Force Feedback For Reliable Robotic Door Opening

Nikolaus Adrian Wittenstein

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Brian Lattimer, Chair
Alexander Leonessa
Tomonari Furukawa

July 17, 2015
Blacksburg, Virginia

Keywords: force feedback, door opening, mobile manipulator, impedance control, guarded move

Copyright 2015 Nikolaus Adrian Wittenstein
Opening a door is still a hard problem in robotics. Many robotic manipulators use open-loop position control to open doors, which reduces reusability and reliability in the face of slight differences or sensor errors. Many others use force feedback or impedance control but skip past the problem of grabbing the handle, which could lead to failures due to sensor errors. This research assumes that perception is faulty, and uses joint-level force feedback to probe the location of the door and its handle before attempting to open it. The resulting control strategy is at least 33% faster than the open-loop control system it replaces, and had an 83% success rate during testing in place of the previous method’s 60% success rate.

This work was supported by HDT through the lending of a pair of Adroit manipulator arms, by ONR through grant N00014-15-1-2128, and by DARPA through grant N65236-12-1-1002.
To Emily, for making me smile every day that I was writing this thesis.
Acknowledgments

Thanks to everyone in TREC who supported my work, to my professors, and especially to Dr. Lattimer for his advice and guidance.

All photos and images by the author unless otherwise attributed.
Contents

1 Introduction ................................. 1
   1.1 Background ................................ 1
   1.2 Research objectives .......................... 3
   1.3 Overview .................................. 4

2 Previous work ................................. 5
   2.1 Manipulator control ............................ 6
      2.1.1 Teleoperation and known (a priori) models with open-loop control .... 6
      2.1.2 Online learning and model discovery ................................. 7
   2.2 Strategies and techniques ....................... 9
   2.3 Summary .................................. 10
   2.4 Contribution ................................. 12

3 Robot Platform ................................. 13
   3.1 DARPA Robotics Challenge ........................ 13
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Future work</td>
<td>51</td>
</tr>
<tr>
<td>Bibliography</td>
<td>53</td>
</tr>
</tbody>
</table>
List of Figures

3.1 The CAD model of ESCHER (Used with permission of Coleman Knabe, 2015) 15

3.2 Manipulation software architecture diagram. The main stack used for door-
opening is highlighted, with associated elements faded slightly. 18

3.3 Wrist kinematic chain order of ESCHER’s HDT Adroit arm. The wrist rotates
at the forearm, and can abduct and flex. (Used with permission of Coleman
Knabe, 2015. Joint arrows added by the author) 20

3.4 Wrist kinematic chain order of ESCHER’s Dynamixel Pro arm. The wrist
rotates at the forearm, followed by a flexion and another rotation joint. (Used
with permission of Coleman Knabe, 2015. Joint arrows added by the author) 21

3.5 ESCHER Reachability (3/4 view). Warmer colors represent a higher number
of kinematic solutions to the target pose. 22

3.6 ESCHER Reachability (side view) 22

3.7 THOR Reachability (3/4 view) 23

3.8 THOR Reachability (side view) 23
4.1 Operator Control System (using RViz). The door handle template (purple) has been inserted to coincide with the LIDAR point cloud of the door handle (grey). The robot is a visualization based on proprioception of joint states.

4.2 The guarded move is the core of force-feedback object discovery approach. It closes the loop around MoveIt!

4.3 The hand is instructed to move through the door during the guarded move. The current proprioceceived position is shown as opaque, and the movement goal is semi-transparent.

4.4 The hand starts away from the door and handle.

4.5 The fingers are closed so that they can feel the door.

4.6 The hand moves forward until the fingers feel the door.

4.7 The hand moves back slightly and closes the fingers somewhat.

4.8 The hand moves down until the wrist senses the handle.

4.9 The hand moves back up to avoid sliding on the handle.

4.10 The hand touches off on the handle cylinder.

4.11 The hand moves out from the handle cylinder a handle’s length.

4.12 The hand moves back down again and senses the handle.

4.13 The hand moves diagonally to unlatch the door.

4.14 The arm pushes the door.
4.15 Finger forces. At $t \approx 3.7s$, the fingers collide with the door. At $t \approx 4.5s$, the ring finger slips on the door. All times are relative to the start of the motion toward the door.

4.16 Wrist forces. All times are relative to the start of the motion down toward the handle. The decreases in force occur when the hand slips sideways toward the end of the handle, a problem which was alleviated by adding the handle cylinder touch-off.

4.17 Backlash in the arm. The upper and lower pictures were taken at the same nominal arm position according to the motor controllers. The hand was able to be moved manually by about 7cm.

4.18 Threshold adjustment. The hand starts in collision with the door, with the wrist joint loaded. As the wrist moves down, the force drops drastically, but the threshold tracks the wrist force so an increase is still regarded as sufficient to stop the hand.

4.19 Finger state when performing a handle cylinder touch-off for a right-hand door handle. The index finger is extended to avoid catching on the end of the handle, whereas the ring finger, which is already inside of the handle, is flexed more to avoid the handle rosette.

5.1 The locations of the door handle and hand relative to the robot for each of the eighteen trials, with successes and total number of trials marked in each area. Green X marks represent range of motion limits for a 45-degree handle orientation.
5.2 The height of the door handle is shown by the plane drawn at about the robot’s elbow height. If the handle lies outside the reachability sphere, the robot will be unable to manipulate it.
List of Tables

2.1 Prior art ................................................................. 11

5.1 Runtimes for 13 trials of the FFOD method, before and after the reliability improvements. ......................................................... 46

5.2 Test course results for the open-loop a priori model manipulation method. Times with a plus sign were derived from video recordings of tests which had been edited to remove operator time; actual times are much higher than evidenced by the recording. ......................................................... 46

5.3 Results. Eighteen trials are shown, from three different robot poses. Fifteen trials are successful, for a success rate of 83%. ................................................................. 48
Chapter 1

Introduction

1.1 Background

Factory robots have been in use for many years, accurately and precisely assembling, welding, and painting parts on assembly lines. These robots only succeed so well, though, because their inputs—the new parts that come down the line, the tools and components—are identical each time. Each body panel that an automobile factory robot bolts onto the frame comes to the robot in exactly the same orientation and position, and likewise the pose of the frame is identical every time. These robots use open-loop controllers and pay no attention to the world they inhabit. They have been trained to, for example, grab the body panel from a certain location, and if the body panel is not there, they will follow their instructions to the letter and grab the air (or whatever else is in its place), swing it over, and bolt it onto the body. Given this description, open-loop position control sounds malfunctional, but in practice it works extremely well in a carefully structured environment like a factory floor.

Modern robots are beginning to leave the factory. They are moving more and more into
unstructured environments, from the Roomba that vacuums the floor to medical assistance robots designed for use in the home, such as described in Hanebeck et al. [1]. For these robots, successful operation is much harder. Unlike factory robots, whose programming relies certainty on the fact that it knows the correct pose of every relevant object in their world, these new robots must perform in environments where they have incomplete knowledge. In many cases, this incomplete knowledge leads to failure: a Roomba becomes stuck on a cable while vacuuming the carpet, requiring human intervention to be saved. An assistive robot attempts to open a door but misses the door handle. Hanebeck et al. [1] themselves assert that service robots need to increase in reliability and robustness before they can be introduced to the general public.

To enable these robots to succeed, there has been a push away from the factory-robot method of open-loop position control. As early as 1984, Hogan [2] pointed out that in dynamic interactions it is insufficient to merely control position or force. Instead, robots in unstructured humanoid environments need to be able to cope with human situations by acting in human ways, such as by using impedance control or other closed-loop control methods. Rather than playing back canned motions, they need to be able to adapt to differences and unknowns. Chung et al. [3] agree that the hardest problem is robotics is succeeding in the face of uncertainty.

