A Compositional Approach to Robotic Impedance Control

by

David Verdi

B.S., Columbia University (2017)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Signature redacted

Author .....
Department of Mechanical Engineering
May 14, 2019

Signature redacted

Certified by...
Neville Hogan
Sun Jae Professor of Mechanical Engineering
Thesis Supervisor

Signature redacted

Accepted by .....
Nicolas G. Hadjiconstantinou
Professor of Mechanical Engineering
Chairman, Committee on Graduate Students

A Compositional Approach to Robotic Impedance Control

by

David Verdi

Submitted to the Department of Mechanical Engineering
on May 14, 2019, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

While the gap between robot and human performance is rapidly closing, humans still vastly outperform robots at dynamic interaction tasks, particularly those which involve manipulation into kinematic singularities, and those which might involve collaborative or closed kinematic chain manipulation with multiple actuators. In this work, compositional impedance control, or linearly superimposing impedance controllers on a robot, is presented as a step towards closing this performance gap.

First, an overview of compositional impedance control is provided, along with a discussion of the control framework's applications to redundancy resolution, controlling closed kinematic chains, managing collaborative manipulation, and tackling high-DOF manipulation tasks. This control scheme was implemented on a Baxter Research Robot, and a series of system identification experiments were conducted to determine how well the robot was able to render the desired impedance parameters, and how well those parameters linearly superimposed between two arms collaboratively manipulating an object. Commanded static stiffness was found to be delivered by each individual arm to within a 2% error, while the linear superposition was verified to within 3% error. Commanded endpoint damping was found to be delivered by the robot with a 17% and 57% error by the left arm and right arm respectively. Linear superposition for damping was verified to within 7% error.

Using this compositional impedance control framework, sample manipulation tasks such as closed-chain manipulation into singularity, and high-speed closed-chain cloth manipulation (in the form of robotic shoeshining) were implemented. Finally, nullspace projection methods for redundant manipulators are discussed, and an impedance based implementation of the nullspace projection method is presented.

Thesis Supervisor: Neville Hogan
Title: Sun Jae Professor of Mechanical Engineering

Acknowledgments

First and foremost I would like to thank my thesis advisor, Professor Neville Hogan for mentoring me throughout my graduate career at MIT. His knowledge of multi-domain system dynamics, control theory, and human motion control is unparalleled, and inspiring. It is truly an honnor to work alongside him, and I couldn't have asked for a better advisor.

I would also like to thank the graduate students and postdocs in the Newman Lab, especially James Hermus, Jongwoo Lee, Meghan Huber, Davi da Silva, Joe Davidson, Moses Nah, Logan Leahy, and Michael West. The innumerable, and extensive conversations on everything from the minute details of implementing eigenvector placement in a state-space control problem, to broader questions on our life trajectories greatly enriched my graduate experience.

Additionally, I would like to thank my professors and teaching assistants for tirelessly holding me to the high standards that give MIT its reputation. My grasp and comfort with many topics in controls, system dynamics, and robotics has grown immensely because of it.

I also want to thank Professor Henry Hess at Columbia University for mentoring me for many years, and teaching me the importance of reasoning from first principles. I also want to thank Professor Kristin Myers at Columbia University, who helped me along the path to graduate school. I would also like to thank Michael Propper and Dr. Robert Muratore, both of whom sparked my interest in physics.

Finally, I want to thank my parents, my friends, and my family. I wouldn't have been able to do any of this without their endless support. I would also like to thank the fantastic communities at the MIT Sailing Pavilion, the MIT Coffeehouse Lounge (particularly on Saturday nights), MIT Hillel, MIT Chabad, and Harvard Hillel for making life in Cambridge so fun.

This research was performed in the Eric P. and Evelyn E. Newman Laboratory for Biomecahnics and Human Rehabilitation at the Massachusetts Institute of Tech-

nology. The research presented in Chapters 1 - 5 was supported by The National Science Foundation - National Robotics Initiative Grant No. 1637824. The research presented in Chapter 6 was supported by the Centers for Mechanical Engineering Research and Education at MIT and SUSTech, Project: Towards Whole-Body Physical Collaboration Between Robots and Humans.

Contents

1 Introduction **15**

1.1 Equivalent Networks and the Human Motion Controller 16

1.2 Impedance Control in Robots 20

1.2.1 Overview 20

1.2.2 Controller Derivation 21

1.2.3 Advantages 29

1.3 Overview of Thesis 31

2 Compositional Impedance Control **33**

2.1 The Linear Superposition Property 33

2.2 Redundancy Resolution 34

2.3 Closed Kinematic Chains 38

2.4 Collaborative Manipulation 39

2.5 Tackling the Scale Up Problem 41

2.5.1 Complex Tasks 41

2.5.2 High-DOF Systems 42

3 Implementation on a Baxter Research Robot **45**

3.1 Robot Hardware Overview 45

3.2 Robot Control Overview 48

3.3 High Level Impedance Controller Implementation Details 50

4 Verifying Impedance Composition and Controller Performance **53**

4.1 Overview and Methodology 53
4.2 Controller Settings 60
4.3 Static Stiffness Identification 61
4.4 Damping and Inertia Identification 62
4.4.1 Damping Parameter Fits 62
4.4.2 Inertial Parameter Fits 72

5 Sample Manipulation Tasks **75**
5.1 Manipulation Into A Singularity 75
5.2 Complex Task Proof of Concept: Robotic Shoe Shining 76

6 Nullspace Projections: An Impedance Based Interpretation **83**
6.1 The Traditional Approach 84
6.2 An Impedance Based Approach 91
6.3 Comparison of Projection Behavior at Singularity 94
6.3.1 Traditional Approach 95
6.3.2 Impedance Based Approach 95

7 Conclusions and Future Work **105**
7.1 Conclusions 105
7.2 Future Work 107
7.2.1 System Identification of Back-Driving Impedances 107
7.2.2 Tackling the Optimization Scale Up Issue 110
7.2.3 Impedance Based Nullspace Projection Behavior at Singularity 111

A Expected Endpoint Stiffness, Damping, and Inertia Matrices for System ID Experiments **113**
A.1 Net Endpoint Stiffnesses 113
A.2 Net Endpoint Damping 114
A.3 Net Endpoint Inertia 115

B Motions Along y and z Coordinates During Step Responses **117**

List of Figures

1-1 Thévenin and Norton Equivalent Circuits 17

1-2 Nonlinear Equivalent Network of the Human Acutator 19

1-3 1-DOF Robot 21

1-4 1-DOF Robot Under Impedance Control 22

1-5 1-DOF Robot Under Impedance Control With Damping to Ground . 23

1-6 4-DOF Planar Robot Operating in 3-DOF Planar Space 24

1-7 Behavior of a Planar Robot Under Impedance Control 26

2-1 Inertial Object With Nonlinear Impedance Elements 33

2-2 4-DOF Planar Robot 35

2-3 Superimposed Endpoint and Joint-Space Impedance Behavior 36

2-4 Closed Kinematic Chain Manipulator 38

2-5 Closed Kinematic Chain Manipulator Under Impedance Control . . . 39

2-6 Collaborative Manipulators 40

2-7 Collaborative Manipulators Under Impedance Control 40

3-1 The Baxter Robot Joint Naming Convention and Internals 46

3-2 The Baxter Robot Dimensions and Coordinate System 47

3-3 Baxter Robot Torque Control Implementation 49

4-1 System ID Initial Pose 55

4-2 Aluminum Linkage Drawing 55

4-3 Baxter Robot With Linkage Installed 56

4-4 500 N/m Force vs. Displacement Stiffness Fits 63

4-5 400 N/m Force vs. Displacement Stiffness Fits 64
4-6 400 N/m Force vs. Displacement Stiffness Fits 65
4-7 500 N/m Step Responses 67
4-8 400 N/m Step Responses 68
4-9 300 N/m Step Responses 69

5-1 Singularity Manipulation Task 76
5-2 Singularity Manipulation Trajectories 77
5-3 Jacobian Condition Numbers During Singularity Manipulation 77
5-4 Robotic Shoe Shining Task 79

6-1 Four Link Planar Robot Approaching Singularity 88
6-2 Jacobian Condition Number Approaching Singularity 89
6-3 Jacobian Pseudoinverse Norm Approaching Singularity 89
6-4 Three Link Planar Robot Approaching Singularity 95
6-5 Nullspace Projected Torques From the Traditional Method 96
6-6 Nullspace Projected Torques From the Traditional Method, With Adjusted Zero Threshold 96
6-7 Eigenvalues of $\boldsymbol{K}_\theta$ 97
6-8 Nullspace Torques From Impedance Method Without λ_1 Replacement 98
6-9 Nullspace Torques From Impedance Method With λ_1 Replacement At Singularity 99
6-10 Nullspace Projected Torques From the Impedance Based Method, With Adjusted Eigenvalue Zero Threshold 99
6-11 Eigenvector Contributions to Joint Torques 102
6-12 Eliminating Discontinuity by Ramping Up k_2 103

7-1 Two 1-DOF Robots 108
7-2 Two 1-DOF Robots Clamped Together 108
7-3 Two 1-DOF Robots Interacting 108

B-1 500 N/m Step Response y and z Displacements 118

B-2 400 N/m Step Response y and z Displacements 119
B-3 300 N/m Step Response y and z Displacements 120

List of Tables

4.1 Nominal Endpoint Coordinates for System ID 54

4.2 Nominal Joint Angles for System ID 54

4.3 Results From Static Stiffness Identification Experiments 66

4.4 Results From Damping Identification Experiments 70

4.5 Results From Inertial Parameter Fits 71

Chapter 1

Introduction

Robot performance has seen tremendous improvements over the past several decades. Nevertheless, humans still vastly outperform robots in tasks that involve complex dynamic interactions with the environment [Hogan, 2017, Krotkov et al., 2018]. For instance, a woodcarver might take a delicate workpiece with complex geometric features and proceed to scrub or sand its intricate contours. A painter might extend her arm to the edge of her dexterous workspace to carefully apply spackle to a difficult to reach location. A shoeshiner might stretch out a cloth between his hands and use it to rapidly buff a leather shoe.

In general, humans seamlessly transition from free to constrained motions, and can readily perform dynamic interaction tasks into and out of kinematic singularities. They easily manipulate a variety of compliant and non-compliant objects with two hands, or even coordinate large object manipulation with several other people. All of these tasks, while performed by humans with relative ease, can be challenging to program on a robot.

1.1 Equivalent Networks and the Human Motion Controller

One source of inspiration for properly managing a robot's dynamic interactions with its environment is our current understanding of the human motion controller. The actual human neuromuscular physiology and control scheme is, of course, exceedingly complex, and our ability to study its internal details are currently quite limited [Kandel et al., 2013]. One approach to gaining insight on the net interactive behavior of this "black box" is to borrow the idea of an equivalent network from circuit theory [Hogan, 2014, Hogan, 2017].

Given a complex circuit with an arbitrary number of interconnected sources and dynamic elements (such as resistors, inductors, capacitors, voltage sources, and current sources), we can exactly model the output port behavior of the circuit with a far simpler Norton or Thévinin equivalent circuit, as depicted in Figure 1-1 [Horowitz and Hill, 2015]. In the case of a Norton equivalent, the complex circuit is replaced by a controlled current source and parallel impedance, while in the case of a Thévenin equivalent, the complex circuit is replaced by a controlled voltage source and series impedance.

In the electrical domain, the impedance of an element is defined as the functional relationship between the input current through the element (flow variable), and the output voltage across the element (effort variable):

$$V = Z\{I\} \tag{1.1}$$

For instance, the constitutive equation of a resistor could be written as $V = RI$, that of a capacitor as $V = \frac{1}{C}\int Idt$, and that of an inductor as $V = L\frac{dI}{dt}$. In the special case of a linear element, the impedance relationship can be expressed as a transfer function in the Laplace domain:

$$Z(s) = \frac{V(s)}{I(s)} \tag{1.2}$$

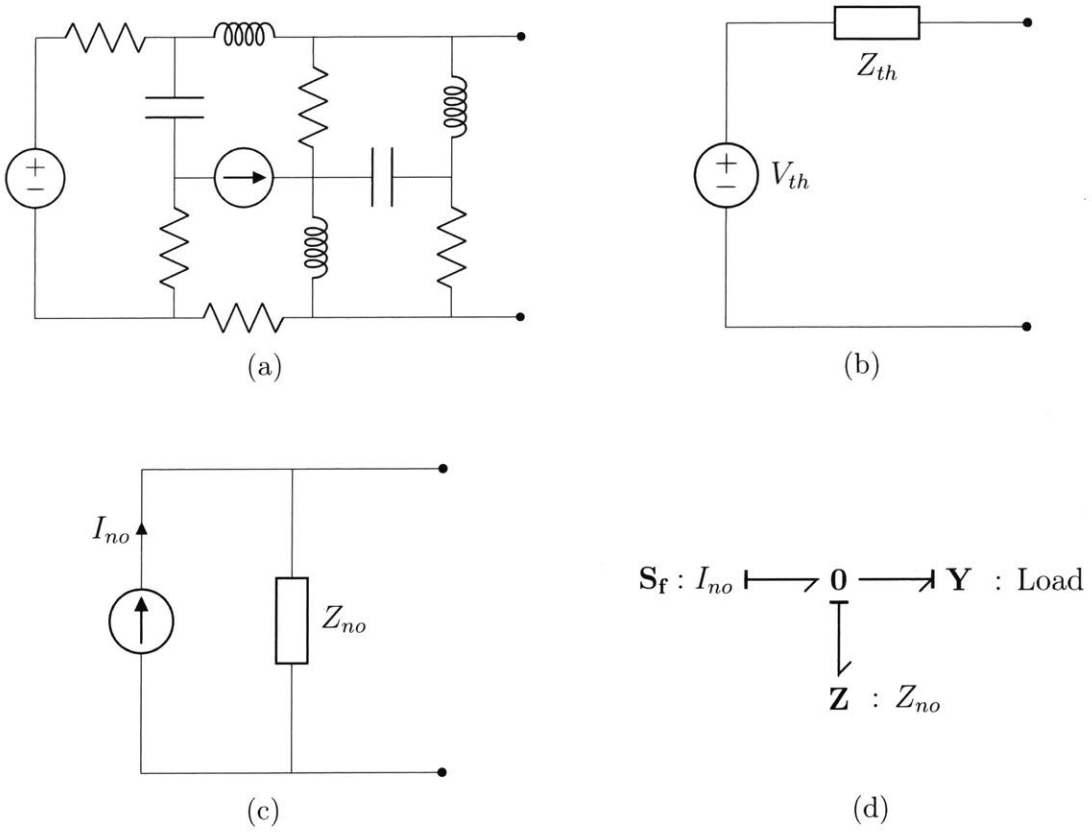


Figure 1-1: (a) A complex circuit with many interconnected dynamic elements and sources. (b) A Thévinin equivalent circuit with a voltage (effort) source and series impedance. (c) A Norton equivalent circuit with a current (flow) source and parallel impedance. (d) A bond graph representation of the Norton equivalent circuit, connected to a load modeled as an admittance.

where s is the Laplace variable.

There exists a one-to-one mathematical analogy between the modeling of electrical system dynamics and mechanical system dynamics [Brown, 2001, Hogan and Breedveld, 2002]. The effort variable (voltage) becomes force, and the flow variable (current) becomes velocity. Similarly, resistance is analogous to viscous damping, capacitance is analogous to stiffness, and inductance is analogous to inertia, with all constitutive equations taking on an identical form. In the mechanical domain the impedance can be written as:

$$f = Z\{\dot{x}\} \tag{1.3}$$

However, the usual convention for mechanical systems is to write the impedance operator in terms of x rather than $\dot{x}$. The encoded information is the same, but more potentially problematic differentiations may be required in the operator [Brown, 2001]:

$$f = Z\{x\} \tag{1.4}$$

Finally, for a linear mechanical system, we can write:

$$Z(s) = \frac{F(s)}{X(s)} \tag{1.5}$$

We can thus extend the ideas of Norton and Thèvinin equivalent networks to the mechanical domain, which includes the human musculoskeletal system [Hogan, 2014, Hogan, 2017].

A block diagram of a Norton[1] equivalent network for a human limb is shown in Figure 1-2. In addition to being competent enough to describe the full interactive dynamics of the limb, this model reflects key insights that we have gained about upper limb motion control. Many findings suggest that the human motion controller specifies a nominal limb position (or trajectory) by setting the relative activation level of opposing antagonist muscle pairs [Feldman, 1966, Bizzi et al., 1982, Bizzi et al.,

[1]See [Hogan, 2017] and [Hogan, 2014] for a full discussion of why the Norton network, and not the Thèvinin network, is the appropriate choice here. In brief, the Norton network is translation invariant (independent of our chosen coordinate frame), and its equivalent source is unambiguously identifiable.

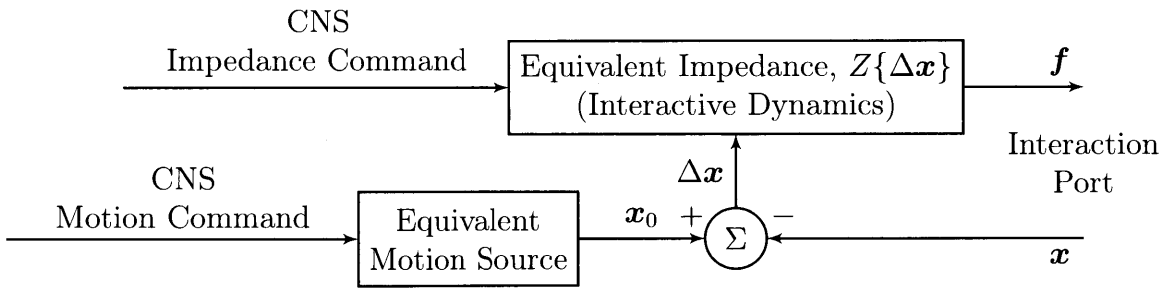


Figure 1-2: A block diagram of a Norton equivalent network of the human motion control system. Here, the central nervous system provides a virtual trajectory $\boldsymbol{x}_0$ and an effective hand impedance, $Z\{\cdot\}$. The difference between the virtual and actual hand position, $\Delta \boldsymbol{x}$, along with the effective hand impedance determines the contact force that will be felt at the interaction port. Figure adapted from [Hogan, 2014].

