated interactions.

Furthermore, goal-to-action and action-to-goal inferences, albeit important, are by no means the only inferences that humans make when observing motion. We infer properties of the agent, of the task, of the current behavior. This thesis is merely one step towards autonomously generating motion that is mindful of and that can influence these inferences.

List of Figures

1.1 Thesis overview. We introduce a formalism for robot motion planning with a human observer. We formalize predictability and legibility as properties of motion that enable the observer's goal-to-action and action-to-goal inferences: we first introduce mathematical measures for these properties that are tractable to evaluate, and then use a combination of trajectory optimization and learning techniques to autonomously generate predictable and legible motion. We also show generalizations to deception, pointing gestures, and assistive teleoperation. Finally, we evaluate the impact of this motion in physical interactions. 9

3.1 Functional motion. 21
3.2 Consistent motion. 22
3.3 Predictable motion. 22
3.4 Legible motion. 23
3.5 We model the observer's expectation as the optimization of a cost function C (above). The observer identifies based on C the most probable goal given the robot's motion so far (below). 25
3.6 $\xi_{S\to Q}$ in black, examples of $\xi_{Q\to G}$ in green, and further examples of $\xi_{S\to G}$ in orange. Trajectories more costly w.r.t. C are less probable. 26
3.7 The end effector trace for the HERB predictable (gray) and legible (orange) trajectories. 28
3.8 We use three characters: a point robot (dot on the screen), a bi-manual manipulator, and a human actor. 28
3.9 The trajectories for each character. 29
3.10 Ratings (on Likert 1-7) of how much the trajectory matched the one the subject expected. 31
3.11 The drawn trajectories for the expected motion, for ξ_P (predictable), and for ξ_L (legible). 31
3.12 Cumulative number of users that responded and were correct (above) and the approximate probability of being correct (below). 32

3.13 Legibility is not obstacle avoidance. Here, in the presence of an obstacle that is not a potential goal, the legible trajectory still moves towards the wall, unlike the obstacle-avoiding one (gray trace). 33

4.1 The obstacle cost tracks a set of body points through time. Each body point at each time point has a workspace gradient, which Eq. 4.3 compounds in a trajectory gradient. 37
4.2 A couples time along the trajectory, turning the trajectory into an elastic band: when a Euclidean gradient would pull one single point away from the rest of the trajectory, the natural gradient pulls the entire trajectory with it (details in Section 4.1.3. 38
4.3 A Euclidean inner product makes trajectory b closer to a than c is. In contrast, our example inner product makes c closer. 38
4.4 The top plots the columns of the identity matrix (each time point is independent), whereas the bottom plots the columns of A^{-1}, for $A = K^T K$ (a change at one time point leads to a propagation to the rest of the trajectory). 39
4.5 Grasping in clutter scenes, with different starting configurations, target object locations, and clutter distribution (from left to right: no clutter, low, medium and high clutter). 40
4.6 From left to right: a paired time comparison between RRT and CHOMP when both algorithms succeed, success rates for both algorithms within the 20 s time interval and the planning time histograms for both algorithms. In the time comparison chart on the left, each data point is one run of the RRT algorithm vs. the discrete run of CHOMP on a problem. Due to the large number of data points, the standard error on the mean is very small. 41
4.7 The start and the goal for a complex problem of reaching into the back of a narrow microwave. The robot is close to the corner of the room, which makes the problem particularly challenging because it gives the arm very little space to move through. The goal configuration is also very different from the start, requiring an "elbow flip". Two starts were used, one with a flipped turret (e.g., J1 and J3 offset by π, and J2 negated), leading to very different straight-line paths. 43
4.8 Top: The trajectory found when using specified single goal. The optimizer cannot avoid collision with the red box. Bottom: A feasible trajectory found by an optimizer that can take advantage of a goal set. 44
4.9 The constrained update rule takes the unconstrained step and projects it w.r.t. A onto the hyperplane through ξ_i parallel to the approximated constraint surface (given by the linearization $B(\xi - \xi_t) + b = 0$). Finally, it corrects the offset between the two hyperplanes, bringing ξ_{i+1} close to $\mathcal{H}[\xi] = 0$. 46

4.10 One iteration of the goal set version of the optimizer: take an unconstrained step, project the final configuration onto the constraint surface, and propagate that change to the rest of the trajectory. 47
4.11 Changing the goal decreases cost. The goal set algorithm modifies the trajectory's goal in order to reduce its final cost. The figure plots the initial vs. the final goals obtained by the single goal and the goal set algorithm on a grasping in clutter problem. The area of each bubble is proportional to the cost of the final trajectory. 49
4.12 A cost comparison of the single goal with the goal set variant of CHOMP on problems from four different environment types: grasping in clutter from a difficult, and from an easy starting configuration, handing off an object, and placing it in the recycle bin. 50
4.14 The end effector trajectory before and after optimization with Goal Set CHOMP. The initial (straight line in configuration space) trajectory ends at a feasible goal configuration, but collides with the clutter along the way. The final trajectory avoids the clutter by reaching from a different direction. 51
4.13 The trajectory obtained by CHOMP for extracting the bottle from the microwave while keeping it upright (a trajectory-wide constraint). 51
4.15 A toy example that exemplifies the idea of attributes: there are two basins of attraction, and a simple attribute (the decision of going right vs. left) discriminates between them. 54
4.16 High-dimensional problems are described by many basins of attraction, but there are often attributes of the trajectory that can discriminate between low cost basins and high cost basins. In this case, such an attribute is around vs. above the fridge door. 55
4.17 Once the right choice is made (above the fridge door), we can easily create a trajectory that satisfies it. This trajectory can have high cost, but it will be in the basin of attraction of a low-cost solution, and running a local optimizer (e.g., CHOMP) from it produces a successful trajectory. 56
4.18 Top: the robot in one of the goal configurations for grasping the bottle. Bottom: for the same scene, the black contour is a polar coordinate plot of the final cost of the trajectory that the optimizer converges to as a function of the goal it starts at; goals that make it hard to reach the object are associated with higher cost; the bar graph shows the difference in cost between the best goal (shown in green and marked with *) and the worst goal (shown in red). 56
4.19 Feature 1: the length of the straight line trajectory. 57
4.20 Features 2 and 3: the obstacle cost of the goal and of the straight line trajectory. 57
4.21 Feature 5: the free space radius around the elbow. 57
4.22 Feature 6: collision with the target object. 58

4.23 From left to right: the actual vs. predicted cost without thresholding, the actual vs. predicted cost with thresholding, and the dependence of the fit error of a validation set of medium and low cost examples on the threshold (on the left of the minimum, the regressors pays too much attention to high costs, on the right it uses too little data. 59
4.24 Two training situations along with their corresponding best goal, and a test situation in which the correct goal is predicted. If the learner were constrained to the set of previously executed trajectories, it would not have been able to generalize to this new scene. 61
4.25 The loss over the minimum cost on the same test set when training on scenes that are more and more different, until everything changes drastically in the scene and performance drops significantly. However, the loss decreases back to around 8% when training on a wide range of significantly different scenes, showing that the algorithm can do far transfers if given enough variety in the training data. 62
4.26 Top: Percentage loss over the best cost for all the methods. Solid bars are the data-efficient versions, and transparent bars are the vanilla algorithms, which perform worse. Bottom: The predicted minimum cost vs. the true minimum cost as function of the number of choices considered. 62