In the recent Defense Advanced Research Projects Agency (DARPA) Robotics Challenge [4], which focused on disaster response, Virginia Tech’s TREC lab fielded a robot named ESCHER (Electric Series Compliant Humanoid for Emergency Response) which could walk and manipulate. As part of this challenge, the robot was required to open a door as the first in a set of manipulation tasks.

In competition, ESCHER failed to open the door. The fingers would consistently close on air, or the hand would touch the door handle but slip off when attempting to turn it. In all
cases, it was difficult to tell from the Operator Control System what the problem was. The world data from the light detection and ranging sensor (LIDAR) and the cameras made it appear as if the fingers were on the handle.

1.2 Research objectives

This research proposes, implements, and validates a method that uses force feedback control to open the door. It gives ESCHER the ability to open a door much like a human would in the dark: not trusting perception sensors (such as LIDAR or cameras) but using proprioception to become certain about the location of the door and the door handle. It gives ESCHER the ability to feel for the door using the force sensors in its joints, which in turn gives it the ability to succeed with incomplete or incorrect knowledge of the door handle location.

This ability is important for a few reasons: it can compensate for faulty or miscalibrated perception sensors, such as a LIDAR that shows the door to be closer than it really is. It could make up for a whole-body controller, which is moving the body to maintain balance while the arm moves, by moving the arm until it hits the door rather than just moving the arm in a torso-relative move until it coincides with where the door should have been if the robot had not just shifted its weight.

The overarching objective of this research is to start with a handle location relative to the robot, and to end with an open door. The system is fully autonomous and able to compensate for sensor error through intelligent use of force feedback. While others have used online force measurements to learn a door model as the door is opened, most of these assume perfect knowledge of the handle location. None of the previous work in the field uses online force feedback to discover the location of the handle itself and then autonomously open the door.
1.3 Overview

This thesis is organized in the following manner. In Chapter 2 previous work in the field of robotic door opening is examined and discussed. Section 1.1 includes the background and description of the robot platform with specific focus on the robot manipulators. The proposed force control approach is provided in Chapter 4. In Chapter 5 the results of using the robot to perform force feedback-controlled door opening is provided, along with a discussion of reliability. Chapter 6 contains the conclusions and recommendations of future work.

The coordinate frame used throughout this thesis is a standard right-handed coordinate frame, with $+x$ forward, $+y$ left, and $+z$ up.
Chapter 2

Previous work

Almost all of the previous work in this field deals with one of two problems: door handle detection or door opening. The work presented here deals with the end-to-end problem of finding the door and handle, for which force feedback is used, followed by turning the handle and opening the door.

In general, handle detection is done using cameras or LIDAR, though some authors, such as Chung et al. [3] also use force feedback to ensure successful grasp. No works were found where force feedback was used as the primary method for locating the door or door handle.

Most of the research that focuses on opening of doors has two features in common: it assumes perfect knowledge of the handle location, and it concentrates on moving the end effector in a smooth path that tracks the door handle as the door opens. Some approaches, such as that of Schmid et al. [5] and Petersson et al. [6] even require the operator to put the robot’s end effector on the door handle. This research does neither: it assumes that the provided handle position is fallible, and does all it can to determine the true location. It also avoids the problem of moving the handle in a perfect arc that matches the door radius.
and location: rather than creating a model of the door, it simply pushes, following Niemeyer
and Slotine’s [7] advice of letting the structure guide the motion.

In summary, the presented system manages both finding the door handle and opening the
door, and does this without relying on low-level control of the arm joint poses.

2.1 Manipulator control

Contemporary door-opening techniques run the gamut from almost completely teleoperated
as presented by Kobayashi et al. [8] to completely automated as in Ott et al. [9]. These
approaches can be roughly sorted into two categories:

1. teleoperation or using known (a priori) models with open-loop control

2. online learning or model discovery.

2.1.1 Teleoperation and known (a priori) models with open-loop
control

This first category includes the original control techniques used in robotics, going back
to automatons that existed even before robot was a recognized term. It extends to high-
precision robotics such as factory robots, which rely on repeatably identical environments in
order to function, and to robots that lack fully autonomous behavior, such as bomb-disposal
or surgery robots. Teleoperation and open-loop control are roughly equivalent; they differ
only in time and in loop closure: trajectories recorded beforehand and then played back as
open-loop position control lack only the continuous operator feedback present in trajectories
followed online during teleoperation.
An example of open-loop position-controlled door opening is the technique used by Brooks et al. [10], which requires that the door be ajar: the arm simply moves through where it thinks the door should be, and the door is expected to be pushed open. There is no feedback to determine whether the door was where it was expected to be or whether it opened successfully. This amount of automation is still more than is used by other groups. Another teleoperated robot, used by Kobayashi et al. [8], moves its gripper near a doorknob and then waits for an operator to manually move in for the final approach.

This category also includes approaches such as that used by Gray et al. [11]; their PR2 robot can open doors whose size and handle positions are preprogrammed. Once a door has been programmed, the robot can autonomously open it by playing back specific motions based on the door dimensions. Another example comes from Gaspers et al. [12], who use a bomb-disposal robot with no force feedback to open a door. In this approach, the manipulator grasps a 2D LIDAR and moves it up and down to create a 3D model of the door. The operator then selects the door plane, hinge, and handle, and the algorithm uses this model to open the door. This technique is designed for robots that are normally teleoperated, such as those used for bomb disposal, which have no force feedback and no sensors other than a camera.

Finally, the technique for door opening originally used by ESCHER falls into this category. This method, described in more detail in section 4.1, has a human operator specify the handle location and then move the hand to predefined relative locations.

### 2.1.2 Online learning and model discovery

This category is distinguished from the previous one chiefly through its use of autonomy. Giving a robot the ability to adjust its manipulation plans without intervention from
an operator can make it more robust, and can potentially give it the ability to open a
door without any operator assistance. Online learning refers to a robot using its sensors
to determine how to manipulate an object in the environment during the process of
manipulating that object. Model discovery is the process of generating an internal model
during this online learning, at which point a robot can act on the object according to the
model it has created.

In 1997, Niemeyer and Slotine \cite{7} presented a seminal paper of a robot using online learning,
and their approach is remarkably straightforward. They used the simple approach of applying
force in the direction of estimated velocity; that is to say, following the path of least
resistance. They note that the door will continue to move as long as the force vector points
outside of the door hinge’s friction cone. Their work was effective and simple to the point
that they skip the model discovery altogether. It is also especially important in relation
to the current work because it applies impedance control to environment learning and door
opening. Later work in this area, such as from Petersson \textit{et al.} \cite{6} and Karayiannidis \textit{et al.} \cite{13}, used the door velocity to estimate the radius and used the radius to determine the
force to apply. This intermediate step is model discovery.

Kim \textit{et al.} \cite{14} used a similar technique to twist door handles. They use a compliant
manipulator to rotate a door handle, and as the handle rotates they read the angle of
the wrist to update their estimate of the axis of rotation of their handle model. They do
not, however, appear to subsequently open the door.

Klingbeil \textit{et al.} \cite{15} also made a robot that can autonomously turn door handles. It finds a
door handle using computer vision combined with LIDAR, then turns the handle and pushes.
Their method automatically creates a model of the handle’s operation from the perception
data, but does not update this model using feedback from the manipulator. They go slightly
further than Kim \textit{et al.} \cite{14}, in that they appear to crack the door open rather than stopping
once the handle is turned, but it is unclear whether the robot can open a door to the extent that it could travel through. Though they create a model of the door handle, they do not create a model of the door. This is similar to the technique presented here, which creates a simple model of the handle location but does not depend upon a model of the door itself.