1984]. For instance, the nominal elbow joint angle is moved and set by changing the ratio of activation between the elbow flexors (brachialis, biceps brachii, and the brachioradialis) relative to the elbow extensors (triceps brachii and anconeus) [Netter, 2014]. This trajectory, denoted as $\boldsymbol{x}_0$ in Figure 1-2, is known as the reference trajectory, zero-force trajectory or virtual trajectory[2], as it is the (potentially non-physically realizable) trajectory along which the hand would move in the absence of external forces or constraints. As long as the impedance operator is invertable, the virtual trajectory exists, regardless of whether the hand is performing free motion tasks, or constrained motion (contact) tasks [Doeringer, 1999, Hermus, 2018].

Another important aspect of human motion control that the model captures is the equivalent impedance at the hand, which is also set by commands from the central nervous system. Hand stiffness is adjusted by the central nervous system by either co-activating opposing antagonist muscle pairs [Humphrey and Reed, 1983, Hogan, 1984], or by modulating reflex gains in spinal cord feedback loops [Nichols and Houk, 1976, Hoffer and Andreassen, 1981]. While the intrinsic mass of the adult musculoskeletal system is fixed, the effective inertia felt in each direction at the hand is heavily dependent on the configuration of the arm [Hogan, 1990]. The combined impact of all of these factors determines the effective arm impedance, $Z\{\cdot\}$. This effective impedance is what determines the interaction force felt when the arm is deflected from

[2]While these terms can denote different concepts in human motion control research, they are used interchangeably here in the context of robot control.

its virtual trajectory in a constrained motion task, or in response to a disturbance [Hogan, 2017]. For humans to successfully perform complex tasks, controlling this effective arm impedance is key [Burdet et al., 2001, Franklin et al., 2007, Schabowsky et al., 2007, Damm and McIntyre, 2008, Selen et al., 2009].

1.2 Impedance Control in Robots

1.2.1 Overview

Traditionally, many robot control frameworks involve tightly controlling a single set of manipulator variables [Spong and Vidyasagar, 1989]. These variables might be joint positions in the case of position control, endpoint forces in the case of force control, or a combination of endpoint positions and forces in orthogonal directions in the case of hybrid position/force control [Craig and Raibert, 1979, Raibert and Craig, 1981, Mason, 1981, Ortenzi et al., 2017]. These methods work perfectly well for many robot applications. Position control, for instance, is widely used for free space tasks such as pick-and-place or spray painting operations [Gasparetto et al., 2012], while force control is often used for interaction tasks, such as robotic deburring in structured industrial environments [Stepien et al., 1987].

These traditional control algorithms are, however, not very versatile. For instance, force control is only usable in situations when the manipulator is guaranteed to be in contact with the environment, otherwise the manipulator will rapidly accelerate to the edge of its workspace. Even in situations where contact is guaranteed, the force control algorithm might still suffer from stability issues [Whitney, 1977, Colgate and Hogan, 1989a]. Position control, by contrast, is only usable in free space, since contact may cause damage to either the robot, or objects in the environment. It is possible to switch between position control and force control via a finite state machine when contact is detected, but this requires very accurate robot perception algorithms, and robust methods to deal with event detection uncertainties [Salehian et al., 2018, Atkeson et al., 2018].

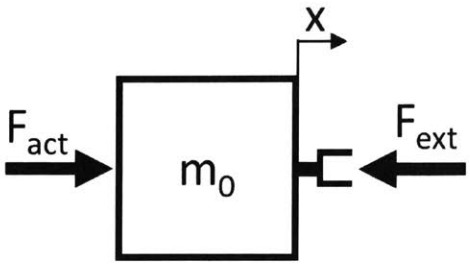


Figure 1-3: 1-DOF manipulator with mass m_0, actuation force F_{act}, and external force F_{ext}. All displacements and forces pointing to the right will be considered positive.

An alternative approach to robot control that is suitable for both free-space and contact tasks might look like our equivalent network model of the human motion control system in Figure 1-2. In this paradigm, we forego rigidly controlling a single set of manpulator variables such as endpoint forces or positions. Instead, we shape the relationship between input flow variables (e.g. manipulator endpoint position or velocity relative to a reference trajectory) and output effort variables (e.g. net endpoint forces and torques) at an interaction point with the environment. By doing this, we are essentially shaping the effective impedance of the robot endpoint. This control technique was introduced by Hogan, and is called known as impedance control [Hogan, 1985a, Hogan, 1985b, Hogan, 1985c].

1.2.2 Controller Derivation

1-DOF Case

To elucidate the idea of impedance control, we can consider the case of a 1-DOF actuated manipulator, with mass m_0, actuation force F_{act}, and an external force F_{ext}, which represents either a directly applied disturbance force, or the interaction force that arises over the course of a contact or manipulation task. Such a manipulator is shown in Figure 1-3.

One simple impedance control implementation is to give a manipulator the apparent behavior of a mass-spring damper system (second order dynamics). The behavior

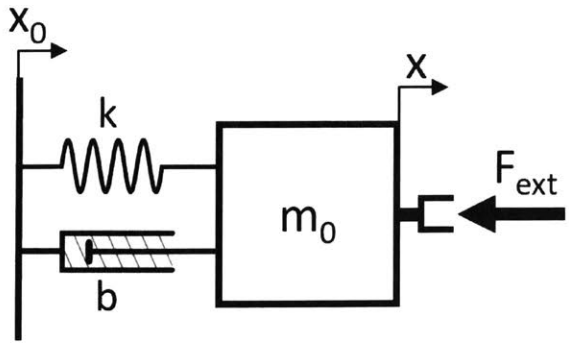


Figure 1-4: A 1-DOF manipulator under an example simple second order impedance controller. All displacements and forces pointing to the right will be considered positive.

of this controller is visualized in in Figure 1-4. To implement this, we can use the control law:

$$F_{act} = k(x_0 - x) + b(\dot{x}_0 - \dot{x}) \tag{1.6}$$

where x_0 is a reference (virtual) trajectory, k is the spring stiffness, and b is the viscous damping. This controller gives us the desired behavior in Figure 1-4. The disturbance response of the manipulator under this control law is:

$$Z = \frac{F_{ext}}{X} = m_0 s^2 + bs + k \tag{1.7}$$

where s is the Laplace variable. This is the effective endpoint impedance that characterizes the interactive behavior of the robot. The forward path dynamics of this controller are as follows:

$$\frac{X}{X_0} = \frac{bs + k}{m_0 s^2 + bs + k} \tag{1.8}$$

The forward path dynamics of this controller has a dynamic zero at $s = -\frac{k}{b}$ which comes as a consequence of defining the damping relative to the reference trajectory, x_0. One important ramification of defining the damper this way is that in order for our control law (Eqn. 1.6) to be implemented, x_0 must be differentiable. If we wish to maintain the system's desired interactive impedance (Eqn. 1.7), while also allowing us to use non-differentiable reference trajectories, we can define the damper relative to mechanical ground, as in Figure 1-5. This is particularly useful for system

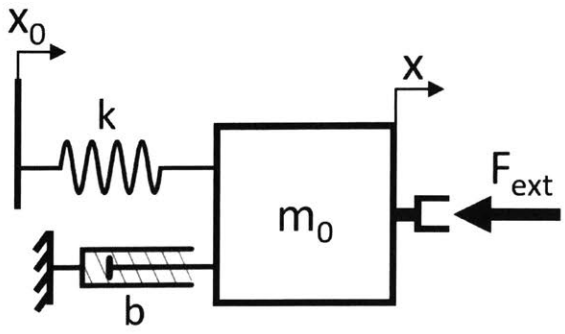


Figure 1-5: A 1-DOF manipulator under a simple second order impedance controller with damping defined relative to a fixed ground point. All displacements and forces pointing to the right will be considered positive.

identification purposes, where a non-continuous and non-differentiable step function in x_0 can be used. This controller is implemented as:

$$F_{act} = k(x_0 - x) - b(\dot{x}) \tag{1.9}$$

This controller has the forward path transfer function:

$$\frac{X}{X_0} = \frac{k}{m_0 s^2 + bs + k} \tag{1.10}$$

All of the above control laws do not attempt to modify the effective manipulator mass, m_0. If we wish to do so, we can write a more advanced controller to replace our given m_0 with a desired m_d [Hogan, 1985b, Hogan, 1987]. Assuming we have a measurement of the external interaction force F_{ext}, we can implement the control law:

$$F_{act} = \frac{m_0}{m_d}\big[k(x_0 - x) + b(\dot{x}_0 - \dot{x})\big] + \big[\frac{m_0}{m_d} - 1\big]F_{ext} \tag{1.11}$$

which gives us the behavior shown in Figure 1-4, but with an arbitrary mass m_d. Replacing $\dot{x}_0$ with zero will yield behavior analogous to that of Figure 1-5. One caveat to this approach is that incorporating force feedback into the control loop for apparent mass modification might be a source of instability, depending on the dynamics between the robot, sensors, and manipulated objects [Colgate and Hogan,

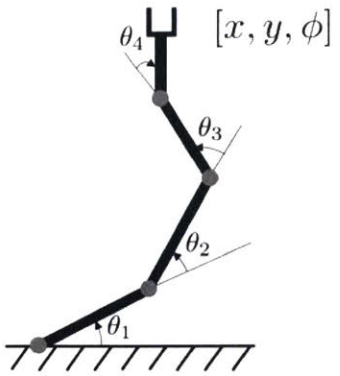


Figure 1-6: A 4-DOF manipulator operating in 3-DOF planar space. The joint-space configuration variables are $\boldsymbol{q} = \begin{bmatrix}\theta_1 & \theta_2 & \theta_3 & \theta_4\end{bmatrix}^T$, while the endpoint spatial coordinates are $\boldsymbol{x} = \begin{bmatrix}x & y & \phi\end{bmatrix}^T$, where x and y represent Cartesian position, and ϕ represents endpoint orientation.

1989a, Colgate and Hogan, 1989b, Newman, 1992].

Multi-DOF Case

We can extend the above analysis to a multi-DOF robot. Consider an n-DOF robot operating in m-DOF space. The configuration of this robot is given by a vector of generalized joint angles/positions $\boldsymbol{q} \in \mathbb{R}^n$, and the spatial coordinates of the robot endpoint are given by a vector $\boldsymbol{x} \in \mathbb{R}^m$. These coordinates are referred to as either endpoint coordinates, or operational space coordinates. For robots operating in the real world, $m = 6$. If a robot has more degrees of freedom in its joint-space relative to its endpoint-space ($n > m$), the robot is known as a redundant manipulator. An example of a 4-DOF planar robot operating in 3-DOF planar space is given in Figure 1-6.

For a robot, the forward kinematics represents the unique mapping from joint space coordinates $\boldsymbol{q}$ to endpoint coordinates $\boldsymbol{x}$ [Spong and Vidyasagar, 1989]:

$$\boldsymbol{x} = L(\boldsymbol{q}) \tag{1.12}$$

This relationship is not, in general, uniquely invertible [Pieper, 1968]. The velocity kinematics of a robot is the mapping of joint space velocities $\dot{\boldsymbol{q}}$ to endpoint velocities

$\dot{\boldsymbol{x}}$ [Spong and Vidyasagar, 1989]:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{1.13}$$

where $\boldsymbol{J} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{q}} \in \mathbb{R}^{m \times n}$ is the robot Jacobian matrix. The entries of this matrix are usually nonlinear functions of $\boldsymbol{q}$.

From the principle of virtual work, we obtain the mapping between joint torques and endpoint forces[3] as follows [Spong and Vidyasagar, 1989]:

$$\boldsymbol{\tau} = \boldsymbol{J}^T \boldsymbol{f} \tag{1.14}$$

where $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of generalized joint torques/forces, and $\boldsymbol{f} \in \mathbb{R}^m$ is the vector of generalized endpoint forces/torques.

The natural dynamics of a robot (neglecting friction and other possible non-conservative forces) are given in joint-space by the manipulator equation [Spong and Vidyasagar, 1989]:

$$\boldsymbol{M}_q \ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g} = \boldsymbol{\tau}_{act} + \boldsymbol{J}^T \boldsymbol{f}_{ext} \tag{1.15}$$

Here, $\boldsymbol{M}_q \in \mathbb{R}^{n \times n}$ is the robot joint-space inertia matrix, whose entries are usually nonlinear functions of $\boldsymbol{q}$. $\boldsymbol{C} \in \mathbb{R}^{n \times n}$ is the Coriolis matrix, whose entries are usually nonlinear functions of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. $\boldsymbol{g} \in \mathbb{R}^n$ is the vector of gravitational torques, whose entries are usually nonlinear functions of $\boldsymbol{q}$. $\boldsymbol{\tau}_{act} \in \mathbb{R}^n$ is the vector of joint actuator torques, and $\boldsymbol{f}_{ext} \in \mathbb{R}^m$ is the vector of external/interaction forces applied on the robot by the environment.

In our example system in Figure 1-6, we wish to implement a simple second order impedance controller such that the system will behave as a mass-spring-damper system (as in Figure 1-4), but along each of the coordinates in $\boldsymbol{x}$. This desired behavior is illustrated in Figure 1-7. In essence, we wish to replace the natural

[3] As is standard in the robotics literature, the term "joint torques" is a generalized term used to refer to both forces and torques in the joint space (torques about revolute joints, and forces on prismatic joints). Similarly, the term "endpoint forces" also refers to forces in linear coordinates, and torques along rotational coordinates. The 6×1 vector of generalized endpoint forces in robots operating in the real world is sometimes referred to as the "endpoint wrench"

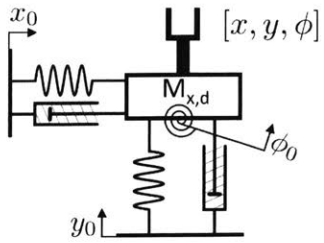


Figure 1-7: The desired behavior of a planar manipulator under impedance control. The system behaves like a mass-spring-damper system along each coordinate in $\boldsymbol{x}$.

dynamics of the robot (Eqn. 1.15) with:

$$\boldsymbol{M}_{x,d}\ddot{\boldsymbol{x}} + \boldsymbol{K}_x(\boldsymbol{x} - \boldsymbol{x_0}) + \boldsymbol{B}_x(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_0) = \boldsymbol{f}_{ext} \tag{1.16}$$

Here, $\boldsymbol{M}_{x,d} \in \mathbb{R}^{m\times m}$ is the desired endpoint-space inertia matrix, $\boldsymbol{K}_x \in \mathbb{R}^{m\times m}$ is the desired endpoint-space stiffness matrix, and $\boldsymbol{B}_x \in \mathbb{R}^{m\times m}$ is the desired endpoint viscous damping matrix. Usually, we would like the mass-spring-damper systems along each coordinate in $\boldsymbol{x}$ to be decoupled from one another. In this case, the matrices $\boldsymbol{M}_{x,d}$, $\boldsymbol{K}_x$, and $\boldsymbol{B}_x$ would all be diagonal, and the entries would represent the inertia, stiffness value, and damping value along each direction in $\boldsymbol{x}$. There are times, however, where off-diagonal couplings between endpoint coordinates may be desirable [Hogan, 1985b].

