# Toward Enabling Safe & Efficient Human-Robot Manipulation in Shared Workspaces

by

Rafi Hayne

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

August 2016

APPROVED:

Professor Dmitry Berenson, Major Thesis Adviser

Professor Craig Wills, Head of CS Department

## Abstract

When humans interact, there are many avenues of physical communication available ranging from vocal to physical gestures. In our past observations, when humans collaborate on manipulation tasks in shared workspaces there is often minimal to no verbal or physical communication, yet the collaboration is still fluid with minimal interferences between partners. However, when humans perform similar tasks in the presence of a robot collaborator, manipulation can be clumsy, disconnected, or simply not human-like. The focus of this work is to leverage our observations of human-human interaction in a robot's motion planner in order to facilitate more safe, efficient, and human-like collaborative manipulation in shared workspaces.

We first present an approach to formulating the cost function for a motion planner intended for human-robot collaboration such that robot motions are both safe and efficient. To achieve this, we propose two factors to consider in the cost function for the robot's motion planner: (1) Avoidance of the workspace previously-occupied by the human, so robot motion is safe as possible, and (2) Consistency of the robot's motion, so that the motion is predictable as possible for the human and they can perform their task without focusing undue attention on the robot. Our experiments in simulation and a human-robot workspace sharing study compare a cost function that uses only the first factor and a combined cost that uses both factors vs. a baseline method that is perfectly consistent but does not account for the human's previous motion. We find using either cost function we outperform the baseline method in terms of task success rate without degrading the task completion time. The best task success rate is achieved with the cost function that includes both the avoidance and consistency terms.

Next, we present an approach to human-attention aware robot motion generation which attempts to convey intent of the robot's task to its collaborator. We capture human attention through the combined use of a wearable eye-tracker and motion capture system. Since human attention isn't static, we present a method of generating a motion policy that can be queried online. Finally, we show preliminary tests of this method.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This work studies the role of motion generation in collaborative manipulation between humans and robots in a shared workspace; as well as how planning with different strategies for robot motion can have an effect on safety and efficiency. Although the field of robotics has been growing substantially in recent history, few people feel comfortable working around a robot in shared workspaces - regardless of if the robot has been designed to work in an industrial or collaborative setting. Certainly, there is an inherent danger in working in close proximity to industrial robots that can move at alarming speeds. Yet still, even when a robot is custom designed to be deployed in close proximity to human collaborators, many feel uneasy. This uneasiness does not come solely from the presence of a non-human collaborator as most are enamored by the site of a robot held statically in a configuration. Instead, this uneasiness rears itself as soon as the robot begins moving.

It is clear that motion generation can have great impact on the comfort of a collaborator in a shared workspace simply by altering a trajectories velocity to either extreme end of the spectrum. Further, altering execution velocities in human-human collaboration typically has minimal effect on the fluency of collaborative manipulation. Sure, in human-human collaboration there is minimal fear of personal danger when compared to human-robot collaboration; yet even when a robot moves extremely slowly in human-robot collaboration there is a lack of fluidity and noticeable inhibition when compared to that of collaboration between humans. There appears to be some other latent, nonverbal aspect of human motion generation missing from robot motion generation.

In this work, we attempt to leverage observations of human-human collaboration in a robot's motion planner in order to facilitate more safe and efficient collaboration. In the first work introduced, we present cost functions that attempt to understand the collaborative nature of a shared workspace over time. More clearly, we attempt to bias robot motions away from areas of the workspace that appear to be reserved for human traversal as well as similarly bias robot motion to being more consistent. Ultimately by considering avoidance and consistency in motion generation, the hope is that robot motion strategies that are better suited to their human collaborator emerge over time. We support this by showing in a human-robot workspace sharing study that our method improves task success rates. Next, we shift focus from long term collaboration strategies to considering strategies of motion generation that improve individual, discrete trajectories for a human collaborator. By considering human attention of the robot collaborator in motion generation we aim to better convey the robots motion intent. The hope being that with better human understanding of the robot's motion intent, the human can take advantage of areas of the workspace it knows with confidence the robot will not be entering.

# Chapter 2

# Considering Avoidance and Consistency in Robot Motion Generation

## 2.1 Introduction

While factory automation has been studied for many years, many manufacturing tasks have proven difficult to automate fully because they must be performed in close proximity to a human, or because parts of the task require human-level perception and/or manipulation capabilities not yet achievable by robots. To overcome this difficulty, a robot and human can collaborate to perform manufacturing tasks. However, when humans and robots share a workspace, the robot must be able to avoid interference with the human and potential collisions so that the task can be completed safely and efficiently.

A robot motion planning algorithm that is used in such a collaborative workspace must consider human safety and the human's expectations in the trajectories it produces. This chapter investigates how to formulate a cost function for the robot's

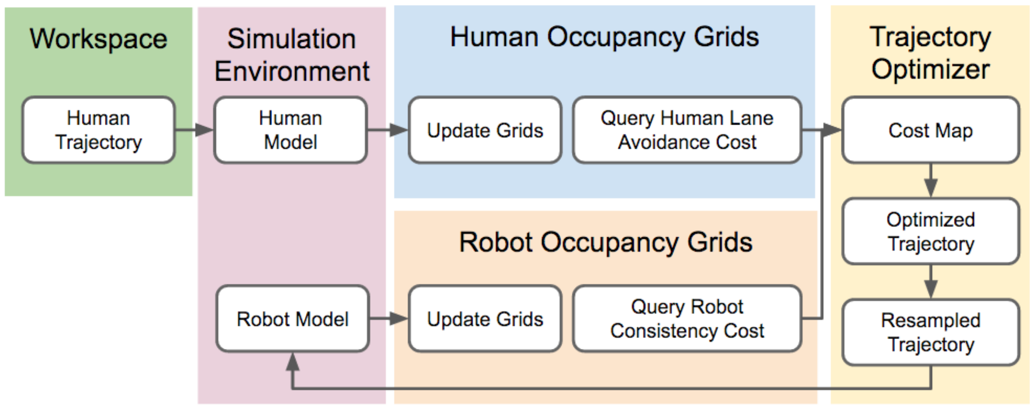


Figure 2.1: Pipeline for the lane framework. A human trajectory in the workspace is captured by a motion capture system. This data is sent to the simulation environment where an occupancy grid is updated. Finally, occupancy grids for both the human and robot are queried to generate two cost functions to be used in trajectory optimization. Planned trajectories are fed back into the occupancy grid to further describe robot lanes of traversal.

motion planner such that both safe and efficient task execution results. Two fundamental factors are considered in the formulation of the cost function: (1) Avoidance of the workspace previously-occupied by the human, and (2) Consistency of the robot's motion.

The first factor is motivated by our observations of humans collaborating on structured manipulation tasks where they are asked to repeatedly reach to the same areas in a shared workspace. After an initial period where the participants may try different strategies, their motion becomes very similar and they consistently enter the workspace of their partner with high speed while avoiding collision. We hypothesize that this behavior is akin to forming separate "lanes" that the humans move in. For instance one person may reach closer to the table (e.g. elbow down) while the other reaches over the first's arm (elbow up). In this chapter we investigate if this apparent lane-forming can be exploited for human-robot workspace sharing tasks by biasing the robot's motion planner to avoid the lanes created by the human.

However, considering the first factor alone may not be sufficient because the human's motion is affected by their partner's motion. This necessitates considering the second factor: consistency. Again, considering this factor is motivated from observations of previous human-robot collaboration experiments: When the robot's behavior (or even the robot itself) is new to the human, the human tends to move very cautiously. We hypothesize that this is because the human finds the robot's motion difficult to predict. However, as the two continue to move in the same space for a consistent set of tasks, the human usually becomes bolder in their motion around the robot, increasing their velocity and exploiting free work space around the robot's motion. Thus we also consider the consistency of the robot's motion as part of the cost function. To produce more consistent motion, we bias the planner to move the robot through the same workspace it used previously. A central challenge in this work is representing both the avoidance and consistency factors using cost functions in a way that works well in the TrajOpt [1] trajectory optimizer we use for motion planning. We propose a human lane penetration cost function to enable the avoidance of the human and a robot self-lane cost function to bias the robot toward consistent motion.