2.2 Strategies and techniques

In between the overarching approaches that are used in the literature, many authors have invented strategies and techniques to solve certain subtle problems that their robots face when opening doors. As early as 1978, Shimano [16] realized that it is a good idea to put a force sensor in the wrist rather than trying to infer hand forces from forces further up the kinematic chain. A force sensor close to the hand eliminates any dynamic effects from the measurements and removes the need to calculate any kinematics or dynamics. Shimano [16] also noted in the same thesis that force sensors in fingers joints can be easily repurposed as touch sensors, a technique which is also used in the approach presented in this work.

Guarded moves have also been used previously in door opening but not to the same extent used here. Petersson et al. [6] use visual servoing to bring the manipulator close to the door handle, but once the manipulator occludes sensor’s view of the door handle, the manipulator makes a guarded move—in their implementation, the manipulator moves toward the handle until the force sensor reads that contact was made. Velocity can also be monitored instead of joint torques when performing a guarded move, as described by Brooks et al. [10] in their 2004 work.

One of the reasons a guarded move is so effective is neatly summed up by Rhee et al. [17], who point out that a mobile robot’s positional error is about an order of magnitude greater than the positional error from its vision system, which is in turn about an order of magnitude
greater than the positional error after touching an object. This means that getting close to a door is not enough and neither is seeing the door handle. Touching the door and handle is the best strategy for determining where they are.

2.3 Summary

A summary of prior art in the field is presented in Table 2.1. Each system is summarized briefly by which approach is used and the method of feedback. Interestingly, many robots forgo force feedback entirely, while others use force feedback to generate a door model; very few use proprioception at all to discover or validate the door or handle location.
<table>
<thead>
<tr>
<th>Author</th>
<th>Year</th>
<th>Robot</th>
<th>Arm DOF</th>
<th>Door model</th>
<th>Feedback</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nagatani and Yuta</td>
<td>1995</td>
<td>YAMABICO type-Ten</td>
<td>6</td>
<td>None</td>
<td>Wrist f/t</td>
<td>Manipulator pushes on door and is compliant in door plane.</td>
</tr>
<tr>
<td>Niemeyer and Slotine</td>
<td>1997</td>
<td>WAMS</td>
<td>4</td>
<td>Learned</td>
<td>Velocity</td>
<td>Apply torque in direction of velocity.</td>
</tr>
<tr>
<td>Peterison et al.</td>
<td>2000</td>
<td>Puma 560</td>
<td>6</td>
<td>Learned</td>
<td>Wrist f/t</td>
<td>Estimate radius from velocity.</td>
</tr>
<tr>
<td>Brooks et al.</td>
<td>2004</td>
<td>Cardea</td>
<td>4</td>
<td>None</td>
<td>Joint torque</td>
<td>Does not turn the handle. The door must be ajar.</td>
</tr>
<tr>
<td>Kim et al.</td>
<td>2004</td>
<td>HomBot</td>
<td>5?</td>
<td>None</td>
<td>Wrist force</td>
<td>Force sensors to minimize internal forces in position-controlled path.</td>
</tr>
<tr>
<td>Rhee et al.</td>
<td>2004</td>
<td>PSR1</td>
<td>6</td>
<td>Learned</td>
<td>Fingertip force</td>
<td>Shake the unlatched door to estimate the model, then position control.</td>
</tr>
<tr>
<td>Kobayashi et al.</td>
<td>2008</td>
<td>UMRS 2007</td>
<td>4</td>
<td>None</td>
<td>None</td>
<td>Teleoperated.</td>
</tr>
<tr>
<td>Chung et al.</td>
<td>2009</td>
<td>PSR1</td>
<td>6</td>
<td>Learned</td>
<td>Fingertip force</td>
<td>Hybrid position/force control.</td>
</tr>
<tr>
<td>Klingbeil et al.</td>
<td>2010</td>
<td>STAIR</td>
<td>5</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Gray et al.</td>
<td>2011</td>
<td>PR2</td>
<td>7</td>
<td>A priori</td>
<td>None</td>
<td>Estimates handle axis from online manipulator pose.</td>
</tr>
<tr>
<td>Kim et al.</td>
<td>2011</td>
<td>SAP1</td>
<td>6</td>
<td>A priori</td>
<td>Joint torque</td>
<td></td>
</tr>
<tr>
<td>Hentout et al.</td>
<td>2012</td>
<td>RobuTER/ULM</td>
<td>6</td>
<td>A priori</td>
<td>Wrist f/t</td>
<td>Estimates velocity to determine radius. Arm has 7 DOF but 5 are frozen.</td>
</tr>
<tr>
<td>Karayiannidis et al.</td>
<td>2012</td>
<td>Custom</td>
<td>2 (7)</td>
<td>Learned</td>
<td>Wrist f/t</td>
<td></td>
</tr>
<tr>
<td>Rühr et al.</td>
<td>2012</td>
<td>PR2</td>
<td>7</td>
<td>Learned</td>
<td>Fingertip pressure sensor.</td>
<td></td>
</tr>
<tr>
<td>Gaspers et al.</td>
<td>2013</td>
<td>tEODor</td>
<td>7</td>
<td>A priori</td>
<td>None</td>
<td>Open-loop based on operator-specified model.</td>
</tr>
<tr>
<td>Winiarski and Banachowicz</td>
<td>2013</td>
<td>KUKA-LWR4+</td>
<td>7</td>
<td>Learned</td>
<td>Joint torque</td>
<td>Impedance control and null space optimization.</td>
</tr>
</tbody>
</table>
2.4 Contribution

The main contribution of this work is the guarded move framework, discussed in more detail in section 4.2. This framework, which allows the robot to carefully move its arms to feel and explore its environment, enables many different behaviors and overall allows for the research objective of using force feedback to open a door. It lets the robot controller use joint force sensors to locate the door and the door handle, starting from the assumption that perception is incorrect. It makes it easy for the robot to open a door, beginning with its hand off the handle. Finally, this framework is written in such a way that it is easily applicable to other tasks and transferrable to other robots: any robot that has force feedback, and uses MoveIt! on top of ROS in its manipulation stack can use the guarded move framework to aid in its manipulation.
Chapter 3

Robot Platform

3.1 DARPA Robotics Challenge

DARPA has a history of organizing competitions to advance the state of the art. In the field of robotics, they hosted two Grand Challenges for autonomous vehicles in 2004 \cite{25} and 2005 \cite{26}, along with a third Urban Challenge in 2007 \cite{27}. In the end of 2012, they announced a new competition: the DARPA Robotics Challenge.

The DARPA Robotics Challenge \cite{4} was an international competition to create robots that were capable of responding to and assisting during natural and man-made disasters. Inspired by the disaster at the Fukushima Daiichi Nuclear Power Plant in Japan, in which a robot capable of entering the plant could have mitigated some of the damage, the competition required robots to be able to enter buildings, navigate over rubble and stairs, and perform manipulation tasks such as opening a door, using a power tool, and turning a valve.

The Challenge ran over the course of two years and a half years, from about January 2013 to June 2015. Twenty-five teams qualified for the finals, and ultimately twenty-three of these
competed in Pomona, CA on June 5 and 6, 2015. One of these teams was from Virginia Tech, and fielded a humanoid robot named ESCHER, described below.