[Hogan, 1987] provides a control law to replace the dynamics in Eqn. 1.15 with those of Eqn. 1.16. First, using the force-torque mapping in Eqn. 1.14, we can transform the actuator torques from joint space torques into endpoint forces. Eqn. 1.15 becomes:

$$\boldsymbol{M}_q\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g} = \boldsymbol{J}^T(\boldsymbol{f}_{act} + \boldsymbol{f}_{ext}) \tag{1.17}$$

Since $\boldsymbol{M}_q$ is a mass matrix, it is guaranteed to be positive definite and therefore invertible. Thus, we can solve for $\ddot{\boldsymbol{q}}$:

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}_q^{-1}\left[\boldsymbol{J}^T(\boldsymbol{f}_{act} + \boldsymbol{f}_{ext}) - \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{g}\right] \tag{1.18}$$

Recalling our velocity kinematics:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{1.19}$$

If we differentiate this relationship with respect to time, we have:

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} \tag{1.20}$$

Substituting Eqn. 1.20 into Eqn. 1.18 and rearranging yields:

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{M}_q^{-1}\boldsymbol{J}^T(\boldsymbol{f}_{act} + \boldsymbol{f}_{ext}) - \boldsymbol{J}\boldsymbol{M}_q^{-1}\big[\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g}\big] + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} \tag{1.21}$$

Here, we can touch upon the relationship between the joint-space inertia matrix of a manipulator $\boldsymbol{M}_q$, and the endpoint-space inertia matrix of a manipulator $\boldsymbol{M}_x$ [Hogan, 1987, Khatib, 1987, Khatib, 1995]:

$$\boldsymbol{M}_x^{-1} = \boldsymbol{J}\boldsymbol{M}_q^{-1}\boldsymbol{J}^T \tag{1.22}$$

Substituting into Eqn. 1.21:

$$\ddot{\boldsymbol{x}} = \boldsymbol{M}_x^{-1}(\boldsymbol{f}_{act} + \boldsymbol{f}_{ext}) - \boldsymbol{J}\boldsymbol{M}_q^{-1}\big[\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g}\big] + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} \tag{1.23}$$

Now, we solve for $\boldsymbol{f}_{act}$:

$$\boldsymbol{f}_{act} = \boldsymbol{M}_x\big[\ddot{\boldsymbol{x}} + \boldsymbol{J}\boldsymbol{M}_q^{-1}[\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g}] - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}\big] - \boldsymbol{f}_{ext} \tag{1.24}$$

This equation gives us the actuator forces that must be applied to the robot in order realize any desired trajectory, $\ddot{\boldsymbol{x}}$, with all modeled robot dynamics compensated for. If we recall, our desired manipulator behavior under impedance control was given by Eqn. 1.16. Writing this in terms of a desired $\ddot{\boldsymbol{x}}$:

$$\ddot{\boldsymbol{x}}_d = \boldsymbol{M}_{x,d}^{-1}\big[\boldsymbol{f}_{ext} + \boldsymbol{K}_x(\boldsymbol{x_0} - \boldsymbol{x}) + \boldsymbol{B}_x(\dot{\boldsymbol{x}}_0 - \dot{\boldsymbol{x}})\big] \tag{1.25}$$

We now substitute $\ddot{\boldsymbol{x}}_d$ from Eqn. 1.25 into $\ddot{\boldsymbol{x}}$ from Eqn. 1.24:

$$\begin{aligned}\boldsymbol{f}_{act} = \boldsymbol{M}_x\boldsymbol{M}_{x,d}^{-1}\big[\boldsymbol{K}_x(\boldsymbol{x_0}-\boldsymbol{x})+\boldsymbol{B}_x(\dot{\boldsymbol{x}}_0-\dot{\boldsymbol{x}})\big] \\ +\boldsymbol{M}_x\big[\boldsymbol{J}\boldsymbol{M}_q^{-1}[\boldsymbol{C}\dot{\boldsymbol{q}}+\boldsymbol{g}]-\dot{\boldsymbol{J}}\dot{\boldsymbol{q}}\big] \\ +\big[\boldsymbol{M}_x\boldsymbol{M}_{x,d}^{-1}-\boldsymbol{I}\big]\boldsymbol{f}_{ext}\end{aligned} \tag{1.26}$$

where $\boldsymbol{I}$ is the identity matrix. We can multiply this equation by $\boldsymbol{J}^T$ to transform $\boldsymbol{f}_{act}$ back to actuator joint-space torques to yield the control law:

$$\begin{aligned}\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\boldsymbol{M}_x\boldsymbol{M}_{x,d}^{-1}\big[\boldsymbol{K}_x(\boldsymbol{x_0}-\boldsymbol{x})+\boldsymbol{B}_x(\dot{\boldsymbol{x}}_0-\dot{\boldsymbol{x}})\big] \\ +\boldsymbol{J}^T\boldsymbol{M}_x\big[\boldsymbol{J}\boldsymbol{M}_q^{-1}[\boldsymbol{C}\dot{\boldsymbol{q}}+\boldsymbol{g}]-\dot{\boldsymbol{J}}\dot{\boldsymbol{q}}\big] \\ +\boldsymbol{J}^T\big[\boldsymbol{M}_x\boldsymbol{M}_{x,d}^{-1}-\boldsymbol{I}\big]\boldsymbol{f}_{ext}\end{aligned} \tag{1.27}$$

This is the multi-DOF equivalent of Eqn. 1.11, with the same stability caveats of incorporating force feedback into the control loop [Colgate and Hogan, 1989a, Colgate and Hogan, 1989b, Newman, 1992].

We can apply several simplifications to this controller. First, if gravity compensation is included in a lower level of the robot control stack, we can neglect the gravitational torque vector ($\boldsymbol{g} = 0$). Next, if the manipulator is moving slowly, we can neglect the Coriolis term ($\boldsymbol{C}\dot{\boldsymbol{q}} = 0$) and the Jacobian derivative term ($\dot{\boldsymbol{J}}\dot{\boldsymbol{q}} = 0$). This eliminates the entire second line of Eqn. 1.27. The forces arising from these terms will now be treated as external disturbances to the controlled system.

If the user does not wish to modify the robot's endpoint inertia matrix, we can set $\boldsymbol{M}_{x,d} = \boldsymbol{M}_x$. This sets the term $\boldsymbol{M}_x\boldsymbol{M}_{x,d}^{-1} = \boldsymbol{I}$, which simplifies the first line, and eliminates the third line of the equation, along with the need to incorporate force feedback into the loop. This will be the approach taken for the remainder of this work.

Our simplified controller thus becomes:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\big[\boldsymbol{K}_x(\boldsymbol{x_0}-\boldsymbol{x})+\boldsymbol{B}_x(\dot{\boldsymbol{x}}_0-\dot{\boldsymbol{x}})\big] \tag{1.28}$$

In a robot, we usually have accurate measurements of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. Through the forward kinematics and velocity kinematics, our knowledge of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ gives us accurate measurements of $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ for incorporation into the controller:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\left[\boldsymbol{K}_x(\boldsymbol{x_0} - L(\boldsymbol{q})) + \boldsymbol{B}_x(\dot{\boldsymbol{x}}_0 - \boldsymbol{J}\dot{\boldsymbol{q}})\right] \tag{1.29}$$

This control law effectively "programs in" the impedance form of the constitutive equations of an operational space spring-damper system at the endpoint of the robot. The impedance control framework is, however, more general than this. We can implement any operational space equivalent dynamic behavior at the endpoint of the robot, as long as it can be described in the form of Eqn. 1.4 [Hogan, 1985c]. The torque control law would look like:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\left[Z\{\boldsymbol{x}, \boldsymbol{x_0}, ...\}\right] \tag{1.30}$$

where Z can depend on an arbitrary number of system states or fixed parameters. We are of course, always free to add in inertial and Coriolis force compensation as in Eqn 1.27. One example application of this might be obstacle avoidance in the environment. We can define a three-dimensional nonlinear impedance field in the environment, with a unilateral non-linear repulsive potential (e.g. non-linear one-way spring force and state-dependent damping) being defined in the immediate vicinity of perceived objects in the robot's work space [Hogan, 1985c, Khatib, 1986, Newman and Hogan, 1987].

1.2.3 Advantages

The control law in Eqn. 1.29 has many favorable attributes from a robot control perspective. For instance, it allows us to control the endpoint trajectory of the robot by directly specifying $\boldsymbol{x}_0$, without the use of any inverse kinematics. This is a highly desirable quality since for many robotic manipulators, the inverse kinematics are difficult to compute in closed form (and thereby require a nonlinear optimization

approach, which is not guaranteed to converge), and are non-unique (in the case of a redundant manipulator, there are infinitely many solutions for the inverse kinematics) [Goodwine, 2004].

Many operational space controllers heavily depend on computing the inverse of the Jacobian matrix inside the control loop [Khatib, 1987, Raibert and Craig, 1981, Miomir et al., 2008]. This becomes problematic when the manipulator is commanded into a singular configuration (i.e. at the edge of the manipulator workspace), since the Jacobian rapidly loses rank and becomes ill-conditioned near singularities. This method, however, does not depend on any Jacobian inverse operations, and therefore is able to seamlessly venture into and out of singular configurations without any numerical instabilities.

Another advantage of this control law is that it is suitable for both manipulation tasks in free space, and in contact with the environment [Hogan, 1985c, Hogan, 1987]. This is in contrast to the aforementioned position control and force/hybrid control paradigms. In free space, the robot behaves as if the endpoint was under a compliant position controller, with inertial lag (i.e. $\boldsymbol{x}_0$ will tend to lead $\boldsymbol{x}$ due to the inertia of the robot). When the manipulator is brought into contact with the environment (i.e. when the virtual trajectory $\boldsymbol{x}_0$ dips below the surface of a rigid object in the environment) the robot simply comes into contact with the object and applies a contact force. The magnitude and direction of the applied contact force can be controlled by modulating the $\boldsymbol{K}_x$ gain, or by adjusting how deeply $\boldsymbol{x}_0$ penetrates into the object.

One final advantage of the impedance control framework is the ability to linearly superimpose a number of arbitrary non-linear impedance controllers onto a single robot manipulator [Hogan, 1985c]. This allows us to compose a number of different impedance controllers on a robot to achieve tasks such as redundancy resolution, closed kinematic chain manipulation, collaborative manipulation, and to tackle complex tasks involving many degrees of freedom. The compositionality property of impedance controllers will be the primary focus for the remainder of this work.

1.3 Overview of Thesis

This introductory chapter has provided the foundation for understanding robot impedance control in terms of its biologically inspired origins, implementation details, and advantages over more traditional robot control approaches.

Chapter 2 delves into the compositionality property of impedance controllers. Applications to redundancy resolution, closed kinematic chains, collaborative manipulation, and scaling up to many degrees of freedom are discussed.

Chapter 3 discusses the particulars of implementing the impedance controller on a Baxter Research Robot, including high and low level control loop details, and the quaternion representation of rotational stiffnesses for numerical stability.

Chapter 4 discusses system identification performed on the Baxter Research Robot to verify how well multiple impedances linearly superimpose in practice, as well as to evaluate how well a commanded endpoint stiffness and damping can be realized at the robot endpoint.

Chapter 5 examines proof of concept tasks such as manipulating the robot into and out of singularities under impedance control, and also the application of the compositional impedance control method to a complex task, such as robotic shoe shining.

Chapter 6 discusses traditional nullspace projection techniques for controlling manipulator redundancies (such that the redundancy resolution criteria does not impact the operational space task), as well as a new approach to generating a nullspace projection operator, based on the kernel of the operational-space stiffness matrix reflected into joint-space.

Finally, Chapter 7 discusses the conclusions of the present work, along with future directions for further inquiry on the subject at hand.

Chapter 2

Compositional Impedance Control

2.1 The Linear Superposition Property

One important property of impedance controllers is the ability to linearly superimpose them in a robot [Hogan, 1985c]. To elucidate this idea, we consider an inertial object of mass m. Let us first attach a nonlinear spring, referenced to a virtual trajectory x_1, with the constitutive equation $f_1 = Z_1\{x, x_1\} = kx^2(x_1 - x)^3$. Next, let us attach a nonlinear damping element, referenced to a virtual trajectory x_2, with the constitutive equation $f_2 = Z_2\{x, x_2\} = b\,\mathrm{sgn}(\dot{x}_2 - \dot{x})(\dot{x}_2 - \dot{x})^2$. Finally, let us attach a general nonlinear dynamic element with the constitutive equation $f_i = Z_i\{x, x_i, ...\}$. This arrangement is visualized in Figure 2-1.

For N arbitrary nonlinear dynamic elements attached to the mass, Newton's sec-

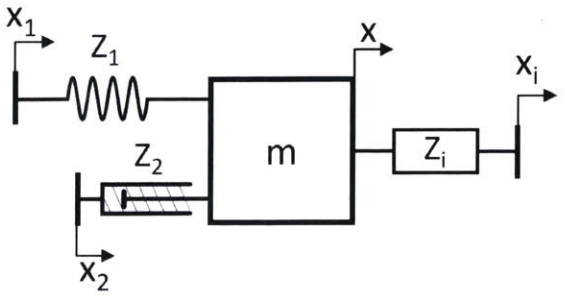


Figure 2-1: An inertial object of mass m with nonlinear impedance elements attached.

ond law yields:

$$m\ddot{x} = Z_1\{x, x_1\} + Z_2\{x, x_2\} + ... + Z_N\{x, x_N\} = \sum_{i=1}^{N} Z_i\{x, x_i\} \tag{2.1}$$

Thus, we see that the net impedance (or net force) of a set of non-linear impedances acting on an inertial object is the linear superposition of those non-linear impedances. We can observe the same behavior for impedance controllers acting on a robot (which is fundamentally a set of inertial linkages). Given a set of N nonlinear endpoint-space impedance controllers that we would like to apply to a robot, our control law would be:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\big[\sum_{i=1}^{N} Z_i\{\boldsymbol{x}, \boldsymbol{x}_i\}\big] \tag{2.2}$$

We can also include impedances relative to any arbitrary coordinate system, as long as we have a Jacobian matrix that maps the joint-space coordinates to the new coordinate system. Finally, we can also add in M joint-space impedances into this framework as well (the Jacobian for this transformation is, of course, the identity matrix):

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\big[\sum_{i=1}^{N} Z_i\{\boldsymbol{x}, \boldsymbol{x}_i\}\big] + \sum_{j=1}^{M} Z_j\{\boldsymbol{q}, \boldsymbol{q}_j\} \tag{2.3}$$

2.2 Redundancy Resolution

One application of this linear superposition principle is redundancy resolution. Consider our example planar manipulator (reproduced in Figure 2-2). Since this robot has more joint-space degrees of freedom than endpoint degrees of freedom ($n > m$), it is a redundant manipulator. This means that the joint angles can be actuated continuously without impacting the position or orientation of the endpoint.

To better understand this, we first recall the velocity kinematics of the robot (Eqn. 1.13). Here, the $m \times n$ Jacobian matrix maps the n-dimensional joint-space velocity

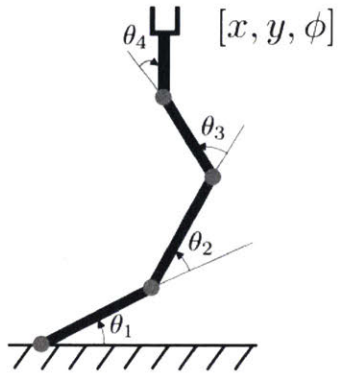


Figure 2-2: A 4-DOF planar robot. In this example, $n = 4$ and $m = 3$

vector to the m-dimensional endpoint-space velocity vector. If we assume the robot is not in a singular configuration, the rank of the Jacobian is m (full row rank). Thus, we see that the Jacobian has a null space with $n - m$ basis vectors. Any joint space velocity vector $\dot{\boldsymbol{q}}$ in this null space will cause no motion in $\dot{\boldsymbol{x}}$, and will therefore not be impacted by the endpoint impedance controller (Eqn. 1.29). Consequently, small disturbance forces in the joint space of the robot can cause large, uncorrected motions in the joint-space.

To counter this, we can add a second impedance controller that operates in the joint space of the robot. A simple controller might specify a small stiffness about a nominal joint-space pose, $\boldsymbol{q}_0$, along with a small joint-space damping. This controller has the form:

$$\boldsymbol{\tau}_{act} = \boldsymbol{K}_q(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{B}_q(\dot{\boldsymbol{q}}) \tag{2.4}$$

Here, $\boldsymbol{K}_q \in \mathbb{R}^{n \times n}$ is the desired joint-space stiffness matrix, and $\boldsymbol{B}_q \in \mathbb{R}^{n \times n}$ is the desired joint-space viscous damping matrix. These matrices are usually diagonal, in which case the entries correspond to independent virtual rotational springs and dampers placed at each joint of the robot. The final control law is simply the sum of the two impedance controllers:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\big[\boldsymbol{K}_x(\boldsymbol{x_0} - L(\boldsymbol{q})) + \boldsymbol{B}_x(\dot{\boldsymbol{x}}_0 - \boldsymbol{J}\dot{\boldsymbol{q}})\big] + \boldsymbol{K}_q(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{B}_q(\dot{\boldsymbol{q}}) \tag{2.5}$$

This is the controller that was implemented on hardware in the later sections of this

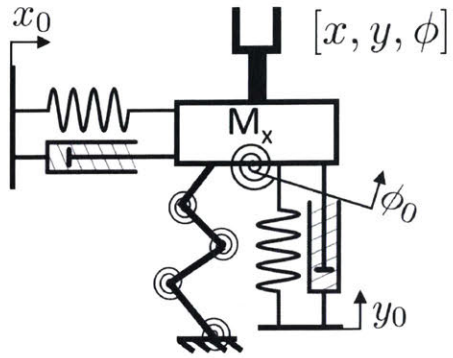


Figure 2-3: The effective behavior of a planar manipulator under a superimposed endpoint and joint-space control law, as in Eqn. 2.5.

work. With this controller, all disturbances in the joint space will be met with a restoring force to bring the manipulator back towards its nominal joint space configuration. The effective endpoint of this control law is visualized in Figure 2-3.

One drawback of this method is that the effective endpoint stiffness and damping is no longer completely described by $\boldsymbol{K}_x$ and $\boldsymbol{B}_x$. Instead, there is also some contribution to the endpoint stiffness and damping behavior from the small virtual springs and dampers that were placed at the robot joints. To analyze the net endpoint impedance behavior, we can map the joint-space stiffness matrix into endpoint-space.

For a robot, the map from a full-rank joint-space stiffness matrix $\boldsymbol{K}_q$ to an endpoint-space stiffness matrix $\boldsymbol{B}_x$ is given by [Mussa-Ivaldi and Hogan, 1991]:

$$\boldsymbol{K}_x = (\boldsymbol{J}(\boldsymbol{K}_q - \boldsymbol{\Gamma})^{-1}\boldsymbol{J}^T)^{-1} \tag{2.6}$$

Here $\boldsymbol{\Gamma}$ is the kinematic stiffness term, which is given by:

$$\boldsymbol{\Gamma} = \frac{\partial \boldsymbol{J}^T}{\partial \boldsymbol{q}}\boldsymbol{f} \tag{2.7}$$

If we evaluate the system about zero endpoint force, the effective endpoint-space stiffness matrix becomes:

$$\boldsymbol{K}_x\big|_{\boldsymbol{f}=0} = (\boldsymbol{J}\boldsymbol{K}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{2.8}$$

Similarly, The endpoint-space damping matrix resulting from a full-rank joint

space damping matrix is given by:

$$\boldsymbol{B}_x = (\boldsymbol{J}\boldsymbol{B}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{2.9}$$

Note that these relationships are only valid when the manipulator Jacobian is of full row rank. In singular configurations (where the Jacobian loses rank), the endpoint stiffness and damping become unbounded in certain directions, since arbitrarily large forces applied in those directions will yield no motion. In contrast, the endpoint compliance matrix (the inverse of the endpoint stiffness matrix), and the inverse of the endpoint damping matrix, are both well defined quantities, and will go to zero in certain directions (instead of becoming unbounded) at singular configurations [Hogan, 1985b]. However, since stiffness and damping are more familiar properties to most readers, those parameters, rather than their inverses, will be used in this work.