5.1 Using a norm M for adaptation propagates the change in the start and goal, from $\{s,g\}$ to $\{\hat{s}, \hat{g}\}$, to the rest of the trajectory, changing ξ_D into $\hat{\xi}$. The difference between the two as a function of time is plotted in blue. 67
5.2 In contrast, DMPs represent the demonstration as a spring damper system tracking a moving target trajectory $\mathcal{T}_D$, compute differences f_D (purple) between $\mathcal{T}_D$ and the straight line trajectory, and apply the same differences to the new straight line trajectory between the new endpoints. This results in a new target trajectory $\hat{\mathcal{T}}$ for the dynamical system to track. When $M = A$, the velocity norm from Eq. 4.7, the two adaptations are equivalent. In general, different norms M would lead to different adaptions. 67
5.3 We adapt ξ_D by finding the closest trajectory to it that satisfies the new end point constraints. The x axis is the start-goal tuple, and the y axis is the rest of the trajectory. M warps the space, transforming (hyper)spheres into (hyper)ellipsoids. The space of all adaptations of ξ_D is a linear subspace of Ξ. 69
5.4 Minimum jerk. 74
5.5 Reweighing time. 74
5.6 Coupling timepoints. 74
5.7 The different changes to the norm structure result in different adaptation effects. 75

5.8 Left: an ideal adapted trajectory (gray), a noisy adapted trajectory (red) that we use for training, and the reproduction using the learned norm (green), with a 6-fold average reduction in noise. Center: the error on a test set as a function of the number of training examples. Right: the error on a test set as a function of the amount of noise, compared to the magnitude of the noise (red). Error bars show standard error on the mean — when not visible, the error bars are smaller than the marker size. 77
5.9 The average waypoint error on a holdout set of pointing gesture demonstrations on the HERB robot, for the adaptations obtained using the learned norm, compared to error when using the default A. 79
5.10 A comparison between adapting trajectories with the default A metric (c) and adapting using a learned metric (d) on a holdout set of demonstrated pointing gestures (shown in black). The trajectory ξ_D used for adaptation is in gray. Note that the adaption happens in the full configuration space of the robot, but here we plot the end effector traces for visualization. The learned norm more closely reproduces two of the trajectories, and has higher error in the third. Overall, the error decreases significantly (see Fig. 5.9). 80
5.11 (Top) One of our users getting more comfortable with working/standing next to the robot after familiarization, as he can better predict how the robot will move. (Bottom) Users identify the robot's actual trajectory (we plot here its end effector trace only, in green, but show users the robot actually moving along it) as the one they expect more often after familiarization. 81
5.12 For the same situation, the trajectories for the more natural motion in Section 5.3.2 (top, green), and for the less natural motion in Section 5.3.3 (bottom, orange). 83
5.13 The overall experimental procedure, consisting of a familiarization phase (b), and a pre- and post-test for predictability (a and c). The tests involve three types of examples (`Levels 1-3`), each with two instances to aid robustness. For each example, we show users three trajectories and ask them to identify which one they expect the robot to perform, as well as rate each on a predictability scale. The grid in (d) depicts target object placements on the table (shown in Fig. 8.12 and Fig. 5.12) to produce the familiarization examples. The ones we re-use for testing (`Level 1`) are highlighted in blue, and the ones we set aside for testing-only (`Level 3`) are highlighted in brown. The crosses represent additional example locations we use in the follow up study with more examples. 84
5.14 Example of the three distance levels. 86

5.15 Overall, familiarization significantly improves the accuracy in recognizing the robot's motion (left). Different test situations, however, show different improvements (right). Error bars show standard error. 88
5.16 Results for familiarization to a less natural motion, as compared to the more natural CHOMP motion from Fig. 5.15. The error bars represent standard error on the mean. Familiarization does improve predictability, but not to the level of the more natural C motions. 90
5.17 The limitation of familiarization on less natural motion is not due to the number of examples, since more examples fail to improve performance. 92
5.18 Markers measuring distance to the robot are spaced 5 inches apart. Familiarization brought users 7.35 inches closer to the robot. 93

6.1 The legibility optimization process for a task with two candidate goals. By moving the trajectory to the right, the robot is more clear about its intent to reach the object on the right. 98
6.2 Legible trajectories on a robot manipulator assuming C, computed by optimizing Legibility in the full dimensional space. The figure shows trajectories after 0 (gray), 10, 20, and 40 iterations. 99
6.3 A full-arm depiction of the optimized trajectories at 0 and 20 iterations. 99
6.4 More ambiguity (right) leads to the need for greater departure from predictability. 100
6.5 Smaller scales (left) lead to the need for greater departure from predictability. 100
6.6 Effects of the weighting function $f(t)$. 100
6.7 Legible trajectories for multiple goals. 100
6.8 Legibility given a C that accounts for obstacle avoidance. The gray trajectory is the predictable trajectory (minimizing C), and the orange trajectories are obtained via legibility optimization for 10, 10^2, 10^3, 10^4, and 10^5 iterations. 101
6.9 Legibility is dependent on the initialization. 101
6.10 The expected (or predictable) trajectory in gray, and the legible trajectories for different trust region sizes in orange. On the right, the cost C over the iterations in the unconstrained case (red) and constrained case (green). 102
6.11 We measure legibility by measuring at what time point along the trajectory users feel confident enough to provide a goal prediction, as well as whether the prediction is correct. 104

6.12 Left: The legibility score for all 7 conditions in our main experiment: as the trust region grows, the trajectory becomes more legible. However, beyond a certain trust region size ($\beta = 40$), we see no added benefit of legibility. Right: In a follow-up study, we showed users the entire first half of the trajectories, and asked them to predict the goal, rate their confidence, as well as their belief that the robot is heading towards neither goal. The results reinforce the need for a trust region. 105
6.13 The distribution of scores for three of the conditions. With a very large trust region, even though the legibility score does not significantly decrease, the users either infer the goal very quickly, or they wait until the end of the trajectory, suggesting a legibility issue with the middle portion of the trajectory. 106

7.1 Snapshots from the three types of motion at the same time point along the trajectory. The robot is reaching for the dark blue cup. The functional motion is erratic and somewhat deceptive, and the participant leans back and waits before committing to a color. The predictable motion is efficient, but ambiguous, and the participant is still not willing to commit. The legible motion makes the intent more clear, and the participant is confident enough to start the task. 110
7.2 The end effector traces of the three types of motion for one part of the task. 110
7.3 For each tea order, the robot starts reaching for one of the cups. The participant infers the robot's goal and starts gathering the corresponding ingredients. Both place their items on the tray, and move on to the next order. For order #3, the cups are further away from the robot, and closer to each other, making the situation ambiguous. 112
7.4 Findings for objective measures. 117
7.5 Some of the participants kept a larger distance to the robot during the functional condition. However, most participants were surprisingly comfortable with the robot during this condition. 117
7.6 Findings for subjective measures. Closeness was on a 5-point scale. 118