Our experiments in simulation and a human-robot workspace sharing study compare a cost function that avoids only the human lanes and a combined cost that uses both factors vs. a baseline method that is perfectly consistent but does not account for the human's previous motion. An overview of our experimental framework is shown in Figure 2.1. We find that using either cost function we outperform the baseline method in terms of success rate of the human's task without degrading task completion time. The cost function that accounts for both avoiding the human's lane and the consistency of the robot's motion has the best success rate. These results suggest that our method produces collaborations that interfere less with the human, allowing humans to move with less error and therefore making the task execution more efficient.

The remainder of this chapter describes related work, the implementation of the cost functions and their use in the TrajOpt motion planner [1], and the results of our experiments, followed by a brief discussion.

## 2.2 Related Work

Because human-robot collaboration is necessary for advanced automation, planning for a robot while maintaining human safety is an active research topic. Most safety efforts in this area appear to fall into two categories based on whether the robot is assumed to be functioning in an active or passive role when interacting with a human.

In the case where a robot is assumed to be taking an active role, the human is generally considered to take the role of a more passive observer. Planned motions are thus made to be more understandable to the human. For example, [2] aims to generate more human-like motions by first generating a reachability map that is used to choose an ergonomic and understandable goal configuration. Unlike this work, our goal focuses on generating robot trajectories that can avoid human motions in the shared workspace environment. Similarly, [3] takes into account a human's visibility, comfort and reachability in the robot's configuration space. This approach is useful when performing hand-over tasks to a human, however, we focus on applications in which a human and robot perform simultaneous pick-and-place tasks in a shared workspace. Finally, [4] aims to convey manipulation intent by purposefully bending a trajectory to better communicate which goal is being reached for. We focus more on generating robot motions that are consistent and avoid human workspace lanes rather than focusing on conveying human intent through motion generation.

In pick-and-place manufacturing tasks, a human should feel safe enough to work freely in a workspace; giving minimal consideration to its robot counterpart. In this scenario, to maintain safety the robot must take an observational role of the "active" human. As a result, research in this case generally aims to predict a human's future motion. Our previous work [5] uses a library of human motions performed in isolation to create pre-computed Gaussian Mixture Models that are queried on-line to predict a human's future workspace occupancy. [6] uses Inverse Optimal Control to predict a human's motion in human-human collaborations. However, these methods gather human motions either in isolation or in human-human collaboration. In this work the costs functions we create are derived from the human's motion in collaboration with the robot. Koppula et al. [7] used Conditional Random Fields (CRFs) to model affordances of objects and 3D trajectories of the human hand. This work has been recently extended in [8] to predict high-dimensional trajectories. Here we avoid parametric modeling of the human's motion, using the workspace occupancy directly. We also consider the consistency of the robot's motion, which is not addressed in the above methods.

While the principles of human motion have been investigated in [9], [10], [11], [12], these works take a more low-level approach by considering muscle activation or neural activity with respect to motion. However, our approach is more high-level in that we learn regions of the workspace to use or avoid in relation to a human. Our approach thus does not require a complex bio-mechanical model of the human.

Other works such as [13] [14] attempt to account for dynamic obstacles in motion planning. While the human may be considered a dynamic obstacle, we would like to avoid re-planning as much as possible due to its time cost. Our method aims to generate motions that could be used with with a dynamic planner, but we aim to minimize the need for re-planning under the assumption that areas with low human lane penetration cost are likely to be collision-free.

## 2.3 Approach

Our approach to generating consistent, human-avoiding reaching motions consists of a voxel based representation of the workspace that records occupancy over time for

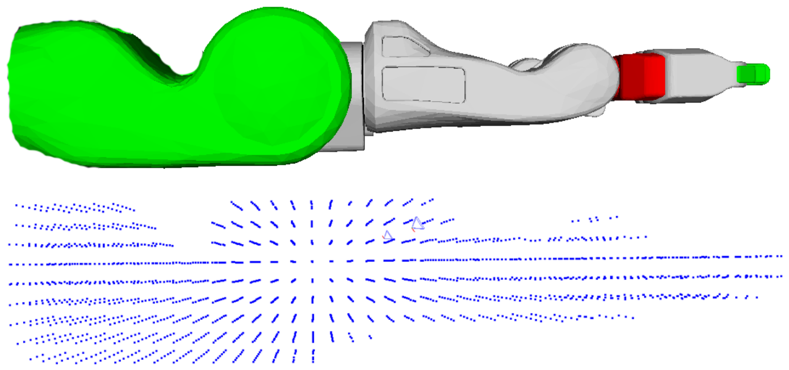


Figure 2.2: Example of samples taken on the PR2's manipulator

both a robot and human collaborator. These occupancy grids are used to define two cost functions. The first of these cost functions, the human lane penetration cost, aims to model the avoidance factor by repelling robot configurations from the areas of the workspace that should be reserved for traversal by a human collaborator. The second, the robot self-lane tracking cost, aims to model consistency factor by attracting robot configurations toward previously used areas of the workspace. These cost functions are then used in a trajectory optimizer.

### 2.3.1 Workspace Modeling

Our method discretizes the shared workspace into two voxel occupancy grids - $H[i, j, k]$ for the human and $R[i, j, k]$ for the robot, where $(i, j, k)$ is the index of the voxel in the occupancy grids. We define a function $m(\mathbf{p}) = (i, j, k)$ which maps a point $\mathbf{p} = (x, y, z)$ to an index of the voxel in the occupancy grids. Each voxel $H[i, j, k]$ and $R[i, j, k]$ is initialized with an occupancy value of 0.

To update the human and robot occupancy, we first pre-compute a set of samples in the geometry of the each link in the arm of both the human and robot simulation models, as shown in Fig. 2.2. To model human motions in simulation we use a motion capture system with inverse kinematics to obtain a 23 DoF configuration of the human's head, torso and right arm. Each pre-computed sample, denoted as

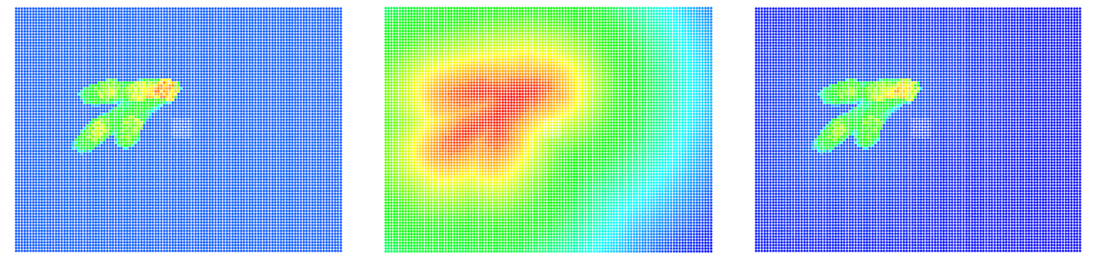

(a) Normalized Occupancy Value (b) Normalized SDF Value (c) Occupancy Value * SDF Value

Figure 2.3: A slice of the human lane penetration cost function from simulated data. From left to right, the first figure shows $occH(H, \mathbf{p})$, the second figure shows $sdfH(H, \mathbf{p})$, while the third figure shows the resulting product of both values, $pen\_cost(H, \mathbf{p})$

$\mathbf{p}_i^j$, represents the $i$th point on the robot/human model link $j$. At each time step, the transformation matrix $T_j^0$ between $F_j$, the frame of link $j$ and $F_0$, the frame of workspace is computed. We then transform $\mathbf{p}_i^j$ to the workspace frame by $T_j^0\mathbf{p}_i^j$. Thus the human and robot occupancy grids are updated as follows:

$$\begin{cases} H[m(T_j^0\mathbf{p}_i^j)] = H[m(T_j^0\mathbf{p}_i^j)] + 1 & \text{if link } j \in \text{Human} \\ R[m(T_j^0\mathbf{p}_i^j)] = R[m(T_j^0\mathbf{p}_i^j)] + 1 & \text{if link } j \in \text{Robot} \end{cases}$$

This sampling and incrementing of voxels allows areas of the workspace that are used most by their respective grid owners(human/robot) to emerge over time.