3.2 ESCHER

The work in this thesis was tested on the humanoid walking robot ESCHER, shown in Figure 3.1 which was designed and built by students in TREC at Virginia Tech as its entry into the DARPA Robotics Challenge as Team VALOR. ESCHER, whose name stands for Electric Series Compliant Humanoid for Emergency Response, has 38 degrees of freedom: 6 per leg, 7 per arm, 3 per hand, and 4 in the torso and head. It stands 180cm tall, but weighs only 75kg and has a runtime of up to 1.5 hours on lithium polymer batteries. As an entry into the DARPA robotics challenge, its design and sensor suite is geared toward robust operation in demanding environments. ESCHER’s walking ability combined with its humanoid arms qualify it as a mobile manipulator, in the sense that Ott et al. define it, as a manipulator attached to a mobile platform.

ESCHER’s arms are HDT Adroit Manipulators, made and generously loaned to the lab by HDT Global. The arms are made of reconfigurable self-contained actuators, in this case assembled in a 7-degree-of-freedom configuration. Each joint in the arm and hand is capable of impedance control and force feedback, and the arm is strong enough that it can push the whole robot over against the whole-body controller bracing with the legs. Each hand has a thumb with two degrees of freedom allowing both flexion and abduction, and two fingers (referred to as the index finger and ring finger) capable of flexion. If the thumb is abducted and flexed, it is capable of apposition with the flexed index finger. The thumb and fingers are all underactuated, giving them the ability to easily grab objects without the added weight and complexity of more motors. This also allows the fingers to hook around a door handle
Figure 3.1: The CAD model of ESCHER (Used with permission of Coleman Knabe, 2015)
or other object as they are closed, preventing the hand from slipping off.

The legs are custom designed and fabricated by Virginia Tech students. All of the leg actuators are series elastic, described by Knabe [29] and based on the ideas presented by Pratt and Williamson [30]. The hip and ankle are parallely actuated biarticular joints, as originally designed by Lahr [31], which lends them additional strength.

ESCHER was designed to replace THOR, which was TREC’s previous humanoid robot. A discussion of the relevant differences can be found in subsection 3.2.3.

3.2.1 Sensors

Most of ESCHER’s sensing is proprioceptive, measuring joint angles, forces and torques, as well as body acceleration and angular velocity. Only two sensors are strictly perceptive: a stereo vision system and a rotating scanning laser rangefinder. The arms have a load cell and encoder in each joint, including in the finger joints, giving the ability to sense torque and to use impedance control. For perception, ESCHER is equipped with a Hokuyo UTM–30LX-EW scanning LIDAR [32] which continuously rotates 360 degrees to create a 3D point cloud, and a MultiSense S7 from Carnegie Robotics [33] which creates a colored depth map using stereo cameras. The software combines these two sensors into a colorized depth map where they overlap, and a colorless depth map where the LIDAR’s 270-degree field of view falls outside of the 80-degree field of view of the MultiSense.

3.2.2 Software

ESCHER has a complicated software stack, shown in Figure 3.2. In broad strokes: it starts at the bottom with leg actuator drivers. Just above these lies the whole-body controller,
which communicates directly with the legs and indirectly with the arms to shift the robot’s center of mass for balancing. Next comes a translation layer which essentially wraps the whole-body controller into a Robot Operating System (ROS) node. Above this lies the rest of the ROS ecosystem, comprising the operator interface, all of the sensing, and the arm drivers.

The whole-body controller was developed entirely at Virginia Tech. It is capable of using impedance control in the legs and position control in the upper body to balance and walk on a variety of surfaces. More detail can be found in Hopkins et al. The remainder of the system uses ROS to tie the robot together. The MultiSense, Hokuyo LIDAR, KVH IMU, Dynamixel motors, and Adroit arms all run as ROS nodes. They communicate in standard ROS fashion by passing messages and otherwise running independently from each other. The operator control station subsystem ties all of these nodes together and gives the operator the ability to visualize the sensed world and the robot in it, and send footstep plans and manipulation commands. Much of the operator control station and high-level planning software was contributed by ViGIR, another contender in the DARPA Robotics Challenge.

Most relevant to this work is the manipulation stack. The core of this stack is MoveIt!, a reasonably mature library which is well-integrated with ROS. MoveIt! is able to perform inverse kinematics to generate arm positions for given hand poses, create motion trajectories from one hand pose to another (or one joint configuration to another), and avoid collisions during these plans. MoveIt! generates plans that allow the door-opening controller itself to forgo any joint limit avoidance, singularity avoidance, or null space damping as e.g. described by Winiarski and Banachowicz.

The guarded move functionality and door-opening controller itself, as described in this work,
Figure 3.2: Manipulation software architecture diagram. The main stack used for door-opening is highlighted, with associated elements faded slightly.
both sit intentionally above MoveIt!, meaning that their only communications with the robot are through the MoveIt! interface. This gives them the advantageous ability to be easily modified to work with any robot that uses MoveIt! and has arm force feedback, but at the same time increases latency and precludes the ability of doing any manual low-level joint control. MoveIt! does not provide feedback of trajectory execution status (such as whether the arm has finished moving), so this status is received directly from the controller.

The parts of the software stack most relevant to manipulation are shown in Figure 3.2 and the parts in the diagram most directly related to door opening are shown in black. As can be seen from the diagram, the door-opening system only communicates with the rest of the stack by issuing commands to MoveIt!. Because MoveIt! provides no feedback of whether a trajectory has succeeded or failed, the guarded move component must listen directly to the component executing the arm trajectories. It also listens to joint states in order to determine the torque at each joint; in this way, it is able to execute a true guarded move by canceling motions when forces reach a predetermined threshold.

### 3.2.3 Kinematic chain

One of the biggest improvements in ESCHER over the previous robot THOR is a complete replacement of the arms. THOR used arms that were made up of Dynamixel Pro motors, which were used structurally and connected to each other with brackets. Both the motors and their communication protocol were unreliable, leading to continuous motor replacement and firmware maintenance. The new arms on ESCHER are Adroit Manipulators, which are designed by the defense contractor HDT. These new lightweight arms were made for prosthetics and have many other advantages over the Dynamixel arms: they are much stronger, they are not backdriveable (so when the robot is emergency stopped they remain
in place rather than falling and hitting the body), and they use CAN instead of a custom serial protocol. They also have a more advantageous kinematic chain.

Up through the elbow, the Dynamixel arms and the HDT Adroit arms have identical kinematics. Each shoulder has three rotational joints with intersecting axes, which together emulate a spherical joint: a flexion joint, an abduction joint, and a rotation joint. The elbow then has a single flexion joint. At the wrist is where the HDT manipulator outperforms the Dynamixel arm. The HDT wrist has a setup similar to that in the shoulder, of a rotation joint, a flexion joint, and an abduction joint (see Figure 3.3). Only two axes intersect, unlike the three in the shoulder, but this still gives the wrist a semblance of a spherical joint. The Dynamixel wrist, on the other hand, has a rotation joint, a flexion joint, and a rotation joint, as shown in Figure 3.4. This means that not only does it not emulate a spherical joint, but in the nominal configuration, the Dynamixel wrist is in gimbal lock.

![Figure 3.3: Wrist kinematic chain order of ESCHER’s HDT Adroit arm. The wrist rotates at the forearm, and can abduct and flex. (Used with permission of Coleman Knabe, 2015. Joint arrows added by the author)](image)

The design of the HDT Adroit arms is also much closer to the design of the human arm. The human arm works quite well, making it a good model to follow, and this also allows intuitive reasoning about reachability without having to rely on inverse kinematics solvers. This was a particular problem with the Dynamixel arms: frequently, poses that seemed quite reasonable were unable to be reached by the solver, and it would turn out that the pose was not in fact reachable because of the kinematic order of the arms.
A reachability analysis was performed on the HDT Adroit and Dynamixel arms to evaluate the number of arm configurations possible to reach given hand poses. The open-source OpenRAVE [38] toolkit, which is a library geared toward manipulation planning and analysis for field robotics, was selected for this purpose. It has the ability to generate a kinematic reachability diagram for arbitrary robot manipulators. These diagrams were generated for both ESCHER and THOR, the former with its HDT Adroit arms and the latter with its Dynamixel arms. In the model, the HDT arms are shown with bent elbows; this is only a change in nominal position from THOR and does not affect the reachability. Both the HDT and Dynamixel elbows have the ability to bend backward (that is, hyperextend compared to a human elbow) and in general the arms have similar ranges of motion.