8.1 The red trajectory works in viewpoint 1, but is not as legible in viewpoint 2. The robot finds a different way to exaggerate when the observer has a different viewpoint (green trajectory). From viewpoint 2, it looks like the robot is exaggerating more, but that is not the case (see green trajectory in viewpoint 1). The two trajectories have the same cost C, but exaggerate in different directions (see viewpoint 3). 123
8.2 The robot does not exaggerate in the occluded region, so that it can exaggerate more outside of it. 124
8.3 The robot uses a smaller than needed hand aperture to convey that it will grasp the smaller object. 124

8.4 The robot uses a larger than needed hand aperture to convey that it will grasp the larger object. 124

8.5 Strategies replicated by the model: the typical exaggeration towards another goal, as well as the switching and ambiguous trajectories. The trajectories in gray show the optimization trace, starting from the predictable trajectory. 125

8.6 The probability of the actual goal along each model trajectory. 126

8.7 A comparison among the three deception strategies: ambiguous, exaggerated and switching. 127

8.8 The correctness rate for the three strategies as evaluated with users. 128

8.9 Optimization trace for deception. 128

8.10 Top: The deceptive trajectory planned by the model. Bottom: a comparison between this trajectory and the predictable baseline. 129

8.11 A snapshot of the deception game, along with the adversary and trust ratings: after deception, users rate the robot's skill as an adversary higher, and trust in the robot decreases. The difference is larger when they perceive the deception as intentional. 131

8.12 Top: An efficient pointing configuration that fails to clearly convey to an observer that the goal is the further bottle. Bottom: Its less efficient, but *more legible* counterpart, which makes the goal clear. 133

8.13 The ray model only takes into account rays that hit the object, weighing them more when they are more aligned with the pointer. 134

8.14 Surface plot for C_G. 134

8.15 Surface plot for LEGIBILITY. 134

8.16 Legibility is different from the ray model because it accounts for the probability that will be assigned to the other objects. In this example, both pointers are equally good according to the ray model, because the other object does not occlude either pointer. However, the pointer in right the right image is more legible. We put this to the test in practice in our last experiment. 135

8.17 The four experimental conditions for our main study, which manipulates legibility and observer viewpoint. From top to bottom: Cost View 1, Legibility View 1, Cost View 2, and Legibility View 2. 136

8.18 Effects of legibility (top) and viewpoint (bottom) on correctness of predictions (left), and correct prediction confidence (right). 137

8.19 (Top) The user provides an input U. The robot predicts their intent, and assists them in achieving the task. (Middle) Policy blending arbitrates user input and robot prediction of user intent. (Bottom) Policy blending increases the range of feasible user inputs (here, $\alpha = 0.5$). 139

8.20 Hard and Right Task 144

8.21 Hard and Wrong Task 144

8.22 The arbitration function for the timid and the aggressive assistance modes. The aggressive mode reaches a higher maximum value earlier. 144
8.23 The results of the assistive teleoperation user study. 145

List of Tables

3.1 Legibility and predictability as enabling inferences in opposing direction. 24

4.1 Comparison of CHOMP and RRT for different time budgets. 43

5.1 The predictability scale. 87
5.2 The utility of familiarization ratings. 89
5.3 The motion ratings. 89

7.1 Subjective measures. 115

8.1 Assistive teleoperation and intent prediction methods. 141

10
Bibliography

[1] D. Aarno, S. Ekvall, and D. Kragic. Adaptive virtual fixtures for machine-assisted teleoperation tasks. In *IEEE ICRA*, 2005.

[2] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[3] Henny Admoni, Caroline Bank, Joshua Tan, Mariya Toneva, and Brian Scassellati. Robot gaze does not reflexively cue human attention. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society, Boston, MA, USA*, pages 1983–1988, 2011.

[4] P. Aigner and B. McCarragher. Human integration into robot control utilising potential fields. In *ICRA*, 1997.

[5] K. Akachi, K. Kaneko, N. Kanehira, S. Ota, G. Miyamori, M. Hirata, S. Kajita, and F. Kanehiro. Development of humanoid robot HRP-3P. In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 50–55. IEEE, 2005.

[6] Baris Akgun, Maya Cakmak, Jae Wook Yoo, and Andrea Lockerd Thomaz. Trajectories and keyframes for kinesthetic teaching: a human-robot interaction perspective. In *HRI*, 2012.

[7] R. Alami, L. Aguilar, H. Bullata, S. Fleury, M. Herrb, F. Ingrand, M. Khatib, and F. Robert. A general framework for multi-robot cooperation and its implementation on a set of three hilare robots. *Experimental Robotics IV*, pages 26–39, 1997.

[8] R. Alami, A. Albu-Schaeffer, A. Bicchi, R. Bischoff, R. Chatila, A. De Luca, A. De Santis, G. Giralt, J. Guiochet, G. Hirzinger, F. Ingrand, V. Lippiello, R. Mattone, D. Powell, S. Sen, B. Siciliano, G. Tonietti, and L. Villani. Safe and Dependable Physical Human-Robot Interaction in Anthropic Domains: State of the Art and Challenges. In *IROS Workshop on pHRI*, 2006.

[9] Rachid Alami, AurÃĺlie Clodic, Vincent Montreuil, Emrah Akin Sisbot, and Raja Chatila. Toward human-aware robot task planning. In *AAAI Spring Symposium*, pages 39–46, 2006.

[10] A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger. The DLR lightweight robot: design and control concepts for robots in human environments. *Industrial Robot: An International Journal*, 34(5):376–385, 2007.

[11] A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone. A versatile computer-controlled assembly system. In *Proceedings of the 3rd Int. Joint Conference on Artificial Intelligence*, pages 298–307, 1973.

[12] S. J. Anderson, S. C. Peters, K. Iagnemma, and J. Overholt. Semi-autonomous stability control and hazard avoidance for manned and unmanned ground vehicles. In *MIT, Dept. of Mechanical Eng.*, 2010.

[13] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[14] Akiko Arita, Kazuo Hiraki, Takayuki Kanda, and Hiroshi Ishiguro. Can we talk to robots? ten-month-old infants expected interactive humanoid robots to be talked to by persons. *Cognition*, 95, 2005.

[15] Ronald C Arkin. The ethics of robotic deception. *The Computational Turn: Past, Present, Futures?*, 2011.

[16] Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. Action understanding as inverse planning appendix. *Cognition*, 2009.

[17] Dare A. Baldwin, Jodie A. Baird, Megan M. Saylor, and M. Angela Clark. Infants parse dynamic action. *Child Development*, 72(3):708–717, 2001.

[18] Jérôme Barraquand and Jean-Claude Latombe. A Monte-Carlo algorithm for path planning with many degrees of freedom. *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1712–1717, 1990.

[19] M. Beetz, L. Mošenlechner, and M. Tenorth. CRAM: A cognitive robot abstract machine for everyday manipulation in human environments. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1012–1017. IEEE, 2010.