### 2.3.2 Human Lane Penetration Cost

In order to generate motions that avoid areas of the workspace used by a human, we define a human lane penetration cost function. The idea comes from the observation and assumption that human motions are always constrained in a subspace of the workspace when the human is working on a set of repetitive tasks in a shared workspace. This constrained subspace is similar to flight lanes used by aircraft. Each aircraft is constrained to a specified lane such that they can avoid each other without considering each other's motion. We likewise model the lanes used by a human through the above occupancy grids. Rather than model human lanes as solid obstacles that the robot needs to avoid completely, we model human lanes as a cost map called lane penetration cost, as the human lanes are not always entirely occupied by the human and the human can also adapt their motions to the robot.

Simply using the raw human occupancy value will assign high cost to configurations that penetrate high use regions of the workspace. However, this approach fails to provide lower cost to configurations far from occupied regions relative to configurations on the boundary of occupied regions. As a result, we employ a combination of the occupancy value, and the Signed Distance Field (SDF) of the human occupancy grid as follows:

$$pen\_cost(H, \mathbf{p}) = occH(H, \mathbf{p}) sdfH(H, \mathbf{p}) \tag{2.1}$$

where $\mathbf{p}$ is any point in the workspace, $occH(H, \mathbf{p})$ produces the normalized occupancy grids value for point $\mathbf{p}$, and $sdfH(H, \mathbf{p})$ is the normalized value of the SDF of the human occupancy grid. The $occH(H, \mathbf{p})$ is defined as:

$$occH(H, \mathbf{p}) \begin{cases} \frac{\log(0.9+1)}{\log(maxH+1)} & \text{if } H[m(\mathbf{p})] = 0 \\ \frac{\log(H[m(\mathbf{p})]+1)}{\log(maxH+1)} & \text{if } H[m(\mathbf{p})] > 0 \end{cases} \tag{2.2}$$

where $maxH$ is the maximum voxel value across the entire human occupancy grid. We use 0.9 when the voxel value is 0 in order to avoid flat regions of cost for the space outside human lanes, as a flat region has no gradient, which causes problems for the trajectory optimizer. We use the log function to reduce the influence of the maximum voxel value, as linear normalization of raw voxel values is overly sensitive to the maximum voxel value. Consider an example in which a human is standing at rest for numerous timesteps followed by a quick reaching motion. The voxels which belong to the reaching portion of the trajectory will be normalized to nearly zero cost relative to the maximum cell value recorded for the resting portion of the trajectory.

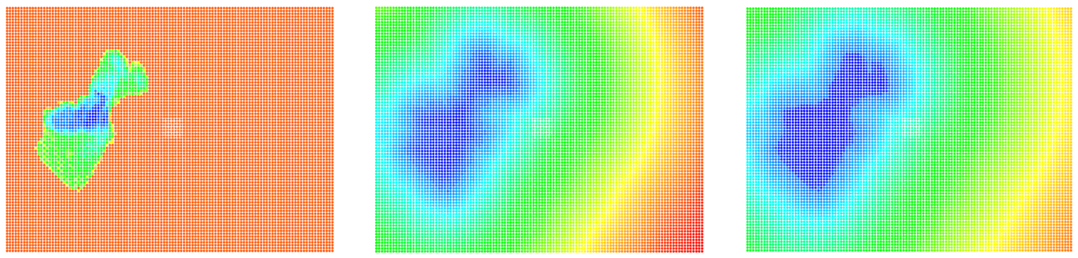

(a) Normalized Occupancy Value (b) Normalized SDF Value (c) Occupancy Value * SDF Value

Figure 2.4: A slice of the robot self lane cost function from simulated data. From left to right, the first figure shows $occR(R, \mathbf{p})$, the second figure shows $sdfR(R, \mathbf{p})$, while the third figure shows the resulting product of both values, $self\_cost(R, \mathbf{p})$.

Using a log here ameliorates this issue. The $sdfH(H, \mathbf{p})$ is defined as:

$$sdfH(H, \mathbf{p}) = \frac{\arctan(maxSH) - \arctan(sdf(H, \mathbf{p}))}{\arctan(maxSH) - \arctan(minSH)} \tag{2.3}$$

where $sdf(H, \mathbf{p})$ is the function that returns the SDF value for a given occupancy grid and a given query point $\mathbf{p}$. $maxSH$ and $minSH$ are the maximum and minimum SDF value for the human occupancy grids $H$. $sdf(H, \mathbf{p})$ will return a negative value when $\mathbf{p}$ is inside the occupied volume described in the occupancy grids $H$, a positive value if $\mathbf{p}$ is outside or 0 if $\mathbf{p}$ is on the boundary. The magnitude is determined by how far $\mathbf{p}$ is from the boundary of the volume, which, in our case, is the outside surface of the lanes. We use the arctan function to give an upper bound and a lower bound to the SDF value and keep it equal to 0 on the boundry, as we want to reduce the influence of the points far away from the volume. Note that this normalization function results in the most negative value of the SDF (the innermost point in the lane) mapping to 1, and the most positive value of the SDF (the point farthest from the lane) mapping to 0. A two-dimensional slice of the human lane penetration costmap can be seen in Figure 2.3.

### 2.3.3 Robot Self-Lane Cost

While the lane penetration cost aims to facilitate human workspace avoidance, we also need to consider the consistency of the robot's motion. We define a robot self-lane cost in a similar fashion to the lane penetration cost with the intention of favoring motions which traverse the voxels most used in the robot's occupancy grid. The robot self-lane cost is defined as follows:

$$self\_cost(R, \mathbf{p}) = occR(R, \mathbf{p}) sdfR(R, \mathbf{p}) \tag{2.4}$$

where $occR(R, \mathbf{p})$ and $sdfR(R, \mathbf{p})$ represent the normalized voxel value of robot occupancy grids and normalized SDF value of robot occupancy grids similar to the human lane penetration cost. However, we invert the upper and lower bound of the normalization for each term as the robot self-lane cost is intended to attract the robot while the human penetration cost aims to repel the robot. The $occR(R, \mathbf{p})$ and $sdfR(R, \mathbf{p})$ are defined as follows:

$$occR(R, \mathbf{p}) \begin{cases} 1 - \frac{\log(0.9+1)}{\log(maxR+1)} & \text{if } R[m(\mathbf{p})] = 0 \\ 1 - \frac{\log(R[m(\mathbf{p})]+1)}{\log(maxR+1)} & \text{if } R[m(\mathbf{p})] > 0 \end{cases} \tag{2.5}$$

$$sdfR(R, \mathbf{p}) = \frac{\arctan(sdf(R, \mathbf{p})) - \arctan(minSR)}{\arctan(maxSR) - \arctan(minSR)} \tag{2.6}$$

where $maxR$ is the maximum voxel value of the robot occupancy grid $R$, and $maxSR$ and $minSR$ are the maximum and minimum SDF value of the robot occupancy grid $R$. The product of these normalized terms produces 0 cost in voxels most occupied in the center of occupied regions, with a smooth increase in cost of unoccupied regions. A two-dimensional slice of the robot self lane costmap can be seen in Figure 2.4.

Unlike the human lane penetration cost which accounts for all observed human motion, regardless of the task the human is performing (we do not assume the robot knows which task the human is doing), the robot self-lane cost function aims to model consistency for each task separately. Thus we maintain one robot occupancy grid for each robot task.

### 2.3.4 Robot Trajectory Optimization

We use the TrajOpt [1] sequential convex optimization algorithm to plan with the cost functions described in Sections 2.3.2 and 2.3.3. We include collision, final end-effector pose, and maximum end effector displacement constraints as well as a weighted joint velocity cost. After trying a number of different initial trajectories, we found the best initialization was a path consisting of the first $n-1$ configurations as the starting configuration and the lowest-cost inverse kinematics solution for the $n$th (final) configuration.