From Eqns. 2.8 and 2.9, the net endpoint-space stiffness and damping matrices for the control law given in Eqn. 2.5 are:

$$\boldsymbol{K}_{x,net}\big|_{\boldsymbol{f}=0} = \boldsymbol{K}_x + (\boldsymbol{J}\boldsymbol{K}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{2.10}$$

$$\boldsymbol{B}_{x,net} = \boldsymbol{B}_x + (\boldsymbol{J}\boldsymbol{B}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{2.11}$$

Another drawback of adding in a joint-space impedance to manage redundancies is that if $\boldsymbol{q_0}$ remains fixed, there will be increasing steady-state errors in positioning if $\boldsymbol{x_0}$ is driven far away from $L(\boldsymbol{q}_0)$. In real systems, this can be mitigated by keeping $\boldsymbol{K}_q$ very small compared to $\boldsymbol{K}_x$, or by periodically re-adjusting $\boldsymbol{q}_0$ to the current joint positions (this approach to mitigating positioning errors was not pursued further in this work).

An alternate approach would be to use null-space torque projections to ensure that the torques generated by the joint-space stiffness controller never produce any torques at the robot endpoint [Dietrich et al., 2015, Khatib, 1987, Khatib, 1995, Siciliano and Slotine, 1991]. These projections, along with their advantages and disadvantages will be discussed in Chapter 6.

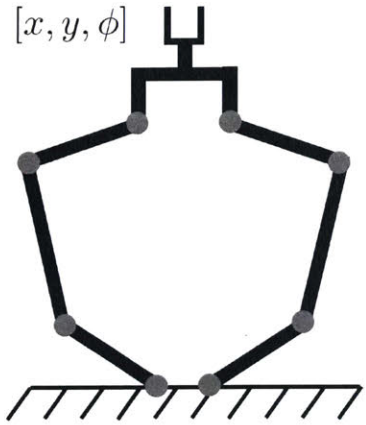


Figure 2-4: A closed kinematic chain manipulator. For simplicity, we can assume that the first three joints of each sub-chain are actuated.

2.3 Closed Kinematic Chains

The idea of impedance composition can also be used to control closed kinematic chain manipulators, such as that shown in Figure 2-4. One way to control these robots is to "cut" the closed kinematic chain into two sub-chains at an un-actuated joint, and use position control techniques to place each subchain endpoint at the desired location [Siciliano et al., 2010]. In addition to the restrictions of position control mentioned in Section 1.2.1, one additional problem is that any imperfections in the kinematic model of the robot (e.g. from manufacturing tolerances or base coordinate system measurement error) will give rise to large, and possibly unacceptable internal forces on the distal link of the closed chain.

An alternative way of controlling this closed kinematic chain mechanism is via impedance control. In the case where every sub-chain is fully actuated or redundantly actuated, we can place the endpoint of each sub-chain under impedance control. The effective behavior of this controller is visualized in Figure 2-5. The net impedance behavior at the manipulator endpoint is again the sum of the two individual chain endpoint-space impedances. Additionally, this directly gives us endpoint space control without any inverse kinematic operations. This method also naturally lends itself to superimposing additional impedances for tasks such as collision avoidance.

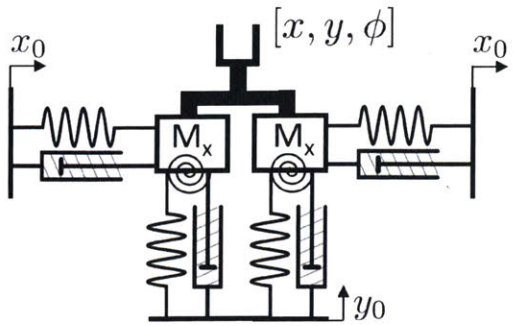


Figure 2-5: The effective behavior of a closed kinematic chain manipulator with each sub-chain under impedance control.

2.4 Collaborative Manipulation

Another (similar) application of impedance control is to simplify collaborative manipulation between multiple robots, as in Figure 2-6. One traditional way of doing this is with a master-slave manipulation paradigm, where one robot (called the master) operates under endpoint position control mode to dictate the 6-DOF positioning of the object, while all of the other robots operate in force control mode to regulate the internal forces in the object [Kosuge and Hirata, 2004, Caccavale and Uchiyama, 2016, Nakano et al., 1974, Luh and Zheng, 1987]. Since this method may put undue burden on the master manipulator to handle most of the load, one can implement hybrid position/force control on all of the cooperative manipulators to divide the positioning and force regulation tasks between them [Kosuge and Hirata, 2004, Hayati, 1986, Uchiyama and Dauchez, 1988]. These methods, while immensely useful, can be sensitive to kinematic modeling errors within and between the robots, and are not robust enough to handle situations where contact might be lost. Additionally, these control methods would not be suitable for situations where the commonly manipulated object must be brought into or out of contact with other objects or rigid constraints in the environment.

Impedance control provides an alternative way to handle the collaborative manipulation problem [Kosuge and Hirata, 2004, Koga et al., 1992]. In this case, we place both robot endpoints under impedance control, and the manipulated object behaves as if it is suspended by a set of mass/spring/damper systems, as in Figure 2-7. Here,

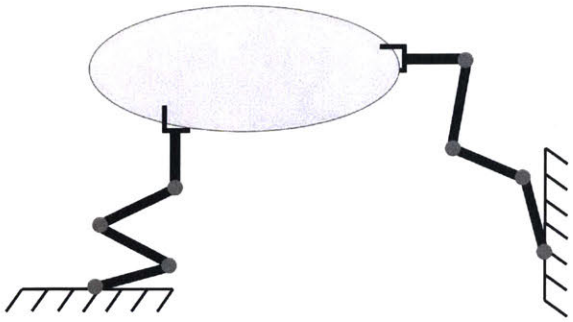


Figure 2-6: An example of two robots collaboratively manipulating an object

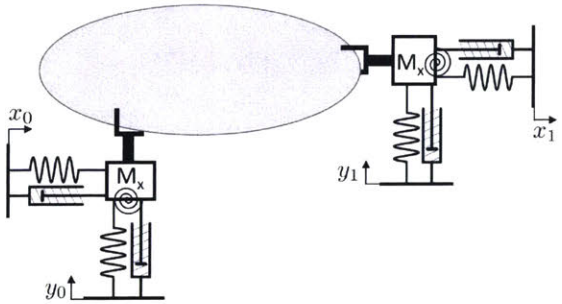


Figure 2-7: Impedance controllers on each collaborative manipulator makes the object behave as if it were suspended by a set of spring-damper systems

any errors in manipulator or object kinematics will only cause small deflections in the virtual springs at each robot endpoint, and will not result in large object internal forces. Additionally, if tensile or compressive forces are desired in the object, they can be exerted by coordinating the equilibrium points of the multiple manipulators in contact with the object. One further advantage of this paradigm is that the system remains stable even if an individual manipulator loses contact with the object. The robustness inherent in this method of manipulator coordination makes it particularly suitable for manipulation in uncertain environments, manipulation of poorly modeled, delicate, flexible, or compliant objects (such as cloths), or manipulation with low-cost robots, which may have relatively large uncertainties in their kinematics due to manufacturing tolerances and variability. As noted in the previous section, the impedance composition method also allows for secondary impedances to be superimposed to handle tasks such as collision avoidance.

2.5 Tackling the Scale Up Problem

2.5.1 Complex Tasks

Managing complex manipulation tasks requires balancing and satisfying many different task requirements. One example of a complex robotic manipulation task might be to rapidly buff a shoe using a cloth outstretched between two robotic manipulators. This is a task that humans can do with relative ease, but might pose a challenge to implement on a set of redundant robotic manipulators. In order to perform this task, the two robotic arms must first pick up a cloth, apply tension to the cloth without snapping it, bring the cloth above the shoe, bring the cloth into contact with the shoe, maintain a reasonable level of normal force on the shoe along with a reasonable amount tension in the cloth, and finally apply rapid oscillatory motions on the cloth to achieve a buffing effect.

If this is to be done with two 7-DOF manipulators, the task would require the coordinated motion of 14 degrees of freedom in a closed kinematic chain configuration incorporating the contact dynamics of a cloth, which are usually not trivial to model [Yamakawa et al., 2011, Bai et al., 2016].

This is an example of a complex problem that is made far simpler by using a compositional impedance programming approach. First, the two robot manipulators would grip a slackened rectangular cloth, with one gripper on each end of the cloth. Next, a set of endpoint-space virtual spring-dampers would be applied to the end effectors to gently pull the arms apart and apply outward tension on the cloth. Next, with the shoe placed under the cloth, a second set of spring-dampers would be applied to draw the cloth downwards to apply a normal force to the shoe. Finally, a third set of stiff spring-dampers would be applied with an oscillatory reference trajectory $\boldsymbol{x}_0$ to rapidly draw the cloth back and forth over the shoe. Throughout the process, another set of small joint-space impedances would be applied to manage the redundant degrees of freedom in the robot. Due to the built-in controller compliance, a detailed computational model of the cloth and shoe are not necessary; an approximate measurement of the cloth length and shoe position will suffice. An implementation of this

strategy on hardware is given in Chapter 5.

The strategy here was to divide a complex task into smaller sub-tasks, and to devise an impedance controller to handle each sub-task or requirement. Additionally, we can sometimes rely on manipulator compliance in lieu of developing more precise (and therefore computationally complex) models of the manipulated object and environment.

2.5.2 High-DOF Systems

Another challenge that compositional impedance control may be poised to tackle is that of controlling high-DOF systems. This may include modern humanoid robots, which may have 28 DOF in the case of Boston Dynamic's Atlas, 44 DOF in the case of NASA's R5 Valkyrie, or as many as 58 in the case of NASA's Robonaut 2 [Boston Dynamics, 2019, Paine et al., 2015, NASA, 2014]. Controlling these high-DOF systems is a challenge, since the dynamics are nonlinear, and in the case of walking humanoid robots, underactuated [Tedrake, 2019]. One common approach to tackling this control problem is to perform large scale model predictive control (MPC)/nonlinear programming (NLP) to solve for a feasible torque trajectory subject to the system dynamics as constraints to the optimization problem, and a (usually) quadratic objective function to minimize [Betts, 1998, Tedrake, 2019]. While these methods scale well for a small to moderate number of degrees of freedom, the computational complexity causes it to fail for high-DOF systems [Kuindersma et al., 2014, Valenzuela, 2016, Betts, 1998]. While progress can be made by linearizing the instantaneous dynamics at each optimization iteration, thereby converting the problem into a quadratic program (QP), the task still remains challenging [Kuindersma et al., 2014].

As mentioned previously, one of the promises of the compositional impedance control method is the ability to divide a complex interaction task into a number of sub-tasks, and to fabricate an impedance controller for each one. If an impedance approach is suitable for a manipulation scenario involving a high-DOF system, we can identify sub-tasks and use optimization methods to aid in the design of impedance controllers for each one of those sub-tasks. In this manner, we would be solving several

lower dimensional NLPs, rather than one large scale NLP (which may not converge, or whose solution may be heavily dependent on the decision variable initializations) [Valenzuela, 2016].

Chapter 3

Implementation on a Baxter Research Robot

3.1 Robot Hardware Overview

All of the experiments in compositional impedance control were performed on a Baxter Research Robot, which was manufactured by (the now defunct) Rethink Robotics (Boston, MA). The Baxter Robot is a low-cost anthropomorphic humanoid robot with two 7-DOF arms. A visualization of the robot joint naming convention, along with an image of the robot arm internals is given in Figure 3-1. A dimensioned drawing of the overall dimensions of the robot, along with the coordinate system is given in Figure 3-2.

Each joint contains a motor with a peak torque of 50 Nm for the shoulder and elbow joints, and 15 Nm for the wrist joints [Rethink Robotics, 2015g]. Each motor has the output of its gearbox connected to a relatively stiff series elastic element (843 Nm/rad on shoulder/elbow joints, and 250 Nm/rad on wrist joints), which allows for torque measurement and torque control at all joints [Rethink Robotics, 2015d, Rethink Robotics, 2015g, Hosford, 2016]. While each joint has a 14 bit encoder (yielding a 0.02° resolution), joint-level non-linearities yield a typical accuracy on the order of ±0.10° [Rethink Robotics, 2015g]. This, combined with manufacturing tolerances, yields a ±5 mm endpoint-space positioning accuracy while in position control mode

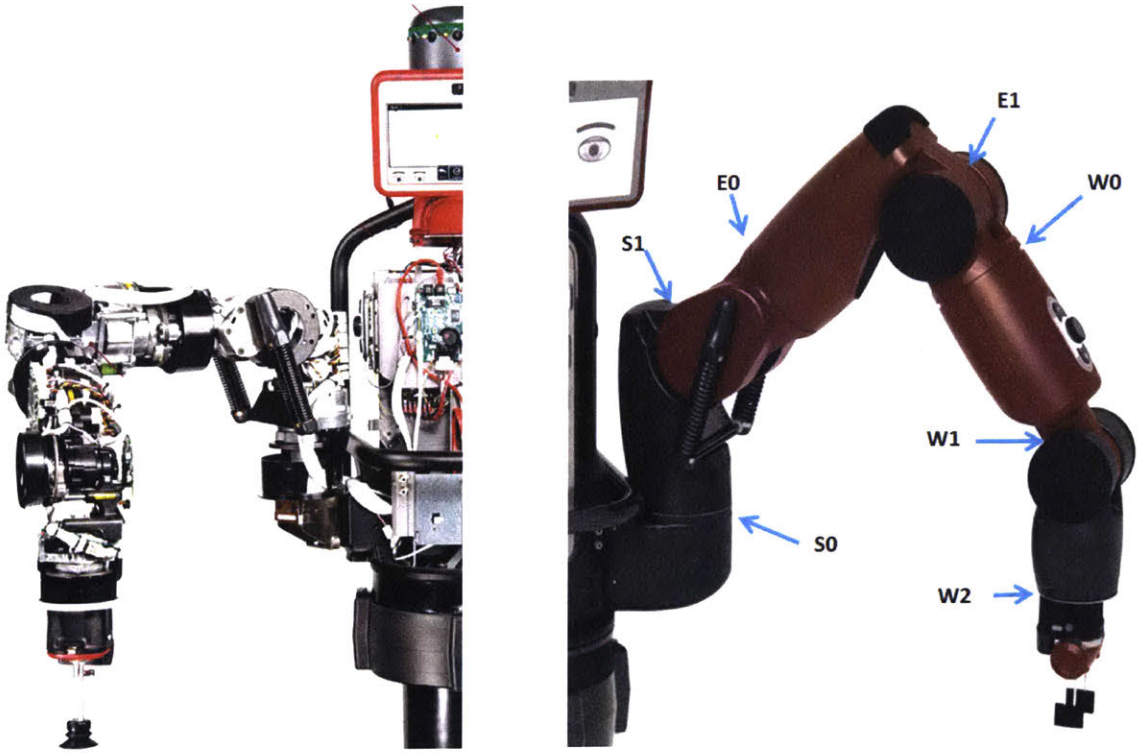


Figure 3-1: The image on the left shows a Baxter Robot arm with its cover removed. Here, one can see the joint motors, series elastic elements, and joint control boards. On the right is the joint naming convention. As this is an anthropomorphic robot, "S" stands for shoulder, "E" stands for elbow, and "W" stands for wrist. Images are adapted from [Knight, 2013] and [Rethink Robotics, 2015g]

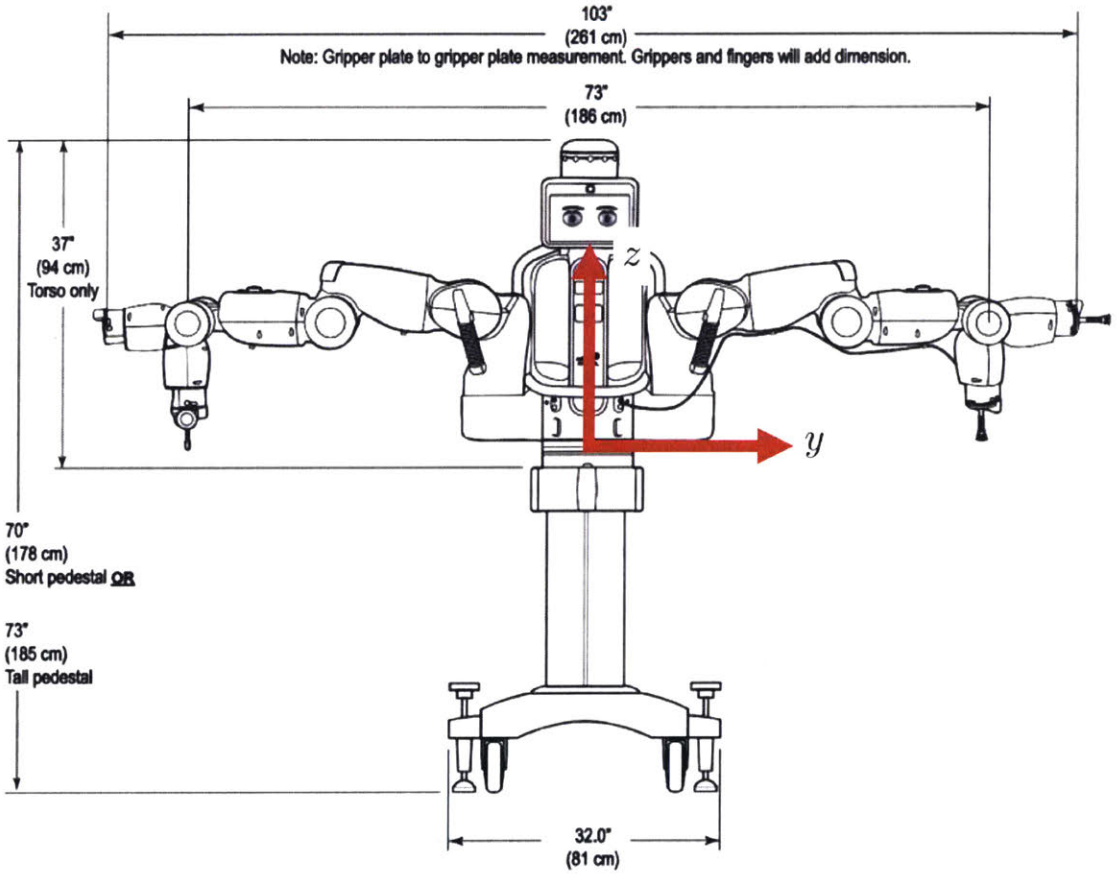


Figure 3-2: The overall dimensions of the robot, and the robot's cartesian coordinate system. The origin is located on the plane where the robot body is mounted to the pedestal. The positive x axis points out of the page. Image modified from [Rethink Robotics, 2015i].