[20] Michael Beetz, Freek Stulp, Piotr Esden-Tempski, Andreas Fedrizzi, Ulrich Klank, Ingo Kresse, Alexis Maldonado, and Federico Ruiz. Generality and legibility in mobile manipulation. *Autonomous Robots*, 28:21–44, 2010.

[21] Tanya Behne, Malinda Carpenter, Josep Call, and Michael Tomasello. Unwilling Versus Unable: Infants' Understanding of Intentional Action. *Developmental Psychology*, 41:328–337, 2005.

[22] D. Berenson, S. S. Srinivasa, D. Ferguson, A. Collet, and J. J. Kuffner. Manipulation planning with workspace goal regions. In *IEEE International Conference on Robotics and Automation*, 2009.

[23] Dmitry Berenson, Siddhartha Srinivasa, David Ferguson, Alvaro Collet Romea, and James Kuffner. Manipulation planning with workspace goal regions. In *IEEE International Conference on Robotics and Automation*, May 2009.

[24] G.R. Bergersen, J.E. Hannay, D.I.K. Sjoberg, T. Dyba, and A. Karahasanovic. Inferring skill from tests of programming performance: Combining time and quality. In *ESEM*, 2011.

[25] D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[26] Dominik Bertram, James Kuffner, Ruediger Dillmann, and Tamim Asfour. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *ICRA*, 2006.

[27] Celeste Biever. Deceptive robots show theory of mind. *New Scientist*, 207(2779):24–25, 2010.

[28] John Blitzer and H Daume. Icml tutorial on domain adaptation, 2010.

[29] Michael S Branicky, Ross A Knepper, and James J Kuffner. Path and trajectory diversity: Theory and algorithms. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1359–1364. IEEE, 2008.

[30] M. Bratman. Shared cooperative activity. *Philosophical Review*, 1992.

[31] Bambi R Brewer, Roberta L Klatzky, and Yoky Matsuoka. Visual-feedback distortion in a robotic rehabilitation environment. *Proceedings of the IEEE*, 94(9):1739–1751, 2006.

[32] O. Brock and O. Khatib. Elastic strips: A framework for motion generation in human environments. *The International Journal of Robotics Research*, 21(12):1031, 2002.

[33] Sylvain Calinon, Florent Guenter, and Aude Billard. On learning, representing, and generalizing a task in a humanoid robot. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(2):286–298, 2007.

[34] M. Carpenter, Nagell K., Tomasello, G. M., Butterworth, and C. Moore. Social cognition, joint attention, and communcative competence from 9 to 15 months of age. *Monographs of the Society for Research in Child Development*, 63(4):1–174.

[35] Arancha Casal. *Reconfiguration planning for modular self-reconfigurable robots*. PhD thesis, Aeronautics and Astronautics Dept., Stanford U., 2001.

[36] Jinxiang Chai and Jessica K. Hodgins. Constraint-based motion optimization using a statistical dynamic model. *ACM Trans. Graph.*, 26(3), July 2007.

[37] Eugene Charniak and Robert P. Goldman. A bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53 – 79, 1993.

[38] M. Ciocarlie, K. Hsiao, E.G. Jones, S. Chitta, R.B. Rusu, and I.A. Sucan. Towards reliable grasping and manipulation in household environments. In *Proceedings of RSS 2010 Workshop on Strategies and Evaluation for Mobile Manipulation in Household Environments*, 2010.

[39] Roberto Cipolla and Nicholas J Hollinghurst. Human-robot interface by pointing with uncalibrated stereo vision. *Image and Vision Computing*, 14(3):171–178, 1996.

[40] Alvaro Collet, Dmitry Berenson, Siddhartha S. Srinivasa, and Dave Ferguson. Object recognition and full pose registration from a single image for robotic manipulation. In *IEEE International Conference on Robotics and Automation*, pages 48–55, Kobe, May 2009.

[41] Alvaro Collet, Manuel Martinez, and Siddhartha S. Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *International Journal of Robotics Research*, 30(10):1284–1306, 2011.

[42] Alvaro Collet and Siddhartha S. Srinivasa. Efficient multi-view object recognition and full pose estimation. In *IEEE International Conference on Robotics and Automation*, Anchorage, May 2010.

[43] Alvaro Collet, Siddhartha S. Srinivasa, and Martial Hebert. Structure discovery in multi-modal data: a region-based approach. In *IEEE International Conference on Robotics and Automation*, Shanghai, May 2011.

[44] J. Cortes and T Simeon. Sampling-based motion planning under kinematic loop-closure constraints. In *Proc. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2004.

[45] J.W. Crandall and M.A. Goodrich. Characterizing efficiency of human robot interaction: a case study of shared-control teleoperation. In *IROS*, 2002.

[46] G. Csibra and Gy. Gergely. The teleological origins of mentalistic action explanations: A developmental hypothesis. *Developmental Science*, 1:255–259, 1998.

[47] Gergely Csibra and GyÃűrgy Gergely. Obsessed with goals: Functions and mechanisms of teleological interpretation of actions in humans. *Acta Psychologica*, 124(1):60 – 78, 2007.

[48] RALPH B D'AGOSTINO. A second look at analysis of variance on dichotomous data. *Journal of Educational Measurement*, 8(4):327–333, 1971.

[49] T. Debus, J. Stoll, R.D. Howe, and P. Dupont. Cooperative human and machine perception in teleoperated assembly. In *ISER*, 2000.

[50] Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Clarifying commands with information-theoretic human-robot dialog. *Journal of Human-Robot Interaction*, 2013.

[51] Y. Demiris and G. Hayes. Imitation as a dual-route process featuring predictive and learning components: a biologically plausible computational model. In *Imitation in animals and artifacts*, 2002.

[52] Munjal Desai, Mikhail Medvedev, Marynel Vázquez, Sean McSheehy, Sofia Gadea-Omelchenko, Christian Bruggeman, Aaron Steinfeld, and Holly Yanco. Effects of changing reliability on trust of robot systems. In *HRI*, 2012.

[53] Michael Dewar. *The art of deception in warfare.* David & Charles Publishers, 1989.

[54] Debadeepta Dey, Tian Y Liu, Boris Sofman, and Drew Bagnell. Efficient optimization of control libraries. Technical report, DTIC Document, 2011.

[55] Rosen Diankov. *Automated Construction of Robotics Manipulation Programs*. PhD thesis, Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213, 10 2010.

[56] A.D. Dragan, S. Bauman, J. Forlizzi, and S.S. Srinivasa. Effects of robot motion on human-robot collaboration. In *International Conference on Human-Robot Interaction (HRI)*, 2015.

[57] A.D. Dragan, G. Gordon, and S. Srinivasa. Learning from experience in manipulation planning: Setting the right goals. In *ISRR*, 2011.

[58] A.D. Dragan, R. Holladay, and S.S. Srinivasa. An analysis of deceptive robot motion. In *Robotics: Science and Systems (R:SS)*, 2014.

[59] A.D. Dragan, R. Holladay, and S.S. Srinivasa. From legibility to deception. *Autonomous Robotics*, 2015.