[Figure 3.5] and [Figure 3.6] show ESCHER’s reachability using the HDT arms. The warmer the color, the higher the number of kinematic solutions for the arm to get the end effector to the given position. Blue represents the fewest possible arm configurations for a given hand position, with movement in hue toward red signifying more configurations: green allows more configurations than blue, and yellow allows more than green. It can be seen that ESCHER can reach almost anywhere within the limits of the length of its arm, and has a large amount of overlap and freedom in a large subset of that space. This overlap and freedom, given by the additional possible arm configurations, shows ESCHER’s ability to hold its hand in more
Figure 3.5: ESCHER Reachability (3/4 view). Warmer colors represent a higher number of kinematic solutions to the target pose.

Figure 3.6: ESCHER Reachability (side view)
orientations for each position, as well as its ability to hold its arm in more configurations in order to reach a hand position while avoiding obstacles. This allows the robot to, for example, move its hand to a certain position and then rotate the hand without moving it, or, conversely, to hold its hand steady while reconfiguring the arm around obstacles or in preparation for another movement.

THOR’s reachability is shown in Figure 3.7 and Figure 3.8 and can be seen to be worse than ESCHER’s. The blue region is only slightly smaller than for the HDT Adroit arms, but the green region, representing the ability to reach a position in space from more arm configurations than are possible in the blue region, is much smaller, especially in that crucial
space in front of the robot where most manipulation happens. And whereas the HDT arms have a large yellow region, representing the most possible arm configurations, the yellow region for the Dynamixel arms is almost nonexistent. With this arm, THOR could reach a similar number of positions as ESCHER, but with a much smaller number of hand orientations and a much more constrained arm configuration.
Chapter 4

Approach

The door-opening method presented here, referred to as the force-feedback object discovery method (FFOD) replaces the previous manual method used on ESCHER. The manual method is described briefly below for comparison with the new FFOD.

4.1 Open-loop a priori model manipulation

The original manipulation system on ESCHER was manually controlled and used templates with open-loop approach trajectories. The template, an STL model of a door handle, was placed by the operator into an RViz visualization of the robot sensor data in the Operator Control System (OCS) and aligned by hand with the LIDAR point cloud of the real-life door handle, as shown in Figure 4.1. The operator could specify that the hand should move to a predefined pose relative to this template, then to a second closer pose, referred to as the pre-grasp and the grasp pose in the user interface. At any point, the operator could move the fingers to a predefined grasp posture, such as bending the fingers to catch the door handle.
The problem with this method was that the hand would move to the instructed position (or as close as it could get) without regard for the actual position of the handle. If the hand did not move to the correct position in real life, the template would be moved slightly to encourage movement of the hand, and the hand would be instructed to move to the grasp pose again. This cycle would iterate until the hand appeared to be in the correct position. Often, attempting to close the hand around the door handle would reveal that the hand was not close enough to grasp the handle, and the fingers would close on air.

The old method required a trained operator, someone who had practiced with the method and drilled door opening procedure. The method was finicky and widely variable in its success rate, and ultimately did not work: in competition, ESCHER was unable to open a door using the grasp system that had been practiced for so long.
4.2 Force-feedback object discovery manipulation

4.2.1 Overview

From a high level, the system presented here works like this: the operator marks the approximate position of the door handle in the world model. ESCHER then reaches toward its best assumption of where the door should be until it actually touches the door, rather than just reaching toward where it thinks the handle should be. With the knowledge of the door position, it reaches down until it touches the handle. It touches the handle from above and then from the side in order to guarantee it knows the position. At this point, the robot has constrained the handle position in three dimensions, and is able to push the handle to unlatch and then open the door. This system is referred to as the force-feedback object discovery (FFOD) method because it uses joint force feedback to develop a model of the handle location.

The new FFOD method uses the same template placement as the previous open-loop method, but otherwise has no similarities. Even the use of the template is improved: whereas the open-loop method was incredibly sensitive to the placement of the template, the force-feedback approach assumes that the template has been placed incorrectly and discovers where the door handle really is. Whereas the open-loop method was completely manually controlled, the FFOD method is fully autonomous: instead of the operator closing the loop by examining the scene in the OCS, the control loop is closed automatically through the use of guarded moves.

The guarded move is the core of the new approach. As defined by Will and Grossman [39], a guarded move is a motion that halts at the occurrence of an expected sensor input. In this case, MoveIt! is instructed to move the hand from one pose to another, while the guarded
move subsystem monitors certain joint force proprioceptors to make sure the forces stay below a specified threshold value. If the forces increase beyond the threshold, the movement is stopped. Figure 4.2 shows how the guarded move fits in with the rest of the system.

Figure 4.2: The guarded move is the core of force-feedback object discovery approach. It closes the loop around MoveIt!.

Not all moves in the plan are guarded moves, but most are. Whitney [40] distinguishes between gross (open-loop) and fine (closed-loop) motions. In this case, the motion from the
arm’s home position to the approach position in preparation for touching off on the door is an example of a gross motion, run open-loop and unguarded. The motion from this approach to the door touch-off is a fine motion, moving slowly and guarded by the finger force sensors. In 1981, Mason [41] wrote that motion plans can be separated into a sequence of guarded moves and compliant (such as impedance-controlled) motions. In this case, the guarded moves are themselves compliant from the impedance control of joints in the manipulator: for example, the fingers are impedance-controlled when they are feeling for the door, meaning they are compliant but also guarding the move.

Another large difference between the open-loop method and the new FFOD method is in the use of impedance control. The old method used impedance control to make up for mistakes in pose commands: if the hand was instructed to move into the door, for example, the impedance control in the fingers would allow the fingers to bend and the arm to crash slowly into the door without doing any damage to the door or the robot. This is markedly different than the use of impedance control in the new method: here, the impedance control is used every time the door is opened, as the hand intentionally collides with the door to discover where it is. Rather than being used as a failsafe, it is used as a way of interacting with and sensing the environment.

The details of the FFOD manipulation approach are described in the following sequence of steps, and in Figure 4.4 through Figure 4.14. A video of the process can also be seen on the Internet [42].

### 4.2.2 Algorithm

1. **Initialize** the robot and read the supposed location of the handle from the Operator Control Station.
2. **Lift the hand** to an approach pose 15cm away from the door and 15cm above the handle, as shown in [Figure 4.4](#). This pose is above the handle to ensure that the hand will not collide with the handle if the handle’s height is wrong, and far from the door in case it is closer than expected. The LIDAR accuracy is $\pm 3$ cm [32], and the largest observed miscalibration has been no larger than 4cm, so 15cm gives a large buffer to ensure the ability to correctly find the door.

3. **Bend the fingers** to a 30-degree angle, as shown in [Figure 4.5](#), so they can be used to detect the door. The joints measure only torque, so if the fingers hit the door straight on the system will be unable to tell. Similarly, the fingers cannot be bent too much, because they must be guaranteed to hit the door before the knuckles do.