[Rethink Robotics, 2015g]. The robot also has a built-in proprietary gravity compensation algorithm, with the parallel elastic elements on the S1 joints bearing much of the gravitational load [Rethink Robotics, 2015f]. The gravity compensation algorithm also accounts for hysteresis effects in the S1 springs, and off axis torques that are detected by the S1 torque sensor [Rethink Robotics, 2015a, Rethink Robotics, 2015f].

3.2 Robot Control Overview

The Baxter Robot control system is built over ROS, which is an asynchronous system for shuttling data between many different components such as sensors, motor controllers, and various software programs running on either the same, or different computers [Open Source Robotics Foundation, 2019]. Each component, or "node" in the ROS framework can broadcast, or "publish" data to the ROS network, and can "subscribe" to data coming from other nodes in the ROS network. No upper bounds are provided for message latency, and message delivery is not guaranteed.

Several different control modes are available on the Baxter Robot, including position control, raw position control, velocity control, and torque control [Rethink Robotics, 2015b, Hosford, 2016]. In this work, the torque control mode was used exclusively. A block diagram outlining the details of running a high-level torque controller on a Baxter Robot in torque control mode is provided in Figure 3-3. The implementation is as follows [Rethink Robotics, 2015c, Rethink Robotics, 2015g]: First, the user programs the high-level torque control law (in our case, an impedance control law) on a Linux Machine, typically in Python. The script can obtain the latest measured values of $\boldsymbol{\tau}$, $\boldsymbol{q}$, and $\dot{\boldsymbol{q}}$ for all 14 joints over a subscription to the ROS Topic (data stream). It should be noted that Rethnk Robotics provides a python API to handle much of the ROS subscribing and publishing [Rethink Robotics, 2015d]. The user's control loop then computes commanded torque values, $\boldsymbol{\tau}_{cmd}$ for all joints. This loop was operated reliably at 520 Hz on the author's hardware (Intel Core i7-7500U @ 2.70 GHz). These torque values are then published over the ROS network and sent to

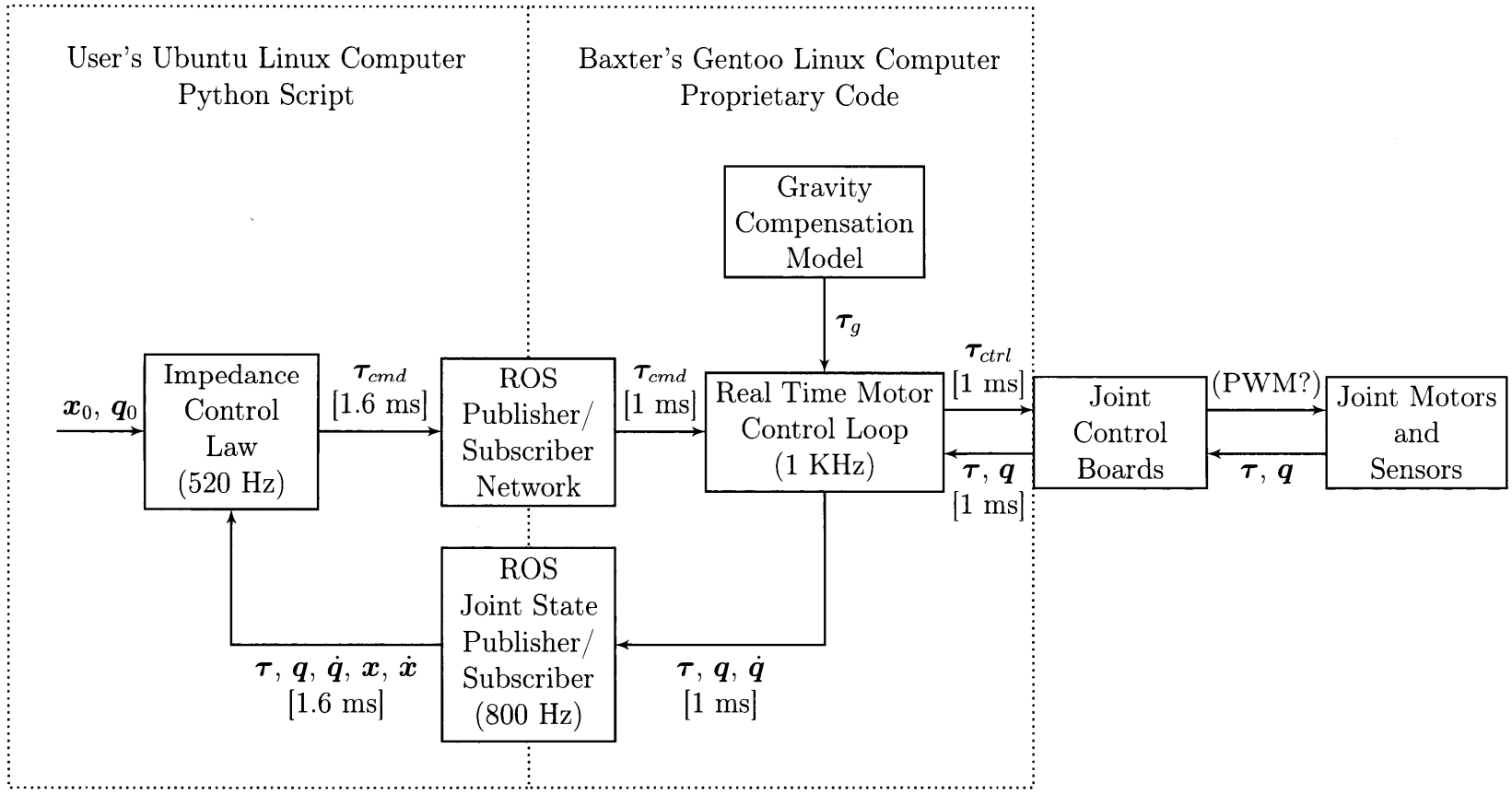


Figure 3-3: Running an impedance controller in the Baxter Robot's torque control mode. All signals are labeled, with typical message latency times in brackets. Note that there are, in actuality, 14 Joint Control Board and Motor/Sensor blocks (one for each joint on the robot). All latency data were obtained from [Rethink Robotics, 2015c]

the robot over an Ethernet LAN connection. This step has a typical latency of 1.6 ms. Next, subscribers on the Baxter Robot's internal Gentoo Linux machine (Intel Core i7-3770 @ 3.4 GHz) listen for the torque commands and forward them to the robot's Real Time Motor Control Loop, with a typical latency of 1 ms. The Real Time Motor Control Loop is the highest priority process on the robot's internal computer, and operates at 1 KHz. Other processes, such as the gravity compensation modules run in parallel with this loop. The Real Time Motor Control Loop incorporates gravity compensation torques, S1 spring compensation torques, and torque limit scaling on $\boldsymbol{\tau}_{cmd}$ in a proprietary manner to produce $\boldsymbol{\tau}_{ctrl}$, which is then forwarded to each of 14 Joint Control Boards. These are microcontrollers associated with each joint. This step has a typical latency of 1 ms. Finally, the Joint Control Boards power the motors, presumably via pulse width modulation. The low-level controller structure and particular gains used are proprietary.

Simultaneously, the Joint Control Boards measure the joint angles and joint torques from the encoders and series elastic elements attached to each motor. These values are fed back to the Real Time Motor Control Loop with a typical latency of 1 ms. The Real Time Motor Control Loop then forwards these with a 1 ms latency to a ROS publisher, which broadcasts these values reliably at about 800 Hz. These values are sent over an Ethernet LAN connection to the ROS subscriber on the user's high-level control loop with a 1.6 ms typical latency. The cumulative loop latency for this control stack is roughly 7.2 ms [Rethink Robotics, 2015c].

3.3 High Level Impedance Controller Implementation Details

The impedance control law used in the following sections was that of Eqn. 2.5, with the endpoint space damping term specified relative to ground, or relative to $\dot{\boldsymbol{x}}_0$, depending on whether or not the refrence trajectory, $\boldsymbol{x}_0$ was differentiable. In the former case, $\dot{\boldsymbol{x}}_0$ was set to zero in the control law, regardless of the value of $\boldsymbol{x}_0$. Additional

impedances were superimposed as necessary. Whenever required, kinematic properties such as the Jacobian matrix, joint-space inertia matrix, and endpoint-space inertia matrix were all computed from the Rethink Robotics Provided URDF (an XML based description of the robot kinematics), using an implementation of the Orocos PyKDL Library [Rethink Robotics and Maroney, 2015, Rethink Robotics, 2015h]. The suitability of this library for use in the impedance controller was confirmed by [Hosford, 2016].

One notable implementation aspect is performing computations with rotation angles in a numerically stable manner. The endpoint stiffness term in Eqn. 2.5 requires us to compute the difference, $\boldsymbol{x}_0 - \boldsymbol{x}$. This is trivial for the Cartesian x, y, and z components of $\boldsymbol{x}$, but the rotational parts are more complicated. If we wish $\boldsymbol{x}$ to be a minimal set of configuration variables (that is, have $\boldsymbol{x} \in \mathbb{R}^6$), one common choice of representation would be Euler angles, which would give $\boldsymbol{x}$ the structure $\begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix}^T$. When dealing with such Euler angles, one must be careful to define the convention being used. This is due to the fact that there are 12 different ways to define rotations from a set of Euler angles, depending on whether fixed (extrinsic) or object (intrinsic) coordinates are used, and whether one of the proper Euler sequences (e.g. ZYZ) or one of the Tait-Bryan/Cardan sequences (e.g. XYZ) are used [Diebel, 2006, Spong and Vidyasagar, 1989].

When using sets of Euler angles to register finite rotations, subtracting them to define $\Delta\boldsymbol{x}$ might not be physically meaningful, except perhaps for small differential rotations. Furthermore, Euler angles are subject to singularities. For instance, in order to define a unique set of angles for each rotation, the Euler angles are given ranges. For example, Tait-Bryan angle ranges are often defined as $\phi \in [0, 2\pi]$, $\psi \in [0, 2\pi]$, $\theta \in [0, \pi]$. In the case of an orientation where one of the angles is at the edge of its permissible range, even a minute rotation can cause drastic differences in all three values, making the subtraction physically meaningless even for infinitesimally small differences. Another example of a singularity is called gimbal lock, when the body is in a configuration such that a rotation in a certain direction can be encoded by multiple sets of Euler angles (i.e. two of the three Euler angles become interchangeable)

[Diebel, 2006].

In order to avoid these problems, encoding rotations in quaternions (a non-minimal representation) is far more desirable, as unit quaternions describing rotations are always continuous, and free from singularities [Hanson, 2006]. Consequentially, the rotational portion of $\boldsymbol{x}_0$ was converted to quaternion form[1], and the rotational stiffness portion of the impedance control law was implemented directly with quaternions, as in [Caccavale et al., 1999, Caccavale et al., 2008].

[1]The rotational part of $\boldsymbol{x}$ provided by the robot's Joint State Publisher is already given in quaternion form

Chapter 4

Verifying Impedance Composition and Controller Performance

4.1 Overview and Methodology

This chapter describes a set of system identification experiments performed on the Baxter Robot, under the impedance control law given in Eqn. 2.5, with endpoint-space damping set relative to ground:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\left[\boldsymbol{K}_x(\boldsymbol{x_0} - L(\boldsymbol{q})) - \boldsymbol{B}_x\boldsymbol{J}\dot{\boldsymbol{q}}\right] + \boldsymbol{K}_q(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{B}_q(\dot{\boldsymbol{q}}) \tag{4.1}$$

The experiments had two primary aims. The first was to verify how well a low-cost torque-controlled robot such as Baxter could provide a commanded endpoint stiffness/damping impedance. The second was to verify how well the impedance at each endpoint would linearly superimpose in a closed kinematic chain configuration, similar to the setup in Figures 2-4 and 2-5. For simplicity, all identification experiments were carried out along a single endpoint Cartesian dimension (the robot's x axis). The robot was also fully calibrated before all experimentation [Rethink Robotics, 2015a].

For these experiments, the left and right robot arm endpoints were first servoed to symmetric positions about the robot's sagittal plane, with all joint positions closely mirrored. An exact joint-space mirroring cannot be guaranteed since each robot arm

Table 4.1: Nominal starting endpoint coordinates for system ID, $\boldsymbol{x}$, with endpoint rotation expressed as a quaternion.

x	y	z	q_x	q_y	q_z	q_w
0.80 m	$\pm$0.135 m	0.211 m	$\frac{\sqrt{2}}{2}$	0.0	$\frac{\sqrt{2}}{2}$	0.0

Table 4.2: Ideal (Perfectly Mirrored) and Actual Joint Angles, $\boldsymbol{q}_0$ [rad] The Right/Left Difference row indicates the difference in magnitudes between the right and left arm joint angles.

	S0	S1	E0	E1	W0	W1	W2
Nominal Angles	$\pm$0.0820	0.2963	$\pm$1.3254	1.7641	$\mp$0.4177	$-$1.1360	$\pm$1.7603
Actual Right Arm	0.0993	0.2848	1.3352	1.7702	-0.4039	-1.1449	1.7374
Actual Left Arm	-0.0842	0.2911	-1.3311	1.7685	0.4081	-1.1381	-1.7512
Right/Left Difference	0.0151	0.0063	0.0041	0.0017	0.0042	0.0068	0.0138

is hand-welded to the robot torso, resulting in slightly different (but known) forward kinematics transformations from the base frame to the endpoint for each arm [Rethink Robotics, 2015e]. The nominal endpoint positions used, along with ideal and actual initial joint angles are shown in Tables 4.1 and 4.2. An image of Baxter at these equilibrium joint positions is shown in Figure 4-1.

The controller in 4.1 was activated, and impedance parameter identification was performed on each arm individually. Next, to test how well the impedances superimpose in the closed chain configuration, the two arm endpoints were bolted together with a 0.265 kg aluminum linkage (Figure 4-2), while maintaining the initial joint angles used in the previous experiment. A bolted linkage was chosen to provide a closed-chain configuration, rather than a commonly gripped object, or custom connectable end effectors, due to its simplicity and rigidity. The experimental setup with the arms bolted is shown in Fig. 4-3. Impedance parameter identification was then performed at the center point of the linkage.

For each individual arm, our expected net endpoint-space stiffness matrix arising from this controller is given by Eqn. 2.10:

$$\boldsymbol{K}_{x,total}\big|_{\boldsymbol{f}=0} = \boldsymbol{K}_x + (\boldsymbol{J}\boldsymbol{K}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{4.2}$$

Figure 4-1: The Robot's pose at the beginning of the system ID experiments. The image on the left is a top view from the Gazebo simulator, and the image on the right is an angled view of the actual robot.