[60] A.D. Dragan, K.T. Lee, and S.S. Srinivasa. Legibility and predictability of robot motion. In *International Conference on Human-Robot Interaction (HRI)*, 2013.

[61] A.D. Dragan, K. Muelling, J.A. Bagnell, and S.S. Srinivasa. Movement primitives via optimization. In *International Conference on Robotics and Automation (ICRA)*, 2015.

[62] A.D. Dragan, N. Ratliff, and S.S. Srinivasa. Manipulation planning with goal sets using constrained trajectory optimization. In *ICRA*, May 2011.

[63] A.D. Dragan and S.S. Srinivasa. Formalizing assistive teleoperation. In *Robotics: Science and Systems (R:SS)*, Sydney, Australia, July 2012.

[64] A.D. Dragan and S.S. Srinivasa. Generating legible motion. In *Robotics: Science and Systems (R:SS)*, Berlin, Australia, June 2013.

[65] A.D. Dragan and S.S. Srinivasa. Familiarization to robot motion. In *International Conference on Human-Robot Interaction (HRI)*, 2014.

[66] Anca Dragan and Siddhartha Srinivasa. Generating legible motion. In *Robotics: Science and Systems*, 2013.

[67] Anca D Dragan, Kenton CT Lee, and Siddhartha S Srinivasa. Legibility and predictability of robot motion. In *HRI*, 2013.

[68] E. Drumwright and V. Ng-Thow-Hing. Toward interactive reaching in static environments for humanoid robots. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[69] A. H. Fagg, M. Rosenstein, R. Platt, and R. A. Grupen. Extracting user intent in mixed initiative teleoperator control. In *AIAA*, 2004.

[70] Jing Fan, Jiping He, and Stephen Tillery. Control of hand orientation and arm movement during reach and grasp. *Experimental Brain Research*, 171:283–296, 2006.

[71] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1778–1785. IEEE, 2009.

[72] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, 1998.

[73] T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *J Neurosci.*, 5:1688–1703, July 1985.

[74] Roger Flynn. Anticipation and deception in squash. In *9th Squash Australia/PSCAA National Coaching conference*, 1996.

[75] Mike Fraser, Steve Benford, Jon Hindmarsh, and Christian Heath. Supporting awareness and interaction through collaborative virtual interfaces. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, pages 27–36. ACM, 1999.

[76] Andrea Frome, Yoram Singer, and Jitendra Malik. Image retrieval and classification using local distance functions. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, volume 19, page 417. Mit Press, 2007.

[77] G. Gergely, H. Bekkering, and I. Kiraly. Rational imitation in preverbal infants. *Nature*, 415(6873), 2002.

[78] GyÃűrgy Gergely, Zoltan Nadasdy, Gergely Csibra, and Szilvia Biro. Taking the intentional stance at 12 months of age. *Cognition*, 56(2):165 – 193, 1995.

[79] M. Gielniak, K. Liu, and A. L. Thomaz. Secondary action in robot motion. In *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 2010)*, 2010.

[80] M. Gielniak, K. Liu, and A. L. Thomaz. Task aware variance for robot motion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2011)*, 2011.

[81] M. Gielniak and A. L. Thomaz. Enhancing interaction through exaggerated motion synthesis. In *ACM/IEEE HRI*.

[82] M. Gielniak and A. L. Thomaz. Spatiotemporal correspondence as a metric for human-like robot motion. In *ACM/IEEE HRI*, 2011.

[83] Michael J Gielniak and Andrea Lockerd Thomaz. Generating anticipation in robot motion. In *RO-MAN*, 2011.

[84] M.J. Gielniak and A.L. Thomaz. Generating anticipation in robot motion. In *RO-MAN*, pages 449 –454, 31 2011-aug. 3 2011.

[85] Michael Gleicher. Retargeting motion to new characters. In *Proceedings of ACM SIGGRAPH 98*, Annual Conference Series, pages 33–42. ACM SIGGRAPH, jul 1998.

[86] Steven J Glynn and Robert A Henning. Can teams outperform individuals in a simulated dynamic control task. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 44(33):141–144, 2000.

[87] Rachel Gockley, Jodi Forlizzi, and Reid Simmons. Natural person-following behavior for social robots. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, HRI '07, pages 17–24, New York, NY, USA, 2007. ACM.

[88] R.C. Goertz. Manipulators used for handling radioactive materials. *Human factors in technology*, 1963.

[89] Mehmet Gokturk and John L Sibert. An analysis of the index finger as a pointing device. In *CHI'99 Extended Abstracts on Human Factors in Computing Systems*, pages 286–287. ACM, 1999.

[90] Noah D Goodman and Andreas Stuhlmüller. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in Cognitive Science*, 5(1):173–184, 2013.

[91] Sami Haddadin, A Albu-Schaffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3356–3363. IEEE, 2008.

[92] J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960.

[93] J. M. Hammersley. Monte-Carlo methods for solving multivariable problems. *Annals of the New York Academy of Science*, 86:844–874, 1960.

[94] PA Hancock, DR Billings, and KE Schaefer. Can you trust your robot? *Ergonomics in Design: The Quarterly of Human Factors Applications*, 19(3):24–29, 2011.

[95] J. Heinzmann and A. Zelinsky. The safe control of human-friendly robots. In *IEEE/RSJ IROS*, 1999.

[96] Martin Herrmann and Siddhartha S. Srinivasa. Exploiting passthrough information for multi-view object reconstruction with sparse and noisy laser data. Technical Report CMU-RI-TR-10-07, Robotics Institute, Pittsburgh, PA, February 2010.

[97] Jon Hindmarsh, Mike Fraser, Christian Heath, Steve Benford, and Chris Greenhalgh. Fragmented interaction: establishing mutual orientation in virtual environments. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 217–226. ACM, 1998.

[98] G Hoffman. Evaluating fluency in human-robot collaboration. In *HRI Workshop on Human Robot Collaboration*, 2013.

[99] R. Holladay, A.D. Dragan, and S.S. Srinivasa. Legible robot pointing. In *International Symposium on Human and Robot Communication (Ro-Man)*, 2014.

[100] David Hsu. *Randomized single-query motion planning in expansive spaces*. PhD thesis, Computer Science Dept., Stanford University, 2000.

[101] David Hsu, Jean-Claude Latombe, and Rajeev Motwani. Path planning in expansive configuration spaces. In *Proc. Conf. IEEE Int Robotics and Automation*, volume 3, pages 2719–2726, 1997.

[102] Tian Huang, Zhanxian Li, Meng Li, Derek G Chetwynd, and Clement M Gosselin. Conceptual design and dimensional synthesis of a novel 2-dof translational parallel robot for pick-and-place operations. *Journal of Mechanical Design*, 126:449, 2004.

[103] C. Igel, M. Toussaint, and W. Weishui. Rprop using the natural gradient. *Trends and Applications in Constructive Approximation*, pages 259–272, 2005.

[104] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373, 2013.

[105] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In *NIPS*, 2003.

[106] Robin C Jackson, Simon Warren, and Bruce Abernethy. Anticipation skill and susceptibility to deceptive movement. *Acta psychologica*, 123(3):355–371, 2006.