The final cost which TrajOpt uses is the weighted sum of all above costs. We manually tuned the weights between different costs using a pre-recorded training dataset. The tuning process aims to find a weighting that is optimal in terms of human lane penetration cost.

## 2.4 Results

In this section we present results illustrating the capability of both of the above cost functions to generate human lane-avoiding and consistent robot reaching motions. We first show our cost functions' ability to generate human lane-avoiding motions in simulation by comparing generated paths to a baseline. Next we test the impact of our cost functions in a human subjects study by comparing robot behavior produced by our cost functions to a baseline in terms of the task completion time and task success rate. We define the baseline for comparison as a straight-line path in the robot's configuration space from the robot's initial configuration to the inverse kinematics solution at the end-effector goal for a given task that is closest to the initial configuration. The baseline thus does not account for any previous observations of the human and would correspond to what the robot would do if it were performing its task without the human. Importantly, these baseline trajectories are optimally-consistent, as they never change for a given task.

### 2.4.1 Recording Method

In order to record human workspace occupancy for simulated and experimental data, we used a Vicon motion capture system. Human subjects wore a suit consisting of four rigid plates and six individual markers which were placed according to biomechanics industry standards [12]. We used as many rigid plates as possible as they provide more robust tracking than their marker counterparts. From our motion capture setup, we extract the center of rotation of the human's right wrist, elbow, shoulder and torso. We obtain a 23 DoF configuration of the human's right arm and torso by performing inverse kinematics with these joint centers.

### 2.4.2 Experiments in Simulation

To evaluate our capacity to optimize trajectories for the presented cost functions, we perform initial comparisons to the baseline method in a simulated environment. Because we are performing this analysis in simulation, we are unable to evaluate the effects of consistent robot motion on a human subject. Instead, in simulation we aim to show that our optimization can produce paths of lower human lane-penetration cost than the baseline method. This analysis is performed in two phases: data collection and planning-in-simulation. (1) In the data collection phase, we record data of a human performing reaching motions to several goals denoted by colored regions on a tabletop. We then manually-segment the recordings into individual reaching motions. (2) In the planning-in-simulation phase, we then place a simulated PR2 robot at the

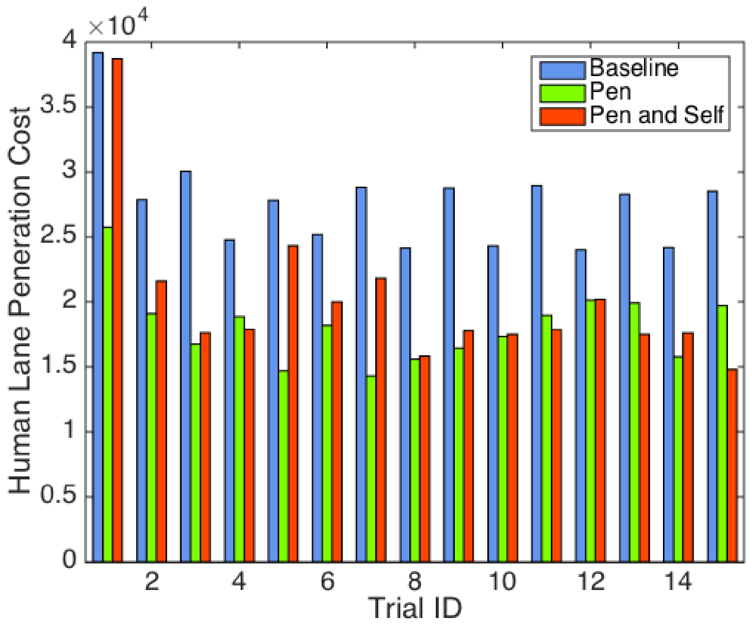


Figure 2.5: Comparison of human lane penetration costs for baseline, human lane avoidance (Pen), and human lane avoidance and robot self-lane consistency (Pen and Self) trajectories in simulation

side of the table opposite the simulated human. We then forward-simulate segmented human reaching motions until task completion, recording workspace occupancy of each configuration in the human's motion. We then plan trajectories for the baseline, a trajectory optimized according to Section 2.3.4 with the human lane penetration cost (Pen), and a trajectory optimized with both the human lane penetration cost and robot self-lane cost (Pen and Self).

The three methods produce trajectories of varying length. In order to produce a fair comparison between all three types of trajectories, we insert a trajectory resampling step prior to evaluation of a trajectory's penetration cost, so that trajectories are discretized at a fixed step size of 0.05rad . The resampling step is necessary because, while the number of waypoints used by TrajOpt is constant, the distance between them is not.

Figure 2.5 provides a comparison, in terms of human lane penetration cost, of the trajectories produced by the three methods on the example scenario shown in

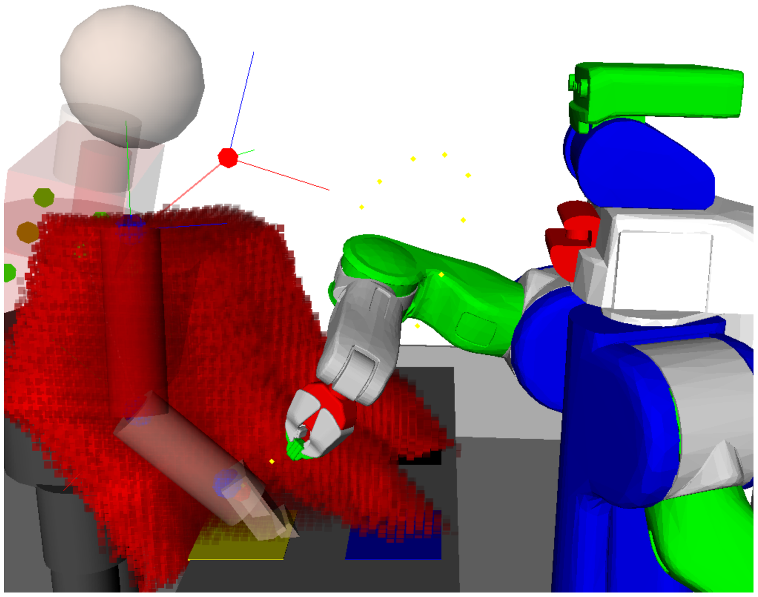


Figure 2.6: An example of the human lane penetration cost map. Unoccupied voxels not drawn.

Figure 2.6. The x axis on the figure shows the sequence of reaching targets for the robot. Note that the voxel grids are updated between the tasks with the human's and robot's previous motion. The results clearly show that optimization with the lane-avoidance cost function produces significant improvements for avoiding lanes over the baseline. Unsurprisingly, if we include the self-lane tracking term in the optimization, the optimizer settles to a solution that trades off self-lane following for avoiding the human's lane, thus we see that the lane-penetration cost of the combined cost function is worse than optimizing for lane-avoidance alone. In most cases, however, the combined cost function still improves over the baseline in terms of avoiding the human's lane.

### 2.4.3 Human-Robot Experiments

While the results performed in simulation demonstrate the capability of both methods to generate trajectories of lower human lane penetration cost than a baseline, it is unclear what effect such robot trajectories will have on a human collaborator. We

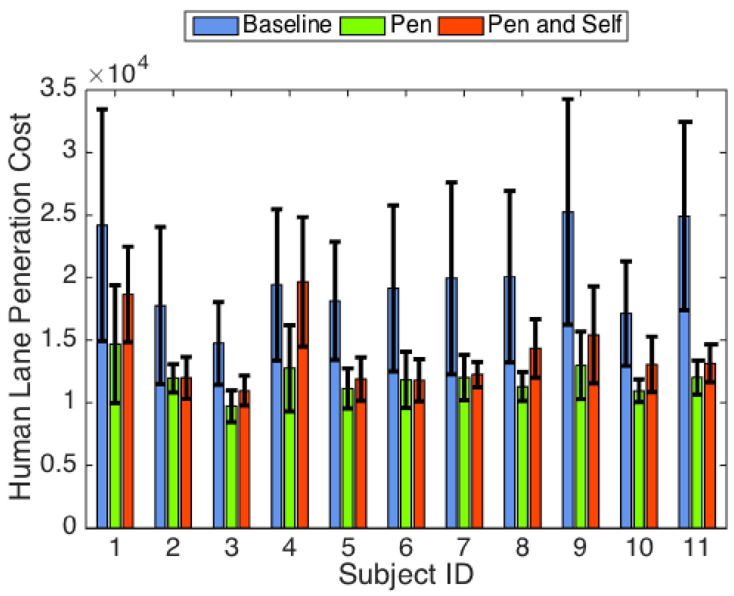


Figure 2.7: Comparison of human lane penetration costs for baseline, human lane avoidance (Pen), and human lane avoidance and robot self-lane consistency (Pen and Self) trajectories for each participant in the user study