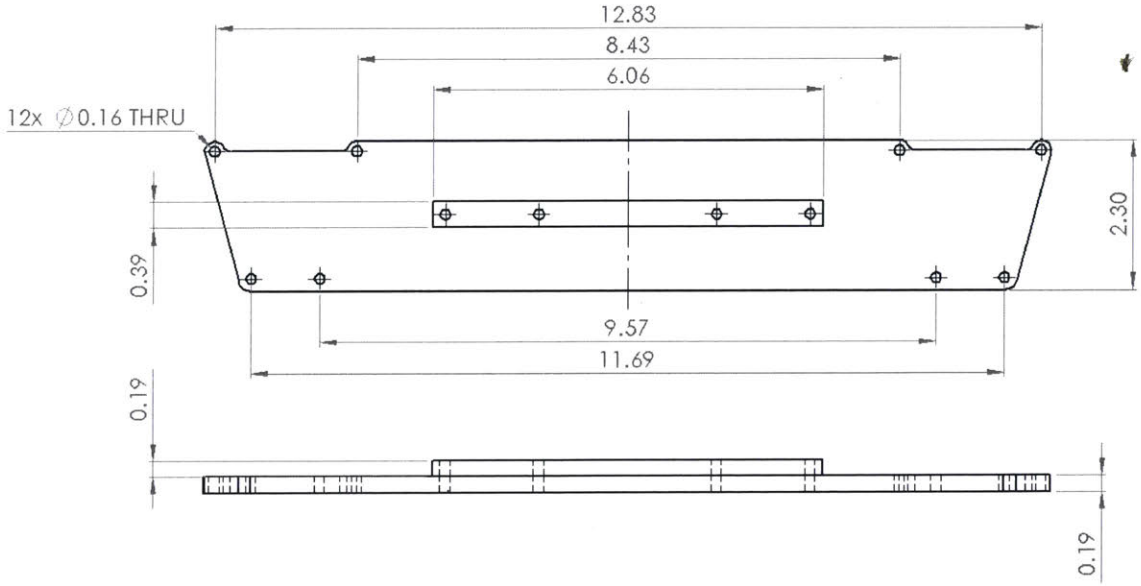


Figure 4-2: A Drawing of the 6061 aluminum linkage used to bolt the arm endpoints together. The small center rectangular reinforcement rib was bolted onto the main linkage body to add extra rigidity. All dimensions are in inches.

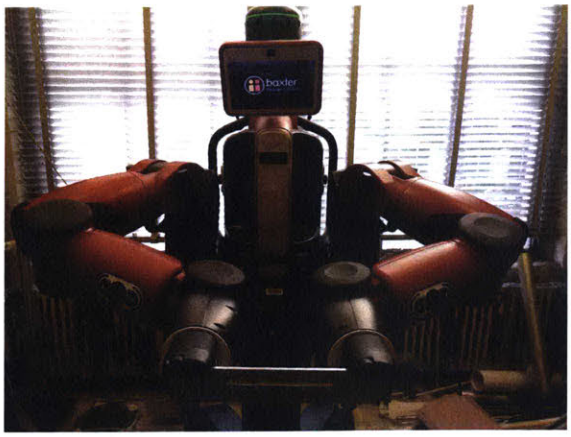


Figure 4-3: Angled view of the Baxter Robot with both arms bolted together in a closed-kinematic chain configuration.

An expression for the expected net endpoint damping matrix was also given by Eqn. 2.11. However, that equation assumed a frictionless robot/mechanism. In a real system such as the Baxter Robot, the joint friction may be considerable. A basic estimate of the Baxter Robot's friction (approximated as joint viscous damping) was provided in the manufacturer's URDF file [Rethink Robotics, 2015h], which we take to be a reasonable first approximation of actual robot friction. The joint-space friction matrix arising from the joint viscous damping parameters given by the manufacturer is:

$$\boldsymbol{B}_f = \text{diag}(\begin{bmatrix} 0.7 & 0.7 & 0.7 & 0.7 & 0.7 & 0.7 & 0.7 \end{bmatrix}) \tag{4.3}$$

Here, the damping vector is structured as: $\begin{bmatrix} b_{S0} & b_{S1} & b_{E0} & b_{E1} & b_{W0} & b_{W1} & b_{W2} \end{bmatrix}$, and $\boldsymbol{B}_f \in \mathbb{R}^{7\times 7}$. All damping values are given in $\frac{\text{Nms}}{\text{rad}}$. We can add this joint space friction matrix to our commanded joint space damping matrix to obtain a more accurate estimate of our endpoint damping matrix:

$$\boldsymbol{B}_{x,total} = \boldsymbol{B}_x + (\boldsymbol{J}(\boldsymbol{B}_q + \boldsymbol{B}_f)^{-1}\boldsymbol{J}^T)^{-1} \tag{4.4}$$

The expected endpoint-space inertia matrix is, at first glance, given by taking the inverse of both sides of Eqn. 1.22:

$$\boldsymbol{M}_x = (\boldsymbol{J}\boldsymbol{M}_q^{-1}\boldsymbol{J}^T)^{-1} \tag{4.5}$$

However, [Hosford, 2016] has shown that the Baxter robot has non-negligible reflected motor inertia. Hosford measured the reflected motor inertias on joints E1 and W1. Since all of the wrist motors have identical specifications in terms of maximum joint speed, joint flexure stiffness, and peak torque, we can extrapolate the W1 measurement to W0 and W2 [Rethink Robotics, 2015g]. Similarly, since all of the shoulder and elbow motors have identical specifications for the aforementioned parameters, we can extrapolate the E1 measurement to S0, S1, and E0 [Rethink Robotics, 2015g]. This approximation gives us [Hosford, 2016]:

$$\boldsymbol{I}_r = \text{diag}(\begin{bmatrix} 0.60 & 0.60 & 0.60 & 0.60 & 0.044 & 0.044 & 0.044 \end{bmatrix}) \tag{4.6}$$

We can incorporate these reflected motor inertias into our computation as follows:

$$\boldsymbol{M}_{x,total} = (\boldsymbol{J}(\boldsymbol{M}_q + \boldsymbol{I}_r)^{-1}\boldsymbol{J}^T)^{-1} \tag{4.7}$$

It is important to note that the joint gear ratio values were not disclosed by the manufacturer. It is thus entirely possible that there are different gear ratios in the joints within each grouping (S0 - E1 and W0 - W2), which would render this approximation invalid. However, since the identical parameters mentioned above heavily depend on the gear ratio, this approximation is a reasonable one to make.

Under the control law in Eqn. 4.1, starting from a static equilibrium position where $\boldsymbol{x}_0 = L(\boldsymbol{q}_0)$, if motion only occurs along the x direction, and $\boldsymbol{q}$ does not deviate significantly from $\boldsymbol{q}_0$, each robot arm should exhibit behavior along the x coordinate like that of the system in Figure 1-5, with forward path transfer function:

$$\frac{X}{X_0} = \frac{k_x}{m_x s^2 + b_x s + k_x} \tag{4.8}$$

and disturbance response (back-driving dynamics):

$$\frac{F_{ext}}{X} = m_x s^2 + b_x s + k_x \tag{4.9}$$

In these equations, k_x is the upper left diagonal entry in $\boldsymbol{K}_{x,net}\big|_{\boldsymbol{f}=0}$, b_x is the upper left diagonal entry in $\boldsymbol{B}_{x,total}$, and m_x is the upper left diagonal entry in $\boldsymbol{M}_{x,total}$. Note that these equations are the same as Eqns. 1.10 and 1.7.

The identification problem now reduces to finding the transfer function of a locally linear SISO system. One common approach to performing this identification is to excite the system with a set of input sinusoidal waves of different frequencies, and measure the ratio of the output to input magnitudes and phase differences. This allows us to construct a Bode plot, and thereby estimate the dynamic parameters [Phillips and Parr, 2011, Franklin et al., 2015]. However, since we are studying a high order system (if we neglect the series elastic actuators, each individual arm is at minimum, a 14th order system, and the closed-chain configuration is a 28th order system with six algebraic constraints), there is some concern that other misleading resonant behaviors in the joint-space may appear at different excitation frequencies. This might have made the resulting Bode plot challenging to interpret. Due to this potential difficulty, the dynamic parameters were identified using step responses in x_0, and fitting the resulting $x(t)$ with time-domain step responses of Eqn. 4.8.

One concern with performing system identification with step responses in the forward path dynamics, is that the transfer function can only be fit to within a constant. For instance, if stiffness, mass, and damping were all doubled, the resulting transfer function in Eqn. 4.8 would be identical. Thus, a minimum of two experiments need to be done in order to uniquely identify all three parameters. In this case, static stiffness was identified by pushing on the robot endpoint extremely slowly, such that inertial and damping forces are negligible. This reduces Eqn. 4.9 to:

$$\frac{F_{ext}}{X} = k_x \tag{4.10}$$

This allows us to identify the endpoint stiffness (k_x) from a simple linear regression. This endpoint stiffness could then be fixed, with the transient forward-path step response data used to uniquely fit b_x and m_x.

The closed chain system identification was carried out in an identical manner

(using both a slow interactive measurement for static stiffness, and a forward step response for damping and inertia). For this configuration, the endpoint of the linkage will have a nominal stiffness of $2k_x$, a damping value of $2b_x$, and an inertia value of $2m_x + m_l$ where m_l is the mass of the linkage connecting the two arms.

One point that should be emphasized is that the parameters k_x, b_x, and m_x would be the same for Eqn. 4.8 (the forward path dynamics) and Eqn. 4.9 (the back-driving dynamics) in an ideal linear system. In an acutal hardware implementation such as Baxter, there may be significant discrepancies between the forward-path dynamics and back-driving dynamics [Hosford, 2016]. While the apparent endpoint inertia and static stiffness should appear to be similar, there might be significant non-linear frictional effects that might differ between the forward-path response and the back-driving response. Coulomb friction, which is present in virtually all mechanical systems, is inadequately accounted for by a linear damping term, and is also dependent on the driving direction. If the robot endpoint is slowly pushed from its nominal position and then slowly allowed to return to its nominal position (as in the aforementioned static stiffness identification experiment), a significant Coulomb friction relative to controller damping would appear as hysteresis in a plot of Eqn. 4.10. Additionally, there may be many other dynamics (e.g. those arising from motor amplifiers, etc.) which may further contribute to differences between the forward-path and back-driving system responses.

For the purposes of this work, however, we regard the forward-path and back-driving dynamics as interchangeable, as we obtain the parameter k_x from the back-driving response, and we obtain b_x and m_x from the forward-path response. A more sophisticated (and challenging) approach would be to directly measure all impedance parameters from the back-driving dynamics (as that is what matters more in an interaction task). One potential method for doing this would be to have one robot arm in a high-gain position control loop drive the impedance controlled second arm. A discussion of the methods and challenges for this approach is given in Section 7.2.1.

Another limitation in the scope of this work is that the robot endpoint impedance was only measured along the x direction. Linear and rotational impedances along

other endpoint-space coordinates were not measured.

4.2 Controller Settings

The system ID was performed for three different sets of controller endpoint stiffness values. The vectors were structured as $\begin{bmatrix} k_x & k_y & k_x & k_{\theta x} & k_{\theta y} & k_{\theta z} \end{bmatrix}$, with linear stiffnesses having units of N/m, and rotational stiffnesses having units of N-m/rad.

$$\boldsymbol{K}_{x,500} = \text{diag}(\begin{bmatrix} 500 & 500 & 500 & 5 & 5 & 5 \end{bmatrix}) \tag{4.11}$$

$$\boldsymbol{K}_{x,400} = \text{diag}(\begin{bmatrix} 400 & 400 & 400 & 5 & 5 & 5 \end{bmatrix}) \tag{4.12}$$

$$\boldsymbol{K}_{x,300} = \text{diag}(\begin{bmatrix} 300 & 300 & 300 & 5 & 5 & 5 \end{bmatrix}) \tag{4.13}$$

The joint space stiffness was held constant across all trials. The joint space vector was structured as: $\begin{bmatrix} k_{S0} & k_{S1} & k_{E0} & k_{E1} & k_{W0} & k_{W1} & k_{W2} \end{bmatrix}$ with all values having units of N-m/rad.

$$\boldsymbol{K}_q = \text{diag}(\begin{bmatrix} 0.005 & 12 & 5 & 0.005 & 0.005 & 0.005 & 0.005 \end{bmatrix}) \tag{4.14}$$

The elevated stiffness values for S1 and E0 were chosen to help counteract observed drift arising from imperfections in the gravity compensation controller.

The endpoint and joint-space damping matrices were constant throughout the transient analysis. They were as follows, with values reported in N-s/m and N-m-s/rad:

$$\boldsymbol{B}_x = \text{diag}(\begin{bmatrix} 20 & 30 & 20 & 0.3 & 0.5 & 0.8 \end{bmatrix}) \tag{4.15}$$

$$\boldsymbol{B}_q = \text{diag}(\begin{bmatrix} 0.01 & 0.01 & 0.01 & 0.01 & 0.01 & 0.01 & 0.01 \end{bmatrix}) \tag{4.16}$$

The resulting net endpoint stiffness, damping, and inertia matrices arising from these controller settings are computed in Appendix A.

4.3 Static Stiffness Identification

The static stiffness values were obtained by pushing the individual robot arm endpoints, and linkage midpoint along the $-x$ direction at a rate of less than 5 mm/s for a total displacement of 5-7 cm. The arm endpoint was then slowly released at the same rate until it reached its nominal starting position. A linear fit was then performed on the resultant force vs. displacement curves.

Since Baxter does not have an endpoint force/torque sensor, the endpoint forces and torques were computed from measured SEA torques at each joint. This conversion was implemented in the robot's software by the manufacturer, and presumably performs the computation [Maroney, 2014]:

$$\boldsymbol{f} = \boldsymbol{J}^{+T}(\boldsymbol{\tau}_{SEA} - \boldsymbol{g}) \tag{4.17}$$

where J^{+T} is the Moore-Penrose pseudo inverse of the Jacobian transpose, $\boldsymbol{\tau}_{SEA}$ are the measured SEA torques, and $\boldsymbol{g}$ are the computed gravity compensation torques, which include compensation for the parallel springs on the S1 joint, and corrections for torque sensor distortion from off-axis torques [Rethink Robotics, 2015a].

For the static stiffness verification, all force vs. displacement curves were linear, and matched quite well with the expected stiffness values. The force vs. displacement curves, along with linear fits are shown in Figures 4-4, 4-5, and 4-6 for all three stiffnesses. The results are summarized in Table 4.3. In this table, the first column indicates the nominal endpoint stiffness value (from $\boldsymbol{K}_x$), and the second column indicates the expected value of k_x for a single arm from $\boldsymbol{K}_{x,total}$. The rest of the columns indicate the fit parameters, and percentage error from the expected value of k_x or $2k_x$, for the single arm and closed-chain configuration, respectively.

Some hysteresis effects were seen on the left arm in the plots given in Figures 4-5 and 4-6. This characteristic, consistent offset between the measured forces which arise from slowly pushing the endpoint versus slowly releasing the endpoint indicates the presence of of non-negligible coulomb friction in the robot.

All measured individual arm stiffnesses were within 2% of the commanded value,

and all closed chain stiffnesses were within 3% of double the commanded stiffness for each arm. These results show that a precise manipulator static impedance can be readily achieved, at least locally, even on a fairly low-cost robot arm. Furthermore, static impedance terms linearly superimpose for the closed chain configuration/for multiple manipulators in contact with a common object.

4.4 Damping and Inertia Identification

The damping and inertia values were obtained by subjecting the manipulator to a 5 cm step in x_0, while recording the resulting actual endpoint trajectory, $x(t)$. Using the fixed stiffness values obtained from the static stiffness identification trials, the parameters b_x and m_x were obtained by performing a nonlinear fit of x(t) to a 5 cm step applied to the second order transfer function in Eqn. 4.8, using the nonlinear least-squares solver lsqcurvefit in MATLAB (Mathworks, Natick, MA). The step response data and corresponding fits are shown in Figures 4-7, 4-8, and 4-9. While some damped residual oscillations occurred in the wrist joints of the left arm (presumably due to backlash) in the single arm experiments, the endpoint data obtained was quite clean.