[107] A. Jain and C.C. Kemp. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonomous Robots*, 28(1):45–64, 2010.

[108] Nikolay Jetchev and Marc Toussaint. Trajectory prediction: learning to map situations to robot trajectories. In *Proceedings of the 26th annual international conference on machine learning*, pages 449–456. ACM, 2009.

[109] Nikolay Jetchev and Marc Toussaint. Trajectory prediction in cluttered voxel environments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2523–2528. IEEE, 2010.

[110] Thierry Simeon Jim Mainprice, E. Akin Sisbot and Rachid Alami. Planning safe and legible hand-over motions for human-robot interaction. In *IARP Workshop on Technical Challenges for Dependable Robots in Human Environments*, 2010.

[111] Kwang jin Choi and Hyeong seok Ko. On-line motion retargetting. *Journal of Visualization and Computer Animation*, 11:223–235, 1999.

[112] S. Kagami, K. Nishiwaki, J.J. Kuffner Jr, Y. Kuniyoshi, M. Inaba, and H. Inoue. Online 3d vision, motion planning and bipedal locomotion control coupling system of humanoid robot: H7. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2557–2562. IEEE, 2002.

[113] Peter H Kahn Jr, Takayuki Kanda, Hiroshi Ishiguro, Brian T Gill, Jolina H Ruckert, Solace Shen, Heather E Gary, Aimee L Reichert, Nathan G Freier, and Rachel L Severson. Do people hold a humanoid robot morally accountable for the harm it causes? In *International conference on Human-Robot Interaction*, pages 33–40, 2012.

[114] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. STOMP: Stochastic trajectory optimization for motion planning. In *Proc. IEEE Int Robotics and Automation (ICRA) Conf*, pages 4569–4574, 2011.

[115] Kazunori Kamewari, Masaharu Kato, Takayuki Kanda, Hiroshi Ishiguro, and Kazuo Hiraki. Six-and-a-half-month-old children positively attribute goals to human action and to humanoid-robot motion. *Cognitive Development*, 20(2):303 – 320, 2005.

[116] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot HRP-2. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 2, pages 1083–1090. IEEE, 2004.

[117] K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, and S. Kajita. Cybernetic human HRP-4C. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 7–14. IEEE, 2009.

[118] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[119] Lydia E. Kavraki, Petr Švestka, Jean-Claude Latombe, and Mark H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[120] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal. Coordination and decentralized cooperation of multiple mobile manipulators. *Journal of Robotic Systems*, 13(11):755–764, 1996.

[121] Cory D Kidd and Cynthia Breazeal. Human-robot interaction experiments: Lessons learned. In *Proceeding of AISB*, volume 5, pages 141–142, 2005.

[122] D.-J. Kim, R. Hazlett-Knudsen, H. Culver-Godfrey, G. Rucks, T. Cunningham, D. PortÃľ ande, J. Bricout, Z. Wang, and A. Behal. How autonomy impacts performance and satisfaction: Results from a study with spinal cord injured subjects using an assistive robot. *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2011.

[123] Robert Kindel. *Motion planning for free-flying robots in dynamic and uncertain environments*. PhD thesis, Aeronaut. & Astr. Dept., Stanford University, 2001.

[124] Sotaro Kita. *Pointing: Where language, culture, and cognition meet*. Psychology Press, 2003.

[125] G. Klien, D.D. Woods, J.M. Bradshaw, R.R. Hoffman, and P.J. Feltovich. Ten challenges for making automation a "team player" in joint human-agent activity. *Intelligent Systems*, 19(6):91 – 95, nov.-dec. 2004.

[126] Ross Knepper, Siddhartha S. Srinivasa, and Matthew Mason. Hierarchical Planning Architectures for Mobile Manipulation Tasks in Indoor Environments. In *IEEE International Conference on Robotics and Automation*, Anchorage, 2010. IEEE.

[127] Jens Kober, Erhan Oztop, and Jan Peters. Reinforcement learning to adjust robot movements to new situations. In *IJCAI*, 2011.

[128] Jens Kober and Jan Peters. Learning motor primitives for robotics. In *ICRA*, 2009.

[129] J. Kofman, X. Wu, T.J. Luu, and S. Verma. Teleoperation of a robot manipulator using a vision-based human-robot interface. *IEEE Trans. on Industrial Electronics*, 2005.

[130] Yoshihito Koga, Koichi Kondo, James Kuffner, and Jean claude Latombe. Planning motions with intentions. In *SIGGRAPH*, 1994.

[131] Takanori Komatsu and Seiji Yamada. Adaptation gap hypothesis: How differences between users' expected and perceived agent functions affect their subjective impression. *Journal of Systemics, Cybernetics and Informatics*, 9(1):67–74, 2011.

[132] George Konidaris and Andrew Barto. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 489–496. ACM, 2006.

[133] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Robot motor skill coordination with em-based reinforcement learning. In *IROS*, 2010.

[134] James J. Kuffner. *Autonomous agents for real-time animation*. PhD thesis, Computer Science Dept., Stanford University, 1999.

[135] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *ICRA*, 2000.

[136] J.J. Kuffner and S.M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, pages 995–1001, San Francisco, CA, April 2000.

[137] Dana Kulić and Elizabeth A Croft. Safe planning for human-robot interaction. *Journal of Robotic Systems*, 22(7):383–396, 2005.

[138] F Lacquaniti and JF. Soechting. Coordination of arm and wrist motion during a reaching task. *J Neurosci.*, 2:399–408, April 1982.

[139] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 951–958. IEEE, 2009.

[140] John Lasseter. Principles of traditional animation applied to 3d computer animation. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 35–44, New York, NY, USA, 1987. ACM.

[141] Steven M. LaValle and James J. Kuffner. Rapidly-exploring random trees: Progress and prospects. *Algorithmic and Computational Robotics: New Directions*, pages 293–308, 2001.

[142] Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 39–48, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[143] Min Kyung Lee, Sara Kiesler, Jodi Forlizzi, Siddhartha Srinivasa, and Paul Rybski. Gracefully mitigating breakdowns in robotic services. In *HRI*, 2010.

[144] A. E. Leeper, K. Hsiao, M. Ciocarlie, L. Takayama, and D. Gossow. Strategies for human-in-the-loop robotic grasping. In *HRI*, 2012.

[145] M. Li and A.M. Okamura. Recognition of operator motions for real-time assistance using virtual fixtures. In *HAPTICS*, 2003.

[146] Christina Lichtenthäler, Tamara Lorenz, and Alexandra Kirsch. Towards a legibility metric: How to measure the perceived value of a robot. In *ICSR Work-In-Progress-Track*, 2011.

[147] C. Karen Liu, Aaron Hertzmann, and Zoran Popović. Learning physics-based motion style with nonlinear inverse optimization. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1071–1081, New York, NY, USA, 2005. ACM.

[148] S.G. Loizou and V. Kumar. Mixed initiative control of autonomous vehicles. In *ICRA*, 2007.