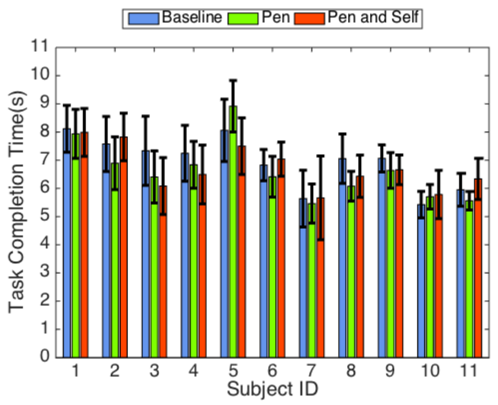
conducted a human subjects study where a human performed pick-and-place motions while a PR2 performed reaching motions in a shared workspace. We evaluated each method in terms of task completion time and task success rates. These metrics were chosen under the assumption that workspace lane avoiding motions from the robot will make human task execution simpler. To ensure fairness between methods we consider the maximum execution time between subjects and the PR2 as the task completion time.

The experiment was designed so that the robot would need to enter the human's workspace to complete its task and vice versa. The task was for the robot to reach to one of two specified end-effector poses near the human and then return to its initial configuration. Simultaneously, the human was to perform two pick-and-place motions to targets near the robot.

When a human moves at their natural speed, they can perform their task much faster than we can safely execute a robot trajectory, allowing the human to avoid the robot entirely. As a result, we directed participants to place two ping-pong balls balanced on top of 3D printed molds with a slight dimple at two goal positions in sequence without letting the balls roll off the molds. This constrained task required the human to move at a speed comparable to the robot. We define a successful completion of a task as a trial in which both of the molds are placed directly in the predefined goal regions and the ping-pong balls remain balanced on the molds for the duration of execution.

Upon entering the experiment area, subjects were read a script which briefly explained they were to perform a collaborative manipulation task. Next, the task to be performed was verbally explained while being visually demonstrated in unison. Participants were asked to move at a comfortable speed while ensuring balance of the ping pong ball throughout the task. Finally five demo executions were performed to ensure basic task understanding.

After being introduced to the study, one of the three methods was selected at random. The human performed a sequence of twenty executions of the aforementioned task which contained a balanced number of trivial executions in which the PR2 reached to an unrelated area of the workspace as well as conflicting executions in which the PR2 reached to the same area of the workspace as the human (eg. Figure 2.9). The experimenter initiated the task by telling the human which targets to place the molds on and simultaneously starting the robot's motion for its own target. A run of the experiment was finished when the human and robot both completed their tasks. A new robot trajectory is planned prior to each task execution in the sequence. On average, planning took 4.3ms, 6.82s, 15.39s with a standard deviation of 3.5ms, 2.77s, 5.89s for the Baseline, Pen and Pen+Self methods, respectively. Upon finishing the sequence subjects were asked to play a simple video game for five minutes before continuing to the next randomly selected method. This is designed to encourage subjects to shift their attention away from the robot and thus engage with each method with similar familiarity.



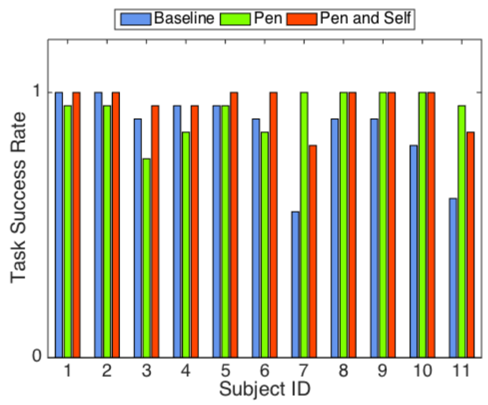


Figure 2.8: Comparison of task completion times (left) and task success rates (right) for baseline, human lane avoidance(Pen), and human lane avoidance and robot self-lane consistency(Pen and Self) trajectories for each participant in the user study

We ran this experiment with 11 subjects, each performing 20 runs of each method generating a total of 660 human reaching trajectories. We had 8 male participants and 3 female participants. The ages of the participants rage from 18 to 25. The median age is 24. Motions generated using TrajOpt were initialized with a trajectory of 12 waypoints as described in Section 2.4.2. The human lane penetration cost was scaled with $\alpha = 0.7$, the robot self-lane cost with $1 - \alpha$, the weighted joint velocity cost was $1000 \times w$ where $w$ is an auto-generated weight vector for the PR2 robot. In addition to this, a maximum displacement of 0.1 meters on end-effector displacement between configurations was used to encourage even spacing of configurations.

Figure 2.7 shows a comparison of average human lane penetration costs for all three methods for every subject in the experiments. Similar with the simulation results, optimization with the human lane penetration cost alone shows significant improvement in terms of path cost when compared to the baseline. Likewise, except for subject four, motions planned including the robot self-lane cost yield trajectories which outperform the baseline, yet are of slightly higher cost than when optimizing for human lane penetration cost alone.

The benefit of this improvement in lane penetration cost for both of our methods can be seen in Figure 2.9, in which individual robot configurations are visually compared. In the figure, the robot is executing a task which conflicts with the subject's task. In the case of a trajectory planned with the baseline, the PR2's arm significantly occludes the subject's goal region, necessitating a pause in the subject's motion. Alternatively, trajectories which include the human lane penetration term provide ample room for the subject to execute his or her task fluidly.

Table 2.8 shows the average task completion time, task success rate and human lane penetration cost for the entire dataset. Both methods slightly outperform the baseline in terms of task completion time. The method which aims to only avoid the human has the lowest execution time while the method with the added robot consistency term has a better task success rate. This is what one would expect as the pure avoidance method would sometimes generate motions that would confuse or alert subjects, causing them to lose balance of the ping-pong ball. Most surprisingly, the optimally consistent baseline method had the lowest success rate of all.

A more detailed break down can be seen in Figure 2.8 which shows average task completion times and task success rates for each human subject under each method. In every subject excluding subject ten, one of the two methods outperforms the baseline in terms of task completion time, though often by a small margin. Additionally, the Pen+Self method consistently outperforms the baseline and Pen method in terms of task success rates. While the improvements of our methods over the baseline may seem modest, it is important to note that these are elementary reaching tasks and that many such motions need to be performed in the course of a practical manufac-

| Method | Task completion Time | Task Success Rate | Cost |
|---|---|---|---|
| Baseline | $6.94 \pm 1.21$ s | $0.8591 \pm 0.1443$ | $2.00e4 \pm 7.33e3$ |
| Pen | $6.62 \pm 1.23$ s | $0.9318 \pm 0.0777$ | $1.19e4 \pm 2.57e3$ |
| Pen+Self | $6.71 \pm 1.17$ s | $0.9591 \pm 0.0668$ | $1.39e4 \pm 3.81e3$ |

Table 2.1: Comparison of average task success rate, path cost, and task completion time for each method over 11 subjects each performing 20 task executions

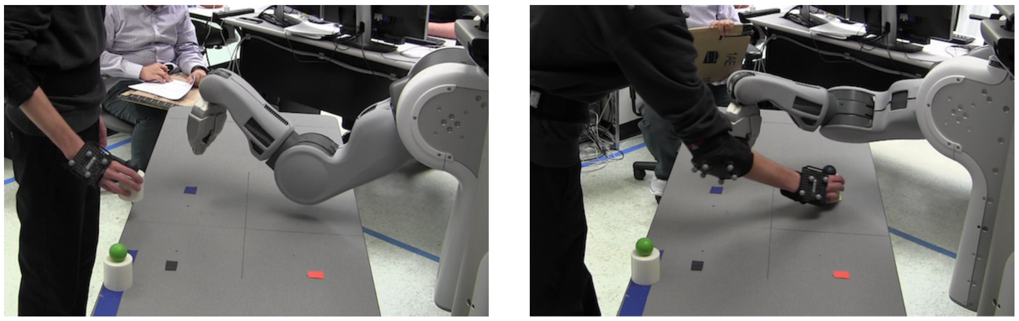


Figure 2.9: A comparison of robot configurations generated by the baseline method (left) and the avoidance-only method (right). The baseline method is occluding the human's goal region while the avoidance-only method leaves the goal region open.

turing task (e.g. assembly), so the time savings can add up to substantial amounts. Task errors, such as dropping the ball, are especially important to avoid as they would require the human to recover from the error and repeat the task, thus requiring more time.