4.4.1 Damping Parameter Fits

The damping data are summarized in Table 4.4. In this table, the first column indicates the nominal endpoint stiffness value of the trial, and the second column indicates the expected value of b_x for a single arm from $\boldsymbol{B}_{x,total}$. The left and right arm columns express the percentage error relative to the expected value of b_x. For the closed chain column, the percentage error was calculated in two ways. The first way was relative to double the single arm expected value (to evaluate controller performance), and other way was relative to the sum of the two measured damping terms (to evaluate compositionality). Since there are many unmodeled sources of friction in an actual robot, it is quite challenging to specify an actual endpoint damping term. The 0.7 Ns/rad viscous damping factor provided in the manufacturer URDF file is

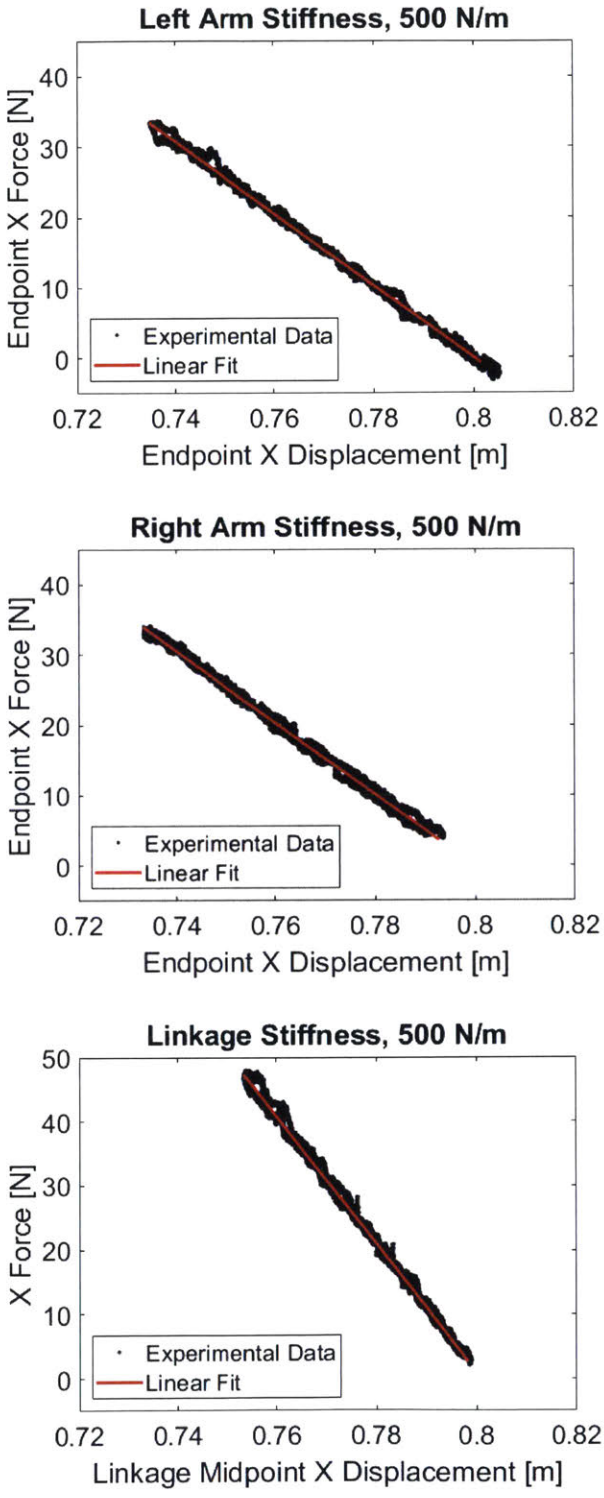


Figure 4-4: Force vs. Displacement curves for the 500 N/m static stiffness trials

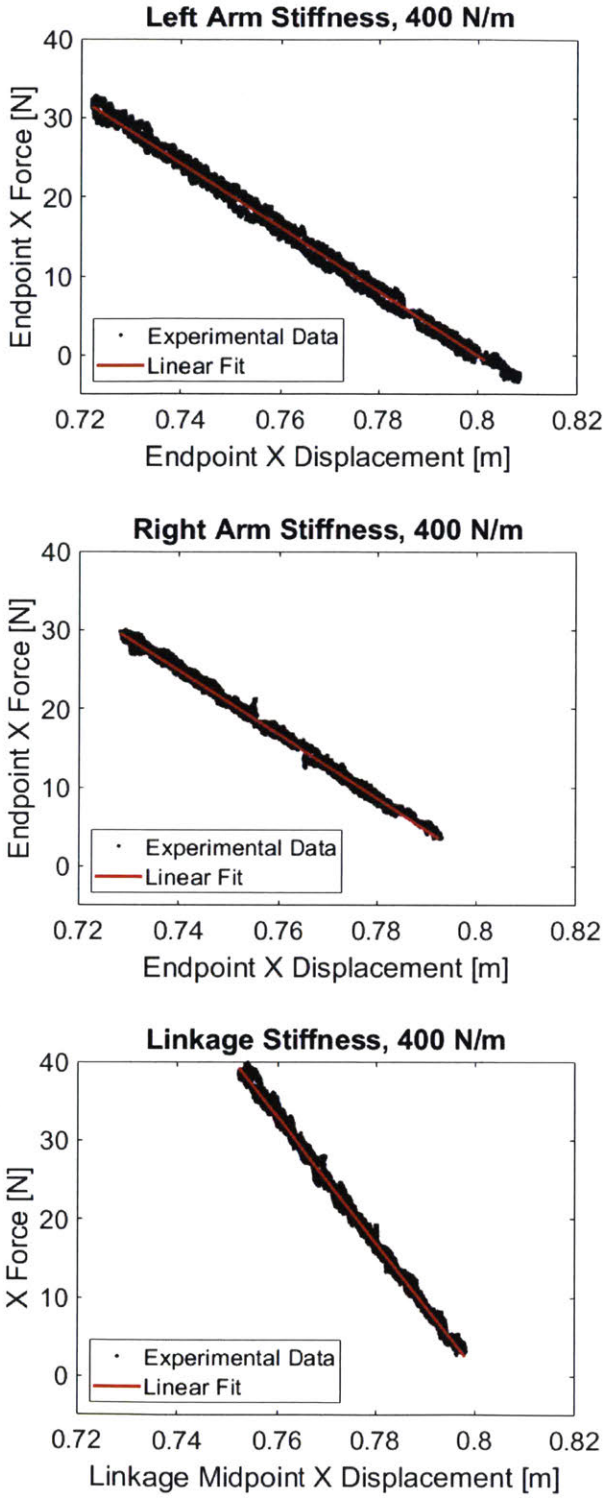


Figure 4-5: Force vs. Displacement curves for the 400 N/m static stiffness trials

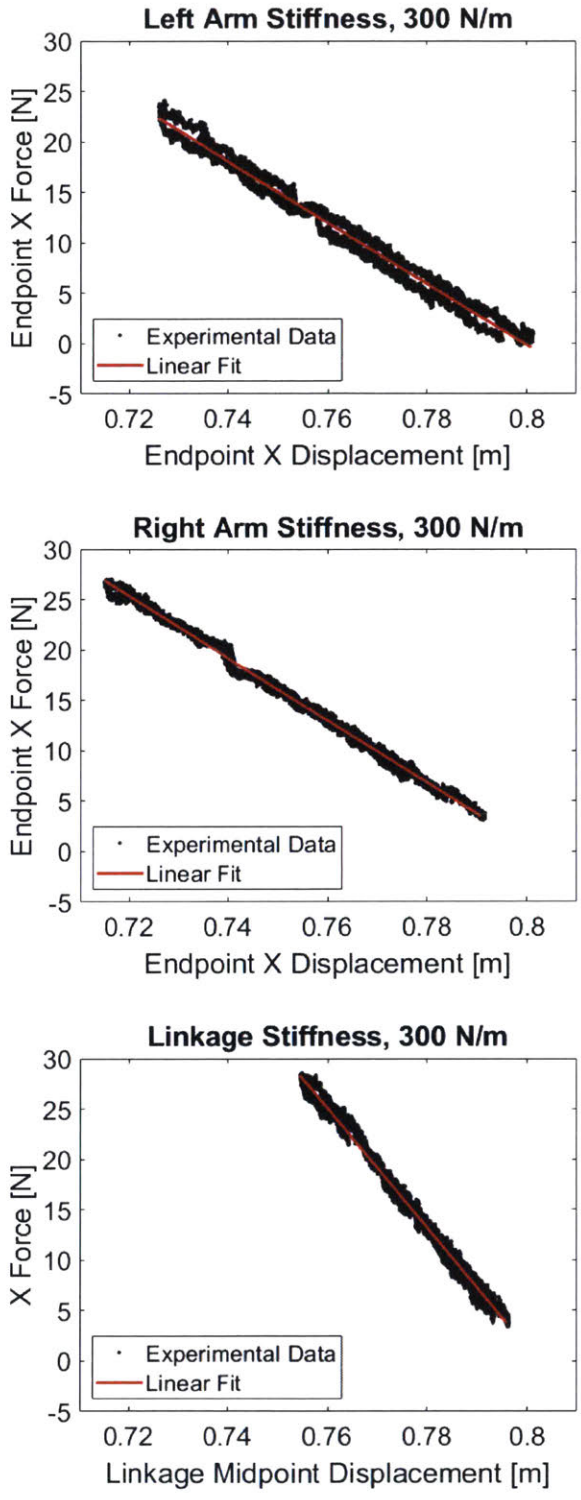


Figure 4-6: Force vs. Displacement curves for the 300 N/m static stiffness trials. Note the hysteresis on the left arm.

Table 4.3: Results From Static Stiffness Identification Experiments

Nominal Value	k_x	Left Arm (95% CI)	Error	Right Arm (95% CI)	Error	Linkage (95% CI)	Error
500 N/m	504.0	509.4 (509.6, 509.2)	1.1%	509 (509.4, 508.5)	0.9%	996.6 (997.2, 996)	1.1%
400 N/m	404.0	404.6 (404.8, 404.3)	0.2%	405.1 (405.5, 404.8)	0.3%	801.1 (801.5, 800.7)	0.9%
300 N/m	304.0	303.4 (304, 302.8)	0.2%	308.1 (308.3, 307.8)	1.4%	594.2 (595.2, 593.2)	2.3%

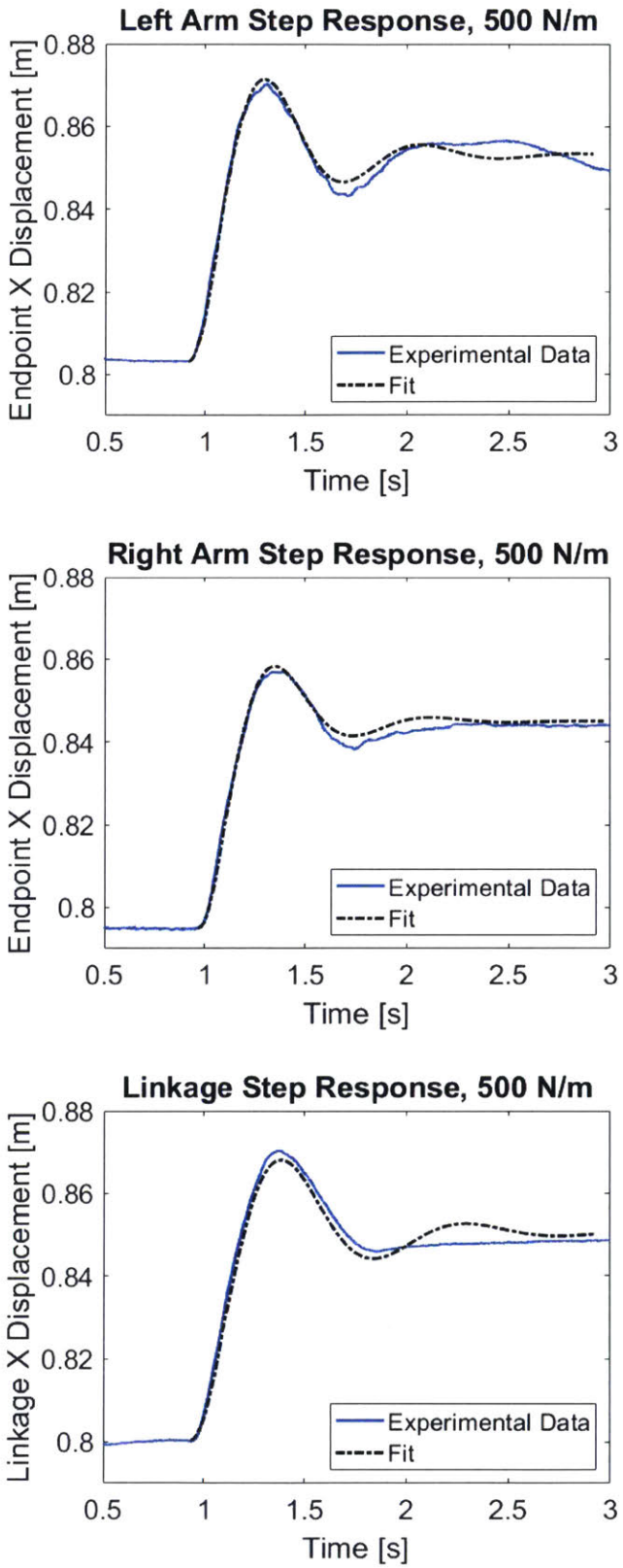


Figure 4-7: Step responses and second order 1-DOF linear model fits for the 500 N/m trials

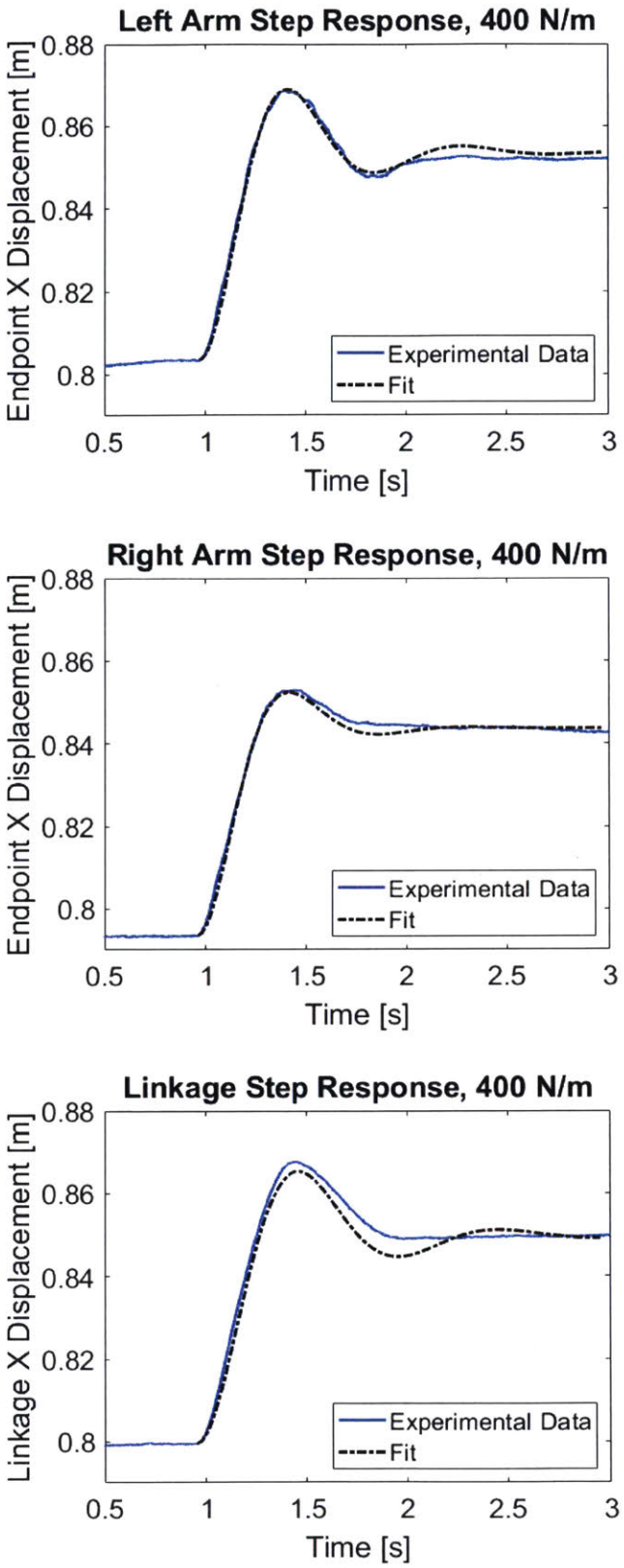


Figure 4-8: Step responses and second order 1-DOF linear model fits for the 400 N/m trials

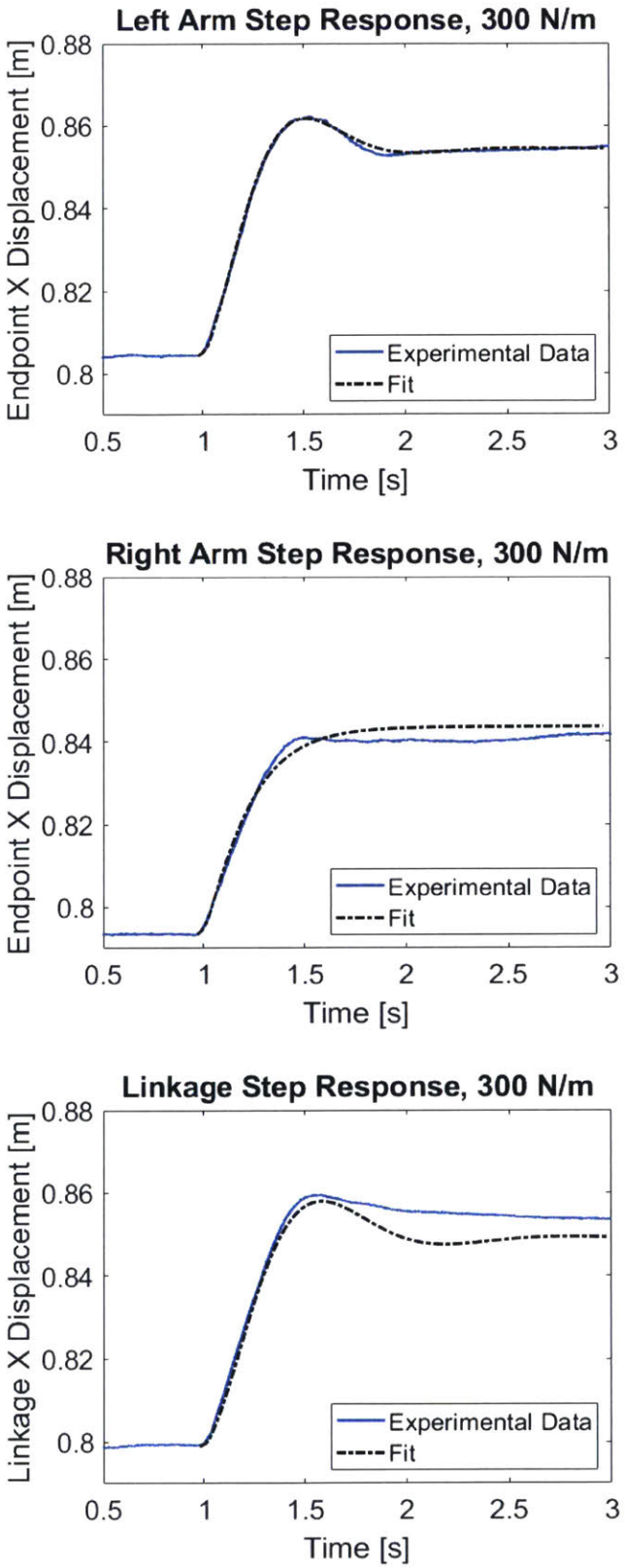


Figure 4-9: Step responses and second order 1-DOF linear model fits for the 300 N/m trials

Table 4.4: Results From Damping Identification Experiments

Trial	Expected b_x	Left Arm	Error	Right Arm	Error	Linkage	Error (From $2b_x$)	Error (From Meas. Sum)
500 N/m	31.4 Ns/m	36.2	15.4%	44.3	41.1%	85.6	36.3%	6.3%
400 N/m	31.4 Ns/m	36.7	16.9%	49.2	56.8%	82.9	32.1%	3.6%
300 N/m	31.4 Ns/m	46.1	47.0%	80.4	156%	96.8	54.1%	23.5 %

Table 4.5: Results From Inertial Parameter Fits

Trial	$m_{x,URDF}$	m_x	Left Arm	Error	Right Arm	Error	Linkage	Error (From $2m_x + m_l$)	Error (From Meas. Sum)
500 N/m	4.40 kg	11.53 kg	6.89	40.1%	6.40	44.3%	18.7	17.6%	41.5%
400 N/m	4.40 kg	11.53 kg	6.81	40.8%	6.31	45.1%	17.9	20.6%	38.0%
300 N/m	4.40 kg	11.53 kg	6.43	44.1%	4.40	61.8%	16.9	25.1%	57.1%

at best a rough approximation meant to make the robot behavior under simulation look comparable to actual robot performance. Nevertheless, for both arms, the values obtained for the 500 N/m and 400 N/m trials were consistent between trials, and the left arm damping was reasonably close to the expected value. For the 500 N/m and 400 N/m trials, the impedance composition error was low ($< 7\%$). Although we were not able to accurately dictate endpoint damping for each arm, the impedances still combined linearly. The parameter estimation at 300 N/m was poor, since (as can be seen from the step-response plots) the 1-DOF linear approximation began to break down at this stiffness level. This is because 300 N/m is too low of a driving stiffness level to rapidly overcome the high levels of unmodelled static friction inherent in the Baxter Robot. Since the linear approximation breaks down, the fitted parameters of a linear model become uninformative.