[149] T. Lozano-Perez, J. Jones, E. Mazer, P. O'Donnell, W. Grimson, P. Tournassoud, and A. Lanusse. Handey: A robot system that recognizes, plans, and manipulates. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 843–849. IEEE, 1987.

[150] T. Lozano-Perez, J.L. Jones, E. Mazer, and P.A. O'Donnell. Handey: a robot task planner. 1992.

[151] P Maes, M Mataric, J Meyer, J Pollack, and S Wilson. Self-taught visually-guided pointing for a humanoid robot.

[152] Jim Mainprice and Dmitry Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 299–306. IEEE, 2013.

[153] Jim Mainprice, E Akin Sisbot, Thierry Siméon, and Rachid Alami. Planning safe and legible hand-over motions for human-robot interaction. In *IARP workshop on technical challenges for dependable robots in human environments*, volume 2, page 7, 2010.

[154] P. Marayong, Ming Li, A.M. Okamura, and G.D. Hager. Spatial motion constraints: theory and demonstrations for robot guidance using virtual fixtures. In *ICRA*, 2003.

[155] P. Marayong, A. M. Okamura, and A. Bettini. Effect of virtual fixture compliance on human-machine cooperative manipulation. In *IROS*, 2002.

[156] Michelle A Marks, Mark J Sabella, C Shawn Burke, and Stephen J Zaccaro. The impact of cross-training on team effectiveness. *Journal of Applied Psychology*, 87(1):3, 2002.

[157] Sean R Martin, Steve E Wright, and John W Sheppard. Offline and online evolutionary bi-directional rrt algorithms for efficient re-planning in dynamic environments. In *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, pages 1131–1136. IEEE, 2007.

[158] Manuel Martinez, Alvaro Collet, and Siddhartha S. Srinivasa. MOPED: A scalable and low latency object recognition and pose estimation system. In *IEEE International Conference on Robotics and Automation*, Anchorage, 2010.

[159] David H. Mayne and David Q. Jacobson. *Differential dynamic programming*. New York: American Elsevier Pub. Co., 1970.

[160] A. N. Meltzoff. Understanding the intentions of others: Re-enactment of intended acts by 18-month-old children. *Developmental Psychology*, 31(5):838–850, 1995.

[161] David P Miller. Assistive robotics: an overview. In *Assistive Technology and Artificial Intelligence*, pages 126–136. 1998.

[162] Rosamond Mitchell and Florence Myles. *Second language learning theories*. 2004.

[163] B. Mutlu, J. Forlizzi, and J. Hodgins. A storytelling robot: Modeling and evaluation of human-like gaze behavior. In *Humanoid Robots*, 2006.

[164] Bilge Mutlu and Jodi Forlizzi. Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction. In *HRI*, 2008.

[165] Bilge Mutlu, Jodi Forlizzi, and Jessica Hodgins. A storytelling robot: Modeling and evaluation of human-like gaze behavior. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 518–523. IEEE, 2006.

[166] Jun Nakanishi, Jun Morimoto, Gen Endo, Gordon Cheng, Stefan Schaal, and Mitsuo Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems*, 47(2):79–91, 2004.

[167] Stefanos Nikolaidis and Julie Shah. Human-robot teaming using shared mental models. In *ACM/IEEE HRI*, 2012.

[168] NJ Nilsson. A mobile automation: An application of artificial intelligence techniques. In *Proceedings of the 1st Int. Joint Conference on Artificial Intelligence*, pages 509–520, 1969.

[169] K. Nishiwaki, T. Sugihara, S. Kagami, F. Kanehiro, M. Inaba, and H. Inoue. Design and development of research platform for perception-action integration in humanoid robot: H6. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 3, pages 1559–1564. IEEE, 2000.

[170] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*, pages 1410–1418, 2009.

[171] Jia Pan, Sachin Chitta, and Dinesh Manocha. FCL: A general purpose library for proximity and collision queries. In *ICRA*, 2012.

[172] Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *ICRA*, 2009.

[173] Peter Pastor, Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, and Stefan Schaal. Skill learning and task outcome prediction for manipulation. In *ICRA*, 2011.

[174] Peter Pastor, Ludovic Righetti, Mrinal Kalakrishnan, and Stefan Schaal. Online movement adaptation based on previous sensor experiences. In *IROS*, 2011.

[175] Stéphane Petti and Thierry Fraichard. Safe motion planning in dynamic environments. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2210–2215. IEEE, 2005.

[176] Giovanni Pezzulo, Francesco Donnarumma, and Haris Dindo. Human sensorimotor communication: a theory of signaling in online social interactions. *PloS one*, 8(11):e79876, 2013.

[177] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, 1999.

[178] L.S. Pontryagin. *The mathematical theory of optimal processes*. Interscience New York, 1962.

[179] Miguel Prada, Anthony Remazeilles, Ansgar Koene, and Satoshi Endo. Dynamic movement primitives for human-robot interaction: comparison with human behavioral observation. In *IROS*, 2013.

[180] Samuel Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 2009.

[181] M. Quigley, E. Berger, and A.Y. Ng. Stair: Hardware and software architecture. In *AAAI 2007 Robotics Workshop, Vancouver, BC*, 2007.

[182] Sean Quinlan. *The Real-Time Modification of Collision-Free Paths*. PhD thesis, Stanford University, 1994.

[183] N. Ratliff, J. A. Bagnell, and M. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning (ICML)*, 2006.

[184] Nathan Ratliff, Matthew Zucker, J. Andrew (Drew) Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *ICRA*, May 2009.

[185] Nathan D. Ratliff, Matthew Zucker, J. Andrew Bagnell, and Siddhartha S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *IEEE International Conference on Robotics and Automation*, pages 489–494. IEEE, 2009.

[186] Monica Reggiani, Mirko Mazzoli, and Stefano Caselli. An experimental evaluation of collision detection packages for robot motion planning, 2002.

[187] L.B. Rosenberg. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Virtual Reality Annual International Symposium*, 1993.

[188] Alla Safonova, Jessica K. Hodgins, and Nancy S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 514–521, New York, NY, USA, 2004. ACM.

[189] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent ASIMO: System overview and integration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2478–2483. Ieee, 2002.

[190] Eri Sato, Toru Yamaguchi, and Fumio Harashima. Natural interface using pointing behavior for human–robot gestural interaction. *Industrial Electronics, IEEE Transactions on*, 54(2):1105–1112, 2007.

[191] Allison Sauppé and Bilge Mutlu. Robot deictics: How gesture and context shape referential communication. 2014.

[192] C. F. Schmidt and J. D'Addamio. A model of the common-sense theory of intention and personal causation. In *IJCAI*, 1973.

[193] John Schulman, Jonathan Ho, Cameron Lee, and Pieter Abbeel. Learning from demonstrations through the use of non-rigid registration. In *ISRR*, 2013.

[194] J. Shen, J. Ibanez-Guzman, T. C. Ng, and B. S. Chew. A collaborative-shared control system with safe obstacle avoidance capability. In *RAM*, 2004.

[195] Jaeeun Shim and Ronald C Arkin. A taxonomy of robot deception and its benefits in hri. 2013.