## 2.5 Discussion

In this work we have presented two cost functions for generating safer motions within proximity to a human in a shared workspace. These costs aim to model avoidance of regions of the workspace that are of importance to human task manipulation, as well consistency in robot motion; factors which were identified as important to human manipulation through observation of human collaboration. To demonstrate the efficacy of the proposed cost functions we provide both simulated results as well as results from a human subjects study. We found that using either cost function we outperform the baseline method in terms of task success rate without degrading the task completion time and the cost function that aims to produce consistent robot motion while avoiding the human produces the highest success rate. These results suggest that the proposed cost functions do indeed improve the efficiency of human robot collaboration in shared workspaces. In future work we aim to study the effects of collaboration strategies on users with different collaboration preferences (e.g. leading vs. following).

# Chapter 3

# Generating Human-Attention Aware Robot Motion via Policies

## 3.1 Introduction

In the previous chapter we discussed two factors to consider in the cost function for the robot's motion planner. The first of these cost functions facilitated avoidance of the human collaborator's workspace for better safety in robot motion execution. The second cost function, the consistency term assists in making robot motions as understandable to the human as possible. However, consistent robot motion in this context can be seen as a long-term temporal measure of motion clarity. As the robot continues to execute trajectories, motions will continuously become more and more similar over time. In this chapter, we shift our focus from trajectories over continuous executions, to instead making individual trajectories more clear to a human collaborator.

More specifically, in this chapter we consider the problem of generating legible robot motions, or rather, motions that communicate the robot's intent to a human collaborator. Or rather, motions which enable an observer to correctly infer a robot collaborator's goal after observing only a portion of the full trajectory. The more quickly this inference occurs, the more legibile the trajectory. Such motions are crucial to producing more efficient execution of collaborative manipulation tasks in shared workspaces. By proactively expressing intent through motion, a collaborator can infer future workspace occupancy and current task goals more readily than in naive motion generation. As a result a collaborator can accordingly plan their own motion without observing a full trajectory.

However, efficiency gains through legible motion are only relevant when the robot is capturing the human collaborator's attention. Without an observer present, robot motion should be as goal directed and efficient as possible. Thus, this work focuses on developing a method of capturing human attention while proactively responding to this attention with varying levels of legible robot motion.

Legibility of robot motions is well studied in the field of trajectory optimization. Unfortunately, trajectory optimization does not well lend itself to the problem space of attention-aware legible motion generation. As human attention, or focus is variable on-line legibility of robot motion must be similarly variable. A central challenge in this work is adapting legibility from current literature to be used in a motion policy that can be queried in near real time.

## 3.2 Related Work

Human eye-tracking has long been studied in computer vision [15] as an image processing problem as well as in the human-computer interaction community as an input device [16]. In a similar fashion eye-tracking has been leveraged in robotics as a means of teleoperation [17, 18] as well as a heuristic for human intent [19, 20] in human-robot interaction. However, as far as we know, eye-tracking has not been paired with motion generation as we do in this work.

While legible motions have been well studied in the field of robot motion planning, there do not appear to be any papers that pair legible motion with varying human attention. For example, Dragan et al. have a series of papers discussing legibility and its generation. [21] discusses a dichotomy between predictability and legibility. More clearly, a naive robot, in an attempt to be legible could deform its path so much that the robots intention is no longer predictable. In [22] they address this dichotomy by introducing a boundary constraint on path deformation. They validate the use of a legibility boundary condition with the inclusion of initial user studies. While this work differs from our method in that the trajectories executed in these user studies are static, we do leverage the intuition behind formulation of a legibility cost in this work. More recently, in [23] legibility research has been extended to include an observer's viewpoint in motion generation. In prior work motions were deemed legible to an omniscient observer, where in reality the observer's viewpoint is what dictates legibility. To solve this problem, prior to trajectory optimization they apply a transformation such that trajectories are projected into the viewplane of an observer. While considering the view of an observer appears closest in nature to this work, it is important to note that their viewpoint is arbitrarily constructed and considered static for the full length of the trajectory.

## 3.3 Approach

Our approach to generating legible, human-attention aware robot motions consists of three parts. First, we start by describing how we capture the human collaborator's attention with the use of an eye tracker. Next we describe the process of generating a graph that will define a motion policy that can be queried on-line. Finally, we present our version of a legibility cost function that can be used with the policy graph by adapting edge weights. A general outline of this pipeline can be seen in Figure 3.1.

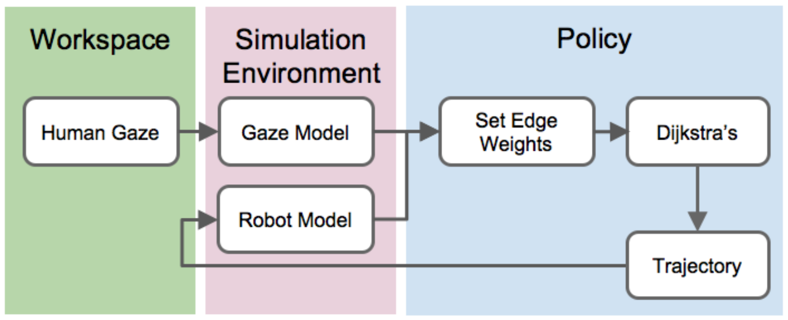


Figure 3.1: Pipeline for attention aware legible motion generation. The human collaborator's attention is captured by an eye tracker and motion capture system. This data is sent to the simulated environment where a gaze model is collision checked with the robot model's manipulator. The gaze collision, or lack thereof then defines the edge weights used in the policy graph. Dijkstra's algorithm is computed on this graph to find shortest paths throughout the workspace. Finally, robot trajectory execution is fed back into the simulation environment's robot model to be used with future gaze collision checks.

### 3.3.1 Capturing Human Attention

In order to capture human attention, we employ the use of two sensors. The first, a head mounted eye tracker and the second, a motion capture system. The eye tracker was built using a DIY guide from pupil-labs [24] which involves de-casing two low cost consumer webcams and minor soldering. An example of the physical eye-tracker and its output can be seen in Figure 3.2. Traditionally, researchers involved in eye tracking focus on fixation points. To do this they record pupil positions with an inward facing camera and project these pupil positions into a 2d plane defined by an outward facing camera. Recently, pupil-labs developed a 3d method which models the pupil as a disc on an anthropomorphically defined sphere, allowing the definition of a gaze vector. The availability of a gaze vector, and not a gaze fixation point is essential to this work. However, the gaze vector must be localized in the collaboration workspace. For this we developed a calibration procedure using a VICON motion capture system.