4. **Move forward** until the fingers detect force. This move forward instructs the arm as far forward as it is physically able to move, stopping when it hits the door. This means that the robot can compensate for an arbitrary error in the $x$ direction as long as the door is still within reach. [Figure 4.3](#) shows the planned trajectory as shown in the OCS, while [Figure 4.6](#) shows how the arm stops the trajectory once it touches the door. At the end of this step, the **handle position is constrained in** $x$.

5. **Free the hand** by moving out 2cm so that the fingers are no longer forced against the door, and close the fingers more (shown in [Figure 4.7](#)). This prevents the fingers from scraping along the door, which prevents unwanted forces in the wrist, prevents the fingers catching on the knob rosette, and, in the case of a paneled door, prevents the fingers catching on the lock rail. The index finger, at 45 degrees, is closed slightly less than the ring finger at 60 degrees (see [Figure 4.19](#)) to avoid catching on the end of the door handle when finding the handle cylinder.

6. **Find the handle.** Move down until the wrist flexor detects the force of the door
handle’s return spring (Figure 4.8). Here the handle position is constrained in $z$.

7. **Free the hand again**, as shown in Figure 4.9 by moving up until it is no longer touching the handle. This allows the hand to move toward the handle cylinder without dragging on the handle.

8. **Find the handle cylinder**. Move right until the wrist abductor detects force from the door handle cylinder, depicted in Figure 4.10. After this move, the handle position is constrained in $y$.

At this point, the robot has found the door, which constrains the solution in $x$; the handle, which constrains the solution in $z$; and the handle cylinder, which constrains the solution in $y$. The position of the handle is now fully known.

9. **Move above the handle** to get ready to open the door. The hand is positioned at an optimal grasp point near the end of the handle, as can be seen in Figure 4.11.

10. **Find the handle** again the same way as before to ensure that the hand is correctly positioned (Figure 4.12).

11. **Unlatch the door** by moving down and toward the handle axis as in Figure 4.13. In the current configuration, the robot normally turns the handle about 54 degrees, which is enough to unlatch all observed handles.

12. **Push on the door** to open it. The arm moves to the extent of its range in order to open the door as far as possible, and then holds it open. The opened door can be seen in Figure 4.14.

At this point, the robot could be instructed to walk forward and continue to push the door in order to get through.
Figure 4.3: The hand is instructed to move through the door during the guarded move. The current proprioceived position is shown as opaque, and the movement goal is semi-transparent.

Figure 4.4: The hand starts away from the door and handle.

Figure 4.5: The fingers are closed so that they can feel the door.
Figure 4.6: The hand moves forward until the fingers feel the door.

Figure 4.7: The hand moves back slightly and closes the fingers somewhat.

Figure 4.8: The hand moves down until the wrist senses the handle.
Figure 4.9: The hand moves back up to avoid sliding on the handle.

Figure 4.10: The hand touches off on the handle cylinder.

Figure 4.11: The hand moves out from the handle cylinder a handle’s length.
Figure 4.12: The hand moves back down again and senses the handle.

Figure 4.13: The hand moves diagonally to unlatch the door.

Figure 4.14: The arm pushes the door
Overall, the door-opening controller is intended to sit on the top of the software stack, working with the rest of the OCS and running on top of MoveIt!. It is explicitly not intended to be integrated at a low level into the whole-body controller, or to control the arm motors directly. This position in the stack allows it to integrate easily as a drop-in addition to other manipulation software, and also makes it more versatile: it requires only access to MoveIt! and force feedback from joints in the arm. Any arm that has force feedback should be controllable using this method.

4.2.3 Contact force thresholds

Contact forces to halt a guarded move were chosen to meet a few criteria. First, the contact force should be high enough that it is distinguishable from the noise in the joint force sensor’s output. Second, it should be low enough that the reaction time is as fast as possible, keeping the sensitivity high to maintain a low latency. Finally, in the case of the fingers, there is a maximum force that can be applied to bent fingers before the knuckles hit the door, based on the stiffness in the impedance controller. The force threshold to stop the guarded move should always be strictly less than this maximum force. Over all of these criteria lies the consideration that false negatives are allowable—if a contact is not registered this timestep, it may well be in the next timestep when the force has increased slightly—whereas false positives should be avoided at all costs. If a contact is sensed where none exists, then the controller will have incorrect knowledge of the environment.

Torque data was collected by instructing the hand to move into the door, and manually stopping the robot when the knuckles hit. The resulting forces are shown in Figure 4.15. As can be seen, the joint force sensors have noise with a standard deviation of 0.03Nm. The force during the collision with the door reliably rises within approximately 0.25s, allowing
easy sensing of the door location. The threshold force for the fingers on the door touch-off was set at 0.2Nm to meet the three criteria explained above. A threshold of 0.10Nm is already three standard deviations away from zero; however, the force measurement is biased at the beginning of the move based on a single sample, meaning that there is a small chance that the force could be biased to an outlier value. Doubling the threshold to 0.2Nm alleviates this concern without unduly impacting the stopping latency.

Figure 4.15: Finger forces. At $t \approx 3.7\text{s}$, the fingers collide with the door. At $t \approx 4.5\text{s}$, the ring finger slips on the door. All times are relative to the start of the motion toward the door.

The wrist flexor force was determined similarly. The hand was instructed to move down, and was stopped by the operator. The resulting force graph is shown in Figure 4.16. The wrist flexor force is slightly more complicated than the finger force for a couple of reasons. While the standard deviation of the noise in the wrist torque, at 0.033Nm, is roughly equivalent to that in the finger joints, the wrist also experiences larger actual forces than the fingers. When moving down the door, if the fingers drag at all, the wrist has to push down with a higher force, and the fingers may exhibit a behavior where they slip suddenly and catch
again. The hand can also slip sideways along the handle once it has hit it, due to backlash in the actuators. The amount of backlash is relatively high, as can be seen in Figure 4.17 which makes precision position control difficult. Luckily this disadvantage is avoided through the use of force feedback, as discussed below in the section about the handle cylinder touch-off. For these reasons, a threshold of 0.5Nm was chosen for the wrist flexor. As shown in Figure 4.18 at about the 2 second mark, a threshold of only 0.2Nm could easily register a false positive. At the torque threshold of 0.5Nm, the collision with the handle is sensed when the handle is turned approximately 8 degrees from its rest position due to the spring constant of the handle return spring.

Figure 4.16: Wrist forces. All times are relative to the start of the motion down toward the handle. The decreases in force occur when the hand slips sideways toward the end of the handle, a problem which was alleviated by adding the handle cylinder touch-off.

The way the force threshold is sensed was changed after testing revealed an intermittent problem. Originally, the initial force value at the start of the guarded move was regarded as 0 force; that is to say, the joint force sensors were automatically biased to 0 at the start of a move. This compensated easily for any bias in the sensors, as well as biasing for the force
Figure 4.17: Backlash in the arm. The upper and lower pictures were taken at the same nominal arm position according to the motor controllers. The hand was able to be moved manually by about 7cm.
of gravity without the need to calculate the Jacobian matrix at the end-effector position (Salisbury [43], among others, describes using the Jacobian to calculate joint torques). This technique had problems when the hand started in collision. As seen in Figure 4.18 which shows the wrist force as the hand moves down the door, sometimes initial forces could be quite high. In this case, as the fingers crashed into the door, they stuck and pushed the hand up as it moved forward, loading the wrist flexion joint. The fingers freed themselves when the hand moved down, leading to a drastic and sudden drop in apparent wrist force. In this scenario, the hand would never believe that it had found the handle, because it would be waiting for a force that was higher than the initial force. The thresholding was changed to a function that tracked the lowest registered force value rather than the initial force value so that a guarded move would stop when it sees any increase in force, rather than simply an increase in force over the initial value.