[196] E. Short, J. Hart, M. Vu, and B. Scassellati. No fair!! an interaction with a cheating robot. 2010.

[197] Elaine Short, Justin Hart, Michelle Vu, and Brian Scassellati. No fair!! an interaction with a cheating robot. In *International Conference on Human-Robot Interaction (HRI)*, pages 219–226, 2010.

[198] Rosanne M Siino and Pamela J Hinds. Robots, gender & sensemaking: Sex segregation's impact on workers making sense of a mobile autonomous robot. In *ICRA*, 2005.

[199] Thierry Siméon, Jean-Paul Laumond, Juan Cortés, and Anis Sahbani. Manipulation planning with probabilistic roadmaps. *International Journal of Robotics Research*, 23(7–8):729–746, July-August 2004.

[200] Emrah Akin Sisbot, Luis Felipe Marin-Urias, Rachid Alami, and Thierry Simeon. A human aware mobile robot motion planner. *Robotics, IEEE Transactions on*, 23(5):874–883, 2007.

[201] NJ Smeeton and AM Williams. The role of movement exaggeration in the anticipation of deceptive soccer penalty kicks. *British Journal of Psychology*, 103(4):539–555, 2012.

[202] C. Smith, M. Bratt, and H.I. Christensen. Teleoperation for a ball-catching task with significant dynamics. *Neural Networks*, 21(4):604 – 620, 2008.

[203] Beate Sodian, Barbara Schoeppner, and Ulrike Metz. Do infants apply the principle of rational action to human agents? *Infant Behavior and Development*, 27(1):31 – 41, 2004.

[204] S.S. Srinivasa, D. Berenson, M. Cakmak, A. Collet, M.R. Dogar, A.D. Dragan, R.A. Knepper, T. Niemueller, K. Strabala, M. Vande Weghe, and J. Ziegler. Herb 2.0: Lessons learned from developing a mobile manipulator for the home. *Proc. of the IEEE, Special Issue on Quality of Life Technology*, 2012.

[205] M. Stilman. Task constrained motion planning in robot joint space. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[206] Martin Stolle and Christopher G Atkeson. Policies based on trajectory libraries. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3344–3349. IEEE, 2006.

[207] Martin Stolle, Hanns Tappeiner, Joel Chestnutt, and Christopher G Atkeson. Transfer of policies based on trajectory libraries. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 2981–2986. IEEE, 2007.

[208] Kristen Stubbs, David Wettergreen, and Illah Nourbakhsh. Using a robot proxy to create common ground in exploration tasks. In *HRI*, 2008.

[209] Karin Sundin, Lilian Jansson, and Astrid Norberg. Communicating with people with stroke and aphasia: understanding through sensation without words. *Journal of clinical nursing*, 9(4):481–488, 2000.

[210] Leila Takayama, Doug Dooley, and Wendy Ju. Expressing thought: improving robot readability with animation principles. In *HRI*, 2011.

[211] Leila Takayama, Doug Dooley, and Wendy Ju. Expressing thought: improving robot readability with animation principles. In *Proceedings of the 6th international conference on Human-robot interaction*, pages 69–76. ACM, 2011.

[212] Stefanie Tellex, Ross Knepper, Adrian Li, Daniela Rus, and Nicholas Roy. Asking for help using inverse semantics.

[213] Kazunori Terada and Akira Ito. Can a robot deceive humans? In *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*, pages 191–192. IEEE, 2010.

[214] A. L. Thomaz and M. Cakmak. Learning about objects with human teachers. In *HRI*, 2009.

[215] A. M. Thompson. The navigation system of the JPL robot. In *Proceedings of the 5th Int. Joint Conference on Artificial Intelligence*, pages 749–757, 1977.

[216] E. Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005. Proceedings of the 2005*, pages 300 – 306 vol. 1, june 2005.

[217] Deepak Tolani, Ambarish Goswami, and Norman I Badler. Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical models*, 62(5):353–388, 2000.

[218] M. Tomasello, M. Carptenter, J. Call, T. Behne, and H. Moll. Understanding and sharing intentions: the origins of cultural cognition. *Behavioral and Brain Sciences*, 2004.

[219] M. Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1049–1056. ACM, 2009.

[220] Sandra Upson. tongue vision. *Spectrum, IEEE*, 44(1):44–45, 2007.

[221] Jur Van Den Berg, Dave Ferguson, and James Kuffner. Anytime path planning and replanning in dynamic environments. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 2366–2371. IEEE, 2006.

[222] D. Vasquez, T. Fraichard, O. Aycard, and C. Laugier. Intentional motion on-line learning and prediction. *Machine Vision and Applications*, 2005.

[223] Marynel Vázquez, Alexander May, Aaron Steinfeld, and Wei-Hsuan Chen. A deceptive robot referee in a multiplayer gaming environment. In *Collaboration Technologies and Systems (CTS), 2011 International Conference on*, pages 204–211. IEEE, 2011.

[224] Manuela M Veloso. Learning by analogical reasoning in general problem solving. Technical report, 1992. Doctoral Dissertation.

[225] Cordula Vesper, Stephen Butterfill, Günther Knoblich, and Natalie Sebanz. 2010 special issue: A minimal architecture for joint action. *Neural Netw.*, 23(8-9):998–1003, October 2010.

[226] Adam Vogel, Christopher Potts, and Dan Jurafsky. Implicatures and nested beliefs in approximate Decentralized-POMDPs. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[227] Karl E Weick. *Sensemaking in organizations*, volume 3. 1995.

[228] R. Weinstock. *Calculus of variations*. Dover publications, 1974.

[229] Andrew Witkin and Michael Kass. Spacetime constraints. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '88, pages 159–168, New York, NY, USA, 1988. ACM.

[230] Nelson Wong and Carl Gutwin. Where are you pointing?: the accuracy of deictic pointing in cves. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1029–1038. ACM, 2010.

[231] J. H. Yakey, S. M. LaValle, and L. E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, 17(6):951–958, 2001.

[232] K. Yamane, J.J. Kuffner, and J.K. Hodgins. Synthesizing animations of human manipulation tasks. In *SIGGRAPH*, 2004.

[233] Zhenwang Yao and K. Gupta. Path planning with general end-effector constraints: using task space to guide configuration space search. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005.

[234] Gu Ye and Ron Alterovitz. Demonstration-guided motion planning. In *ISRR*, 2011.

[235] E. You and K. Hauser. Assisted teleoperation strategies for aggressively controlling a robot arm with 2d input. In *R:SS*, 2011.

[236] Wentao Yu, R. Alqasemi, R. Dubey, and N. Pernalete. Telemanipulation assistance based on motion intention recognition. In *ICRA*, 2005.

[237] M. Zefran and V. Kumar. A variational calculus framework for motion planning. In *IEEE Conference on Advanced Robotics*, pages 415–420, Monterey, CA, 1997.

[238] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

[239] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa. Planning-based prediction for pedestrians. In *IROS*, 2009.

[240] M. Zucker, N. Ratliff, A.D. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. Bagnell, and S.S. Srinivasa. Covariant hamiltonian optimization for motion planning. *International Journal of Robotics Research (IJRR)*, 2013.