The documentation provided with the DIY guide does not describe what coordinate system the gaze vector is defined in. Additionally, as the eye-tracker we chose is head mounted it is imperative to employ some other technique to track the humans

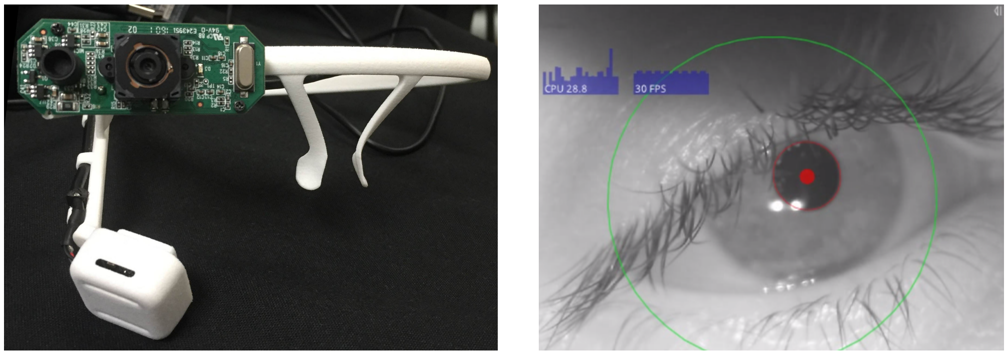


Figure 3.2: The completed eye-tracker (left) and output from the provided 3d tracking software (right).

head movements. As a result we involve the use of an additional head-mounted mocap plate to record ground truth data in addition to gaze vectors recorded from the eye-tracker. This ground truth data is then used to recover a transform between the gaze frame and the mocap world frame. The process of recovering an unknown affine-transform between two sets of points, named the problem of point set registration, is well studied in the field of computer graphics.

In order to collect calibration samples we start by finding the translation from the subject's mocap head plate to their eye. We do this by simply having the subject hold a single mocap marker over their closed eye, being careful to subtract the height of the marker from the recorded position for better accuracy. Next, we have the subject look at a known mocap calibration object a number of times recording the reported gaze vector and positions of the calibration object and head plate. This process defines $n$ pairs of gaze vectors and ground truth mocap vectors. The problem of point set registration for affine-transforms is best defined for perfectly aligned data in different frames. Unfortunately, due to ocular drift and microsaccades of the human eye we cannot assume that our recorded gaze vectors can be perfectly aligned with our recorded mocap vectors. As a result we first recover an initial guess transform, under the assumption that the data is perfectly aligned, with [25]. We use this as the initial guess in a bundle adjustment optimization using [26].

### 3.3.2 Policy Generation

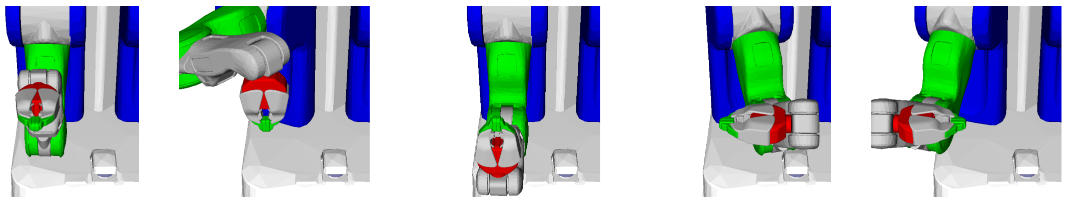


Figure 3.3: Example inverse kinematics solutions using 5 uniformly distributed quaternions. A 6th orientation which points the robot's end-effector backwards is removed from the set as it is inherently non goal-directed and thus less legible.

Before we can generate legible motions that respond to human attention defined in Section 3.3.1 we must first define a graph that adequately covers the robot's workspace. One initial approach may be to simply generate a Probablistic Roadmap, however, PRMs have no guarantees on path quality. Instead, we uniformly sample configurations from a reachability map of the robot's workspace. To generate this reachability map we first initialize a voxel grid centered at the robots base. For each cell in this voxel grid we uniformly sample a set of quaternions. An example of the uniform quaternion sampling used in this work can be seen in Figure 3.3. We then compute and store inverse kinematics solutions for each quaternion in the voxel. The successful IK solutions stored in this voxel grid will represent the nodes in our policy graph.

In order generate edges for the policy graph we then employ two procedures: one to ensure connectedness of the graph and the other to refine path quality. Naively connecting every node in the graph to one-another quickly becomes intractable. Additionally, simply connecting based on k-nearest neighbor does not ensure connections between neighboring voxels, often resulting in a disconnected graph. In the first procedure we iterate over all voxels. First, configurations that share a voxel are all connected to one-another. Next configurations of minimal distance in neighboring voxels are computed and connected. This procedure ensures connectivity between voxels but does not produce paths of good quality. We finally call an additional k-nearest neighbor connection on all nodes in the graph to refine path quality. The full pseudo code for policy generation can be seen in Algorithm 1.

**Algorithm 1** Policy Generation

```
procedure GeneratePolicyGraph
    $v \leftarrow$ GenerateReachabilityMap
    $G \leftarrow \{v, \emptyset\}$
    $G \leftarrow$ EnsureConnectedness($G$)
    $G \leftarrow$ RefinePathQuality($G$)
    return $G$
procedure GenerateReachabilityMap
    $v \leftarrow \{\emptyset\}$
    for all $x, y, z \in RobotWorkspace$ do
        for all $quat \in$ GetNQuaternions do
            $sols \leftarrow$ ComputeIKSolutions($x, y, z, quat$)
            $v \leftarrow v \cup sols$
    return $v$
procedure EnsureConnectedness($G$)
    for all $v_i \in G$ do
        for all $v_j \in$ Neighbors($v_i$) do
            $q_i, q_j \leftarrow argmin_{q_1 \in v_i, q_2 \in v_j}$ dist($q_1, q_2$)
            for all $q \in v_i$ do
                $G$.AddEdge($q, q_i$)
                $G$.AddEdge($q_i, q$)
            $G$.AddEdge($q_i, q_j$)
    return $G$
procedure RefinePathQuality($G$)
    for all $v \in G$ do
        for all $q_i \in v$ do
            $nearest \leftarrow$ GetKNearestNeighbors($q_i$)
            for all $q_j \in nearest$ do
                $G$.AddEdge($q_i, q_j$)
                $G$.AddEdge($q_j, q_i$)
    return $G$
```

## 3.3.3 Policy Querying

Now that generation of the policy graph is defined, we must define a legibility cost function that can be computed on this policy graph. While it is true that generation of legible motion is a well studied topic in motion planning (Sec. 3.2), these approaches are from the trajectory optimization field and operate in trajectory space, not on individual configurations. Thankfully, in these works they describe the intuition behind their method as simultaneously minimizing effort (in their case sum squared velocities) while maximizing distance to other goals. We build on this intuition to define a cost function that is similar in spirit, yet can be computed on graph edges.

Given a single active goal, $G_A$ and a set of $n$ passive goals $G_P$, we would like to select paths which minimize distance to $G_A$ and maximize distance to $G_P$. Because the nodes of our policy graph are configurations while our goal positions exist in workspace, we need an approximation of configuration distance to each goal. We employ the use of link samples like in Figure 2.2 for this approximation. The distance of a given configuration $q$ to a given goal $G$ is defined as the summed distances of all samples on the manipulator where samples from a given link are averaged such that each link has equal weighting. This metric can additionally be extended to edges as the change in distance to a given goal.

$$Gdist(G, q) = \sum\nolimits_{l \in links(q)} \sum\nolimits_{sample \in l} \frac{dist(sample, G)}{|l|} \tag{3.1}$$

$$Change_A(q1, q2) = Gdist(G_A, q2) - Gdist(G_A, q1) \tag{3.2}$$

$$Change_P(q1, q2) = \sum\nolimits_{G \in G_P} \frac{Gdist(G, q2) - Gdist(G, q1)}{|G|} \tag{3.3}$$

$Change_A$ and $Change_P$ define the change in distance towards the active and passive goals for a given edge. As mentioned earlier, we would like to minimize distance to the active goal and maximize distance from all passive goals. The final edge cost is defined as follows:

$$EdgeLegibility(q1, q2) = dist(q1, q2) + exp(Change_A(q1, q2)) + exp(-Change_P(q1, q2)) \tag{3.4}$$

Finally, for each goal $G$ that may be an active goal during collaborative manipulation, we pre-compute $EdgeLegibility$ and compute a Dijkstra's solution to be used online.