Figure 4.18: Threshold adjustment. The hand starts in collision with the door, with the wrist joint loaded. As the wrist moves down, the force drops drastically, but the threshold tracks the wrist force so an increase is still regarded as sufficient to stop the hand.
4.2.4 Handle position constraints

When first designed, the door opening method required the operator to exactly specify two of the three translational degrees of freedom of the handle; only its relative $x$ (that is, the distance from the robot to the door) was unspecified. The $x$ dimension was then constrained when the robot touched its hand off on the door. This approach was chosen because the largest problems with the previous method of door opening had been observed to be from an incorrect robot-to-door distance ($x$ in the door’s frame of reference).

However, to make the opening more autonomous and to reduce reliance on operator accuracy, the other two translational dimensions are now also discovered by the robot. First a touch-off on the handle to ascertain its $z$-height was added, which allows the operator to misplace the handle by a substantial amount in height. The arm approaches from 20cm above where it thinks the handle is, then moves down until it hits a handle. Next, a touch-off on the handle cylinder was added to constrain the $y$ dimension. Slipping off the end of the handle has been a problem for other authors, such as Hentout et al. [22], and was a problem for ESCHER’s hand as well before this change. This addition was crucial not just because it finds the $y$ position of the handle to accurately position the hand, but also because as a side effect it takes up all of the backlash in the arm when the hand moves into the cylinder. Before this touch-off was added, the hand could slip off the handle when the handle was angled down, because the backlash in the arm allowed it to move in the $y$ direction.

4.2.5 Reliability changes

As the algorithm was further developed and tested, additional steps were added that increased reliability of the door opening. One step that was added was a pulling back from the door after the $x$-axis touch-off. The door in the DARPA Robotics Challenge Finals
was a paneled door, as are many other doors, meaning that if the robot touched off inside of the panel, the fingers would catch on the lock rail on the way down to the handle. To clear the fingers, the hand is pulled back 2cm until the force in the fingers returns to its nominal zero value, and then the fingers are closed an extra 30 degrees as an extra precaution.

A second change is that the index finger is closed slightly less than the ring finger (see Figure 4.19) to avoid catching on the end of the door handle when finding the handle cylinder. If the hand was too far away from the handle cylinder so that the ring finger was in open air rather than behind the handle, a small orientation mismatch between the hand and the handle could cause the index finger to catch on the end of the door handle when the hand moved toward the cylinder. The torque sensors in the arm and the hand only measure torque in the direction of joint rotation, so the finger would not sense this force. It would only be sensed at the wrist abductor joint, which was unable to distinguish between this scenario and a successful collision with the handle cylinder.

Figure 4.19: Finger state when performing a handle cylinder touch-off for a right-hand door handle. The index finger is extended to avoid catching on the end of the handle, whereas the ring finger, which is already inside of the handle, is flexed more to avoid the handle rosette.
4.2.6 Latency

The forces generated in the arm can become relatively large, and the HDT Adroit arm handles them without difficulty. The force required to open the door, measured from the point on the handle where ESCHER’s hand pushes it, starts at 49.8N from when the door is closed (a force which Rühr et al. mention is “extraordinary” and “virtually impossible for the PR2”) and tapers off after 14cm (or 9 degrees of door swing) to 33.4N. These forces were measured by pulling with a crane scale attached to the outside handle.

The door-opening algorithm sits at the top of the software stack, as seen in Figure 3.2. This, combined with the fact that ROS is based around message-passing, means that there can be a considerable latency between requesting that a move plan be stopped and the move plan stopping. The door-opening controller has to receive a proprioception message from the arms and then send a halt request message to MoveIt! telling it to stop. The round-trip time for this halt request to get to MoveIt! and the response to come back to the controller is 77.6ms, with a large standard deviation of 76.8ms. This, combined with the fact that the arms have the ability to apply large amounts of force, means it is all the more important for the arm to move slowly and for the stopping-force thresholds to be sufficiently low for stopping conditions to be recognized quickly.

4.2.7 Impedance control vs. position control

During operation the fingers are constantly impedance-controlled so that they are able to deflect when coming into contact with an object. The rest of the joints in the arm are position-controlled. Experiments were conducted with impedance control of the rest of the arm, but unfortunately the position tracking became very poor between the whole-body controller and the joint-level impedance controllers. Impedance control of individual joints
was also experimented with, such as setting the wrist rotation joint to a very low stiffness when pushing down on the handle so that the hand would automatically rotate to track the handle as it turned. However, a stiffness that was low enough for the wrist to turn appreciably proved to be unstable and provided no increase in reliability over simple position control.

Ultimately, position control allowed better tracking of the arm position. Between impedance control of the fingers, the backlash in the arms, and the ability of the whole body to move in response to forces on the arm, impedance control of the arm itself was proved to be unnecessary. For example, when touching off on the door handle cylinder, the guarded move observes the force in the wrist abductor joint. This joint is in position control, but when it touches the door handle cylinder, the backlash in the wrist joint and the rest of the arm allows it to deflect as the force increases. As mentioned above, this taking up of the backlash is in fact a desirable side effect because it prevents the hand from slipping off the end of the handle in subsequent steps.

Position control can also be advantageous when interacting with compliant objects in the environment. For example, when finding the handle, which is itself impedance-controlled in a sense (it moves back to its closed position with a force dictated by its return spring), the wrist is position controlled but the collision between the hand and the handle is still compliant. This is roughly equivalent to using position control to move the arm toward the door while impedance-controlling the fingers: in both situations, there is compliance during the collision which prevents the rapid rise in force characteristic of pure position control.
Chapter 5

Results

5.1 Speed

The total amount of time to open the door using the initial algorithm (only the door touch-off, without the reliability improvements) averaged 39.2s, with a small standard deviation of 0.40s primarily dependent on the distance from the arm’s home position to the handle position. This time starts with the arm at the home position and ends when the arm has finished the door-opening motion. By contrast, the mean time to open the door with the two additional touch-offs is 58.5s ± 1.32s, meaning that the two extra touch-offs (with their concomitant increases in reliability) only increase the amount of time to open the door by about 19.2 seconds, or 49%. These results can be seen in Table 5.1. Most of this time comes from the fact that the hand moves very slowly to avoid damaging itself or the door. Both of these times are much better than opening the door manually, which ranged from a potential minimum of a minute and twenty seconds in the fastest case to over 30 minutes without success at the DARPA Robotics Challenge. Times for this open-loop method are shown in Table 5.2. Overall, the new FFOD method is 33% faster in the worst case than the old
open-loop \emph{a priori} model manipulation method, with a much larger improvement in other cases.

Table 5.1: Runtimes for 13 trials of the FFOD method, before and after the reliability improvements.

<table>
<thead>
<tr>
<th>Constrain $x$ and $z$ (door and handle touch off)</th>
<th>Constrain $x$, $z$, and $y$ (door, handle, and cylinder touch off)</th>
</tr>
</thead>
<tbody>
<tr>
<td>39.28</td>
<td>56.86</td>
</tr>
<tr>
<td>39.80</td>
<td>56.45</td>
</tr>
<tr>
<td>38.79</td>
<td>58.68</td>
</tr>
<tr>
<td>39.37</td>
<td>59.33</td>
</tr>
<tr>
<td>39.26</td>
<td>59.50</td>
</tr>
<tr>
<td>38.72</td>
<td>59.88</td>
</tr>
</tbody>
</table>

Average: \textbf{39.20} \hspace{1cm} Average: \textbf{58.45}

Table 5.2: Test course results for the open-loop \emph{a priori} model manipulation method. Times with a plus sign were derived from video recordings of tests which had been edited to remove operator time; actual times are much higher than evidenced by the recording.