4.4.2 Inertial Parameter Fits

The apparent mass at the manipulator endpoint was the most challenging parameter to identify. Our controller did not attempt to modify the mass, so we simply aimed to identify the given manipulator endpoint x mass value. The results for the inertia fits are summarized in Table 4.5. Two expected inertial values are shown here. The first, $m_{x,URDF}$, was computed from $\boldsymbol{M}_x$, which excludes measurements of the reflected motor inertias, and relies only on structural data contained in the URDF. The second, m_x, was computed from $\boldsymbol{M}_{x,total}$, which includes reflected motor inertia measurements. The left and right arm columns express the percentage error relative to the expected value of m_x. For the closed chain column, the percentage error is calculated in two ways. The first way was relative to $2m_x + m_l$(to evaluate controller performance), and other way was relative to the sum of the two measured inertia parameters, plus m_l (to evaluate compositionality).

Here, the inertial values deviated significantly from what we would expect. For the individual arms, the measured inertia value was greater than the inertia predicted from the links alone, but less than the inertia predicted from the links and motors combined. The measurements were, however, closer to the former. Regardless of the

accuracy of the robot inertia model, these individual arm inertias did not appear to add linearly when coupled by the link. One possible explanation for this phenomenon is that the upper left element of the endpoint Cartesian space inertia might not correctly predict the 1-DOF behavior of the system. During the step response, although the robot was only commanded along the x coordinate, the robot motion was not confined to solely the x direction. Due to the non-linear nature of the linkages, the off-diagonal terms in the inertia and net stiffness matrix induced slight accelerations in orthogonal endpoint-space directions, which may have affected the apparent mass observed in the x direction. Displacements observed along the y and z directions during the step response are plotted in Appendix B. While the residual motions are negligible along the z direction, there are some residual oscillations along the y coordinate which peaked at 0.8 - 1.5 cm from the nominal y position during the individual arm trials. These motions were mirrored, and peaked outwards, away from the robot's XZ plane. When the linkage step responses were performed, the linkage midpoint did not exhibit any residual motion along the y direction, since the rigid linkage prevented any anti-phase oscillations between the two arms along the y direction. Had the motions been directed along the same direction, they would have also appeared in the linkage responses. This lack of consistency in residual motions may have been an explanation as to why there were errors in the individual arm inertia estimates, and furthermore, why the measured closed-chain configuration inertia did not reflect he sum of the individual arm inertias. Additionally, further examination should be performed into any residual oscillations that may have occurred in the manipulator nullspace, as those may also have impacted the measured inertia at the endpoint.

While this result is disappointing, it is widely acknowledged that obtaining accurate inertia measurements for a robotic manipulator is a challenging task [Dietrich et al., 2015, Nakanishi et al., 2008]. Despite these difficulties, the measurements were consistent between the right and left arms, across the various stiffness values. As before, the parameters derived from the 300 N/m trial are unreliable, especially for those in which the both arms are coupled through the link.

Chapter 5

Sample Manipulation Tasks

5.1 Manipulation Into A Singularity

One of the major advantages of the compositional impedance approach is that each impedance controller does not require any inverse kinematics computations, or Jacobian inversions in the control loop. This enables the manipulator to venture seamlessly into and out of singularities, which is an important prerequisite for the goal of programming robots to be as dexterous and adaptable as humans.

In order to verify that the controller would behave as expected in a manipulator singularity, the robot was first brought to the endpoint and joint-space positions listed in Tables 4.1 and 4.2. Next, the impedance control law given by Eqn. 4.1 was activated on the robot. The equilibrium endpoint position, x_0, was then increased at a rate of 0.02 m/s, to a point which was 17 cm past the edge of the robot's workspace. After pausing at this position, x_0 was then brought back at the same rate to its starting position. This task was then repeated in the closed chain configuration with the linkage installed, and also at a variety of speeds. This task is visualized in Figure 5-1.

At singularity, the Jacobian condition number increased by roughly two orders of magnitude, going from an initial value of about 8.31, to a peak value of 278.4 in the left arm, and 152.1 in the right arm, in the closed chain configuration. However, at the singularity point, the jacobian condition number is extremely sensitive to small

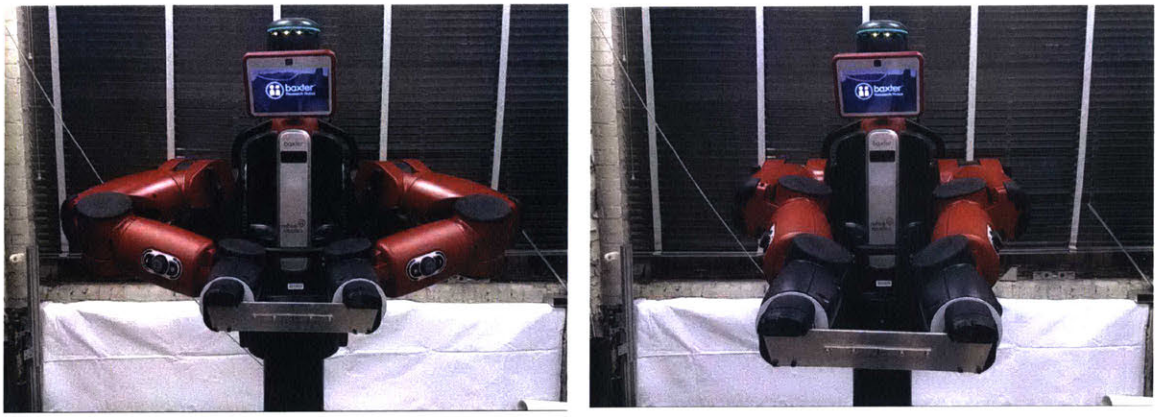


Figure 5-1: The robot's pose at the beginning of the singularity manipulation task, and the robot's pose at singularity, performed in the closed kinematic chain configuration

changes in position. Lightly tugging at the linkage brought the Jacobian condition numbers to values on the order of 10^3. One very critical advantage of introducing an additional joint space impedance is that as the manipulator is brought out of singularity, the joints will always return to their original configurations in a deterministic manner, and not get locked in an alternate, less desirable configuration. Exiting a singularity configuration in a deterministic manner is not a straightforward task to implement using motion control methods which rely on inverse kinematic computations, or Jacobian inversions. Plots of the equilibrium x_0 trajectory along with the actual x trajectory, as well as plots of the Jacobian condition numbers in the closed kinematic chain case, are given in Figures 5-2 and 5-3.

5.2 Complex Task Proof of Concept: Robotic Shoe Shining

One of the promises of the compositional approach to impedance control is the ability to break down a complex task, which involves balancing many task requirements, into a number of sub-tasks, with an impedance controller designed for each one. The example of robotic high-speed dual-arm shoe buffing that was introduced in Section

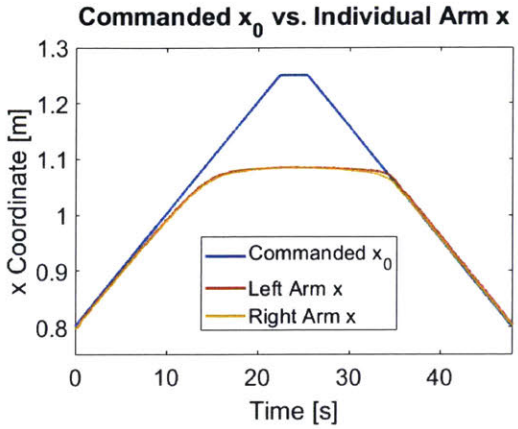


Figure 5-2: The reference trajectory (x_0), and the actual trajectory (x) for the closed kinematic chain singularity manipulation task

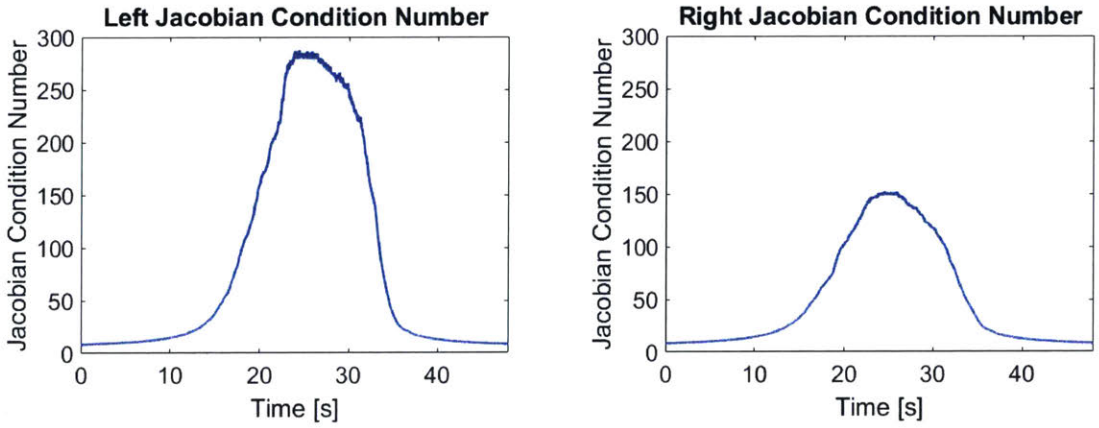


Figure 5-3: The Jacobian condition number of the left arm and right arm during the singularity manipulation task in the closed kinematic chain configuration.

2.5.1 was implemented on the Baxter Robot as a proof of concept for the method.

First the robot arms are brought to a symmetric starting position (Figure 5-4a). Next, the robot arms navigate in free space, under an impedance controller (Controller I), to pick up a slackened cloth at pre-determined pick-points (Figure 5-4b). This impedance controller has damping defined relative to the reference trajectory, as in Eqn. 2.5.

The parameters of Controller I are as follows:

$$\boldsymbol{K}_x = \text{diag}(\begin{bmatrix} 500 & 300 & 500 & 5 & 5 & 5 \end{bmatrix}) \tag{5.1}$$

$$\boldsymbol{B}_x = \text{diag}(\begin{bmatrix} 10 & 10 & 10 & 0.3 & 0.5 & 0.8 \end{bmatrix}) \tag{5.2}$$

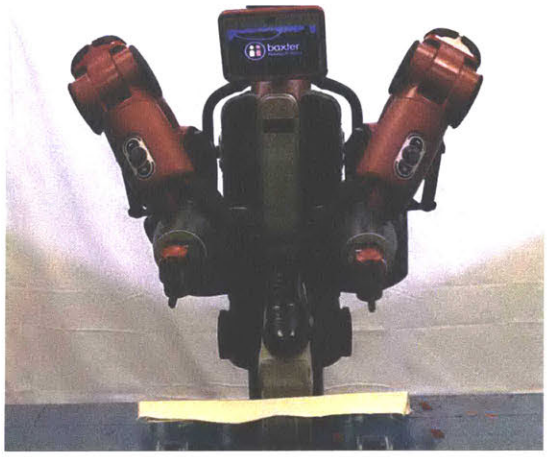
$$\boldsymbol{K}_q = \text{diag}(\begin{bmatrix} 0.5 & 10 & 5 & 0.7 & 0.7 & 0.08 & 0.05 \end{bmatrix}) \tag{5.3}$$

$$\boldsymbol{B}_q = \text{diag}(\begin{bmatrix} 1 & 4 & 2 & 1 & 0.5 & 0.6 & 0.5 \end{bmatrix}) \tag{5.4}$$

The robot then moves this cloth to a position above the shoe (Figure 5-4c). A second endpoint impedance controller (Controller II) is then applied to the robot to tension the cloth. This impedance controller applies 40 N/m virtual springs to both robot endpoints, and stretches them by 10 cm relative to the Controller I reference trajectory, in the $\pm y$ direction (Figure 5-4d). This impedance controller is kept active throughout the duration of the task to ensure tension on the cloth is upheld. As this is a relatively compliant spring, there is little risk of tearing the cloth, or compromising robot's grip on the cloth. An alternative approach that could also be applied is to saturate the output force arising from the Cartesian impedance controller (or from any sum of Cartesian impedance controllers). This is a particularly effective approach when the maximum forces that a manipulated object can tolerate are known.

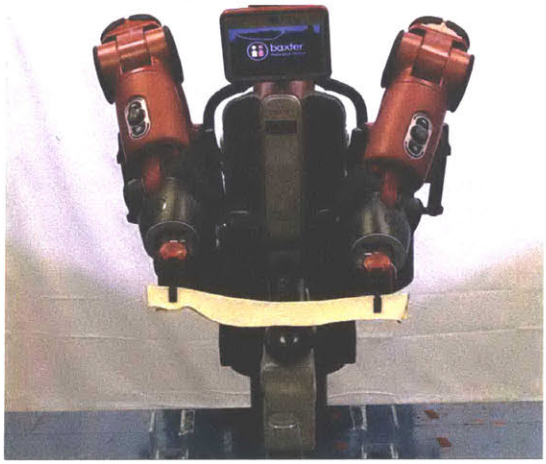
Now, the cloth must be brought into contact with the shoe. The impedance parameters of Controller I are modified mid-task for this purpose, as follows:

$$\boldsymbol{K}_x = \text{diag}(\begin{bmatrix} 200 & 250 & 250 & 5 & 8 & 5 \end{bmatrix}) \tag{5.5}$$

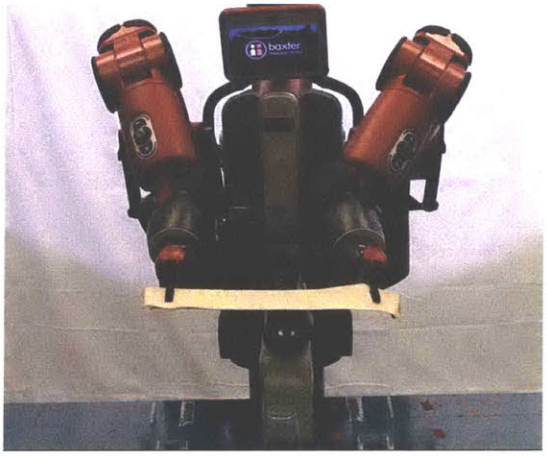


(a) Initial Position

(b) Grabbing Cloth

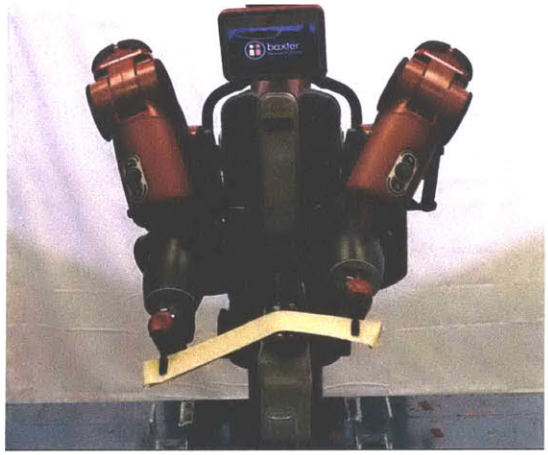


(c) Bringing Cloth Over Shoe



(d) Tensioning Cloth

(e) Bringing Cloth Into Contact With Shoe



(f) Buffing

Figure 5-4: Various stages of the robotic shoe shining task.

$$\boldsymbol{B}_x = \text{diag}(\begin{bmatrix} 10 & 10 & 10 & 0.3 & 0.5 & 0.8 \end{bmatrix}) \tag{5.6}$$

$$\boldsymbol{K}_q = \text{diag}(\begin{bmatrix} 0.005 & 12 & 5 & 0.005 & 0.005 & 0.005 & 0.005 \end{bmatrix}) \tag{5.7}$$

$$\boldsymbol{B}_q = \text{diag}(\begin{bmatrix} 1 & 4 & 2 & 1 & 0.6 & 0.9 & 0.9 \end{bmatrix}) \tag{5.8}$$

In this set of controller parameters, more compliance