## 3.4 Results

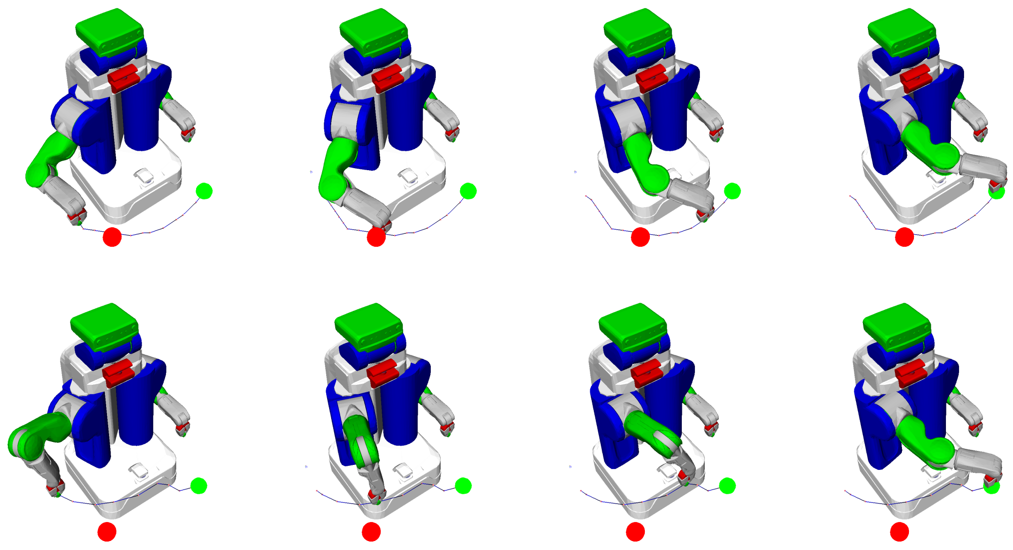


Figure 3.4: Comparison of a baseline trajectory recovered from our policy (top) to a legible path recovered from our policy (bottom). The red circle represents the passive goal position while the green circle represents the active goal position.

In this section we present initial results of our cost functions' ability to generate legible motions in simulation, as well as describe an initial implementation of a human-robot workspace sharing experiment. Figure 3.4 shows a comparison of paths recovered using only configuration space distance as edge weights to those recovered using *EdgeLegibility* edge weights. The figure shows that our cost function does indeed maximize distance from passive goals while minimizing distance to the active goal, however it is unclear if this leads to more legible motions. For example, the robot's shoulder is raised abnormally high, deviating a large amount from the robot's active goal configuration. It's possible that even with motions of this style, a human collaborator would be unable to properly infer the robots goal configuration until very late in trajectory execution.

In addition to results in simulation, we present an initial implementation of a motion planner which generates paths of varying legibility in response to human attention (Figure 3.5). In this implementation the robot begins executing a baseline trajectory to its intended goal. The human's gaze is collision checked with the robot's manipulator in real time. If the human is focused on the robot's execution it will query our policy graph with *EdgeLegibility* weights.

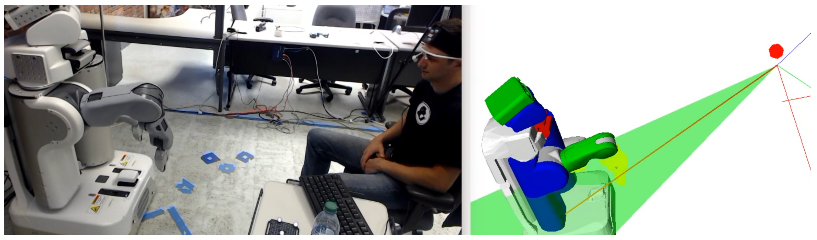


Figure 3.5: Example of our framework generating legible motions in response to human attention

## 3.5 Conclusions & Future Work

In this chapter we have presented an initial approach toward generating robot motions with the intent of more efficient human-robot collaboration. This method attempts to generate legible motions within the context of human attention. We have shown an initial motion policy that minimizes distance to passive goals while maximizing distance to active goals as well as a trajectory controller which queries this policy based on human gaze. However, it is unclear if the cost function presented is indeed more legible. In future work we would like to run a user study in which subjects are asked to predict a robot goal configuration from a set of potential goal locations without human attention in the loop. If we can indeed produce motions which better communicate intent when compared to a baseline we would additionally like to run a user study which does include human attention in the loop. This user study would verify the assumption that generating more expressive motions will have an impact on task efficiency.

# Bibliography

[1] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *IJRR*, vol. 33, pp. 1251–1270, Aug. 2014.

[2] F. Zacharias, C. Schlette, F. Schmidt, C. Borst, J. Rossmann, and G. Hirzinger, "Making planned paths look more human-like in humanoid robot manipulation planning," in *ICRA*, May 2011.

[3] J. Mainprice, E. Sisbot, L. Jaillet, J. Cortes, R. Alami, and T. Simeon, "Planning human-aware motions using a sampling-based costmap planner," in *ICRA*, May 2011.

[4] A. Dragan and S. Srinivasa, "Generating legible motion," in *Robotics: Science and Systems*, June 2013.

[5] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion.," in *IROS*, Nov 2013.

[6] J. Mainprice, R. Hayne, and D. Berenson, "Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning," in *ICRA*, May 2015.

[7] H. S. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," in *Robotics: Science and Systems*, RSS, 2013.

[8] Y. Jiang and A. Saxena, "Modeling high-dimensional humans for activity anticipation using gaussian process latent crfs," in *Robotics: Science and Systems, RSS*, 2014.

[9] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[10] E. Burdet and T. E. Milner, "Quantization of human motions and learning of accurate movements," *Biological cybernetics*, vol. 78, no. 4, pp. 307–318, 1998.

[11] E. Demircan, T. Besier, S. Menon, and O. Khatib, "Human motion reconstruction and synthesis of human skills," in *Advances in Robot Kinematics: Motion in Man and Machine*, pp. 283–292, Springer, 2010.

[12] Wu and et al, "Isb recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion – part ii: shoulder, elbow, wrist and hand," *Journal of biomechanics*, vol. 38, no. 5, pp. 981–992, 2005.

[13] C. Park, J. Pan, and D. Manocha, "Real-time optimization-based planning in dynamic environments using gpus," in *ICRA*, 2013.

[14] O. Brock and O. Khatib, "Elastic strips: A framework for motion generation in human environments," *The International Journal of Robotics Research*, vol. 21, no. 12, pp. 1031–1052, 2002.

[15] D. D. Salvucci and J. H. Goldberg, "Identifying fixations and saccades in eye-tracking protocols," in *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*, ETRA '00, pp. 71–78, ACM, 2000.

[16] L. A. Granka, T. Joachims, and G. Gay, "Eye-tracking analysis of user behavior in www search," SIGIR '04, pp. 478–479, ACM, 2004.

[17] D. Zhu, T. Gedeon, and K. Taylor, "moving to the centre: A gaze-driven remote camera control for teleoperation," *Interacting with Computers*, vol. 23, no. 1, pp. 85–95, 2011.

[18] D. P. Noonan, G. P. Mylonas, A. Darzi, and G. Z. Yang, "Gaze contingent articulated robot control for robot assisted minimally invasive surgery," in *IROS*, pp. 1186–1191, Sept 2008.

[19] M. Staudte and M. W. Crocker, "Investigating joint attention mechanisms through spoken humanrobot interaction," *Cognition*, vol. 120, no. 2, pp. 268 – 291, 2011.

[20] D. Kuli and E. A. Croft, "Estimating intent for human-robot interaction," in *ICRA*, 2003.

[21] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *HRI*, pp. 301–308, 2013.

[22] A. Dragan and S. Srinivasa, "Generating legible motion," in *RSS*, June 2013.

[23] S. Nikolaidis, A. Dragan, and S. Srinivasa, "Viewpoint-based legibility optimization," in *HRI*, March 2016.

[24] M. Kassner, W. Patera, and A. Bulling, "Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction," April 2014.

[25] B. K. P. Horn, “Closed-form solution of absolute orientation using unit quaternions,” *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[26] S. Agarwal, K. Mierle, and Others, “Ceres solver.” `http://ceres-solver.org`.