<table>
<thead>
<tr>
<th>Time</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:57</td>
<td>Success</td>
</tr>
<tr>
<td>4:06+</td>
<td>Success</td>
</tr>
<tr>
<td>1:20+</td>
<td>Success</td>
</tr>
<tr>
<td>4:08+</td>
<td>Fail (fell over)</td>
</tr>
<tr>
<td>31:35</td>
<td>Fail</td>
</tr>
</tbody>
</table>

### 5.2 Reliability

After initial testing and development of the guarded move force thresholds, data was recorded from eighteen trials over three robot locations. These three locations were at roughly a 20-degree angle with the door, a 45-degree and, and a 90-degree angle, as shown in Figure 5.1. The diagram shows both the relative location of the door handle, as well its orientation, represented by the approaching hand. Black x marks represent successful trials, and red
x marks represent unsuccessful trials. Overall there were 15 successes and three failures, for a total success rate of 83%. This is better than the 60% success rate exhibited by the open-loop \textit{a priori} model method.

Figure 5.1: The locations of the door handle and hand relative to the robot for each of the eighteen trials, with successes and total number of trials marked in each area. Green X marks represent range of motion limits for a 45-degree handle orientation.

The results are also summarized in Table 5.3. Two of the failures led to changes in the algorithm, and no failures were noted after the second of these changes. In the first failure (Note 1), the hand slipped off the handle as it pushed on the door, allowing the door to relatch and failing to open it. In the second failure (Note 2), the hand collided with the handle cylinder when moving downward. Reoccurrences of this failure were mitigated by starting laterally further away from the handle cylinder and adding a touch-off on the cylinder as depicted in Figure 4.10. The third failure (Note 3) occurred when the index finger caught on the end of the handle during the cylinder touch-off; this problem was worked around by leaving the index finger slightly extended, as shown in Figure 4.19. For the last five runs (Note 4), the transform from the robot to the door handle was recorded at the start of the runs but was not recaptured between runs. The original location is used in the table for all five
runs.

Table 5.3: Results. Eighteen trials are shown, from three different robot poses. Fifteen trials are successful, for a success rate of 83%.

<table>
<thead>
<tr>
<th>Robot-door angle (degrees)</th>
<th>x offset (cm)</th>
<th>y offset (cm)</th>
<th>Success?</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>43.8</td>
<td>0.2</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>46.0</td>
<td>-2.1</td>
<td>N</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>46.0</td>
<td>-1.8</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>46.2</td>
<td>-1.8</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>46.4</td>
<td>-2.1</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>42.1</td>
<td>7.7</td>
<td>N</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>40.5</td>
<td>2.1</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>37.5</td>
<td>4.7</td>
<td>N</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>37.3</td>
<td>4.7</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>53.4</td>
<td>-13.5</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>48.0</td>
<td>-10.2</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>53.2</td>
<td>-11.8</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>52.7</td>
<td>-11.4</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>46.6</td>
<td>1.1</td>
<td>Y</td>
<td>4</td>
</tr>
<tr>
<td>86</td>
<td>46.6</td>
<td>1.1</td>
<td>Y</td>
<td>4</td>
</tr>
<tr>
<td>86</td>
<td>46.6</td>
<td>1.1</td>
<td>Y</td>
<td>4</td>
</tr>
<tr>
<td>86</td>
<td>46.6</td>
<td>1.1</td>
<td>Y</td>
<td>4</td>
</tr>
</tbody>
</table>

Note 1. Hand slipped off handle.
Note 2. Hand hit handle cylinder when moving down.
Note 3. Index finger caught on handle end.
Note 4. Start location was not recaptured between runs.

5.3 Reachability

Figure 5.1 also shows two green x marks, which represent range of motion limits. The robot was moved closer and farther from the door at a 45-degree angle to see where it could no longer open the door. At the closer green x mark, the robot could not move its hand into the correct approach orientation without its arm colliding with its body. This range of
motion limit applies only to this particular handle pose (comprising both the position and orientation). If the handle were in the same location but with a different orientation, the hand could easily move and allow the arm to clear the body. This is reflected in the fact that the hand is in the yellow region of the reachability diagram, meaning that the shown position can be achieved with many different hand orientations.

At the furthest green x mark, the robot was able to turn the handle but could not extend the arm far enough to push the door open. Though it appears that this handle position is still far enough within the reachability map for the hand to be able to push it open, the height of the door handle means it is far below the vertical midplane of the reachability sphere. Figure 5.2 shows the reachability sphere compared to the height of the handle, and why a door near the edge of the arm’s reach could be unlatched but not opened.

Figure 5.2: The height of the door handle is shown by the plane drawn at about the robot’s elbow height. If the handle lies outside the reachability sphere, the robot will be unable to manipulate it.
Chapter 6

Conclusion

Presented here was a novel door-opening approach that uses force feedback to detect and localize the door and its handle in the course of opening it, allowing the robot to compensate for body movement and sensor error and reliably reach the same point on the door handle during each attempt. This approach was tested with the robot ESCHER, and was shown to be 33 percent faster in the worst case and tens of times as fast in the best case as the previous open-loop a priori model approach used by ESCHER. In testing it exhibited an 83% success rate, an improvement over the previous method’s 60%. The system was implemented in a platform-agnostic way that sits on top of the commonly-used MoveIt! framework, allowing it to be easily adapted to any robot that has force sensors and uses MoveIt! and ROS. The core of the system—the guarded move—also allows easy adaptation to other tasks, as it allows arbitrary moves with arbitrary guards.
6.1 Future work

One of the most interesting applications for this research is using the guarded move framework for other manipulation tasks, such as other tasks in the DARPA Robotics Challenge. For example, the same guarded moves could be used for to locate a valve in space, which could then be turned to open or close it.

One of the modifications to the algorithm was in pulling the hand back after touching off on the door. This prevents the fingers from catching on the lock rail or dragging on the door, which was increasing the wrist torque and could trick the arm into thinking it had found the door handle. An alternative approach could be to watch the finger force as well when moving down, and compensating for rises in the wrist torque if the fingers see torque. This may not work, however, if the fingers are jammed, in which case they would be under a large compressive force but the joints would be under comparatively little torque.

The system currently requires an operator to cue the robot to where the handle should be, by inserting a marker into the robot’s internal representation of the world. This could be handled autonomously, such as how Andreopoulos and Tsotsos use stereo vision to extract the handle location. Adding this functionality would be the most interesting improvement, because it would allow the robot to open a door entirely autonomously.

Non-arm limbs could also be controlled using the guarded move; for example, legs could be moved in a guarded fashion when climbing down a ladder. The robot could feel for the lower steps without being able to see them, allowing it to climb down a ladder without a highly accurate ladder model. The kinematic chain analysis could be extended to legs, and could also be used to design limbs rather than to analyze them after the fact.

The guarded move closes the loop around MoveIt! to allow more robust interaction with the environment. A further step could be a high-level error handler which closes the loop
around the whole door-opening approach, such as using vision or LIDAR to detect whether
the door has actually opened by the end of the algorithm or whether an unexpected object
(such as a person) has moved into the way. This high-level error handler would make this
approach even more robust.
Bibliography


[40] Daniel E. Whitney. “Force Feedback Control of Manipulator Fine Motions”. In: *Journal of Dynamic Systems, Measurement, and Control* 99.2 (June 1, 1977), pp. 91–97. ISSN: 0022-0434. DOI: [10.1115/1.3427095](http://dx.doi.org/10.1115/1.3427095) URL: [http://dx.doi.org/10.1115/1.3427095](http://dx.doi.org/10.1115/1.3427095) (visited on 03/21/2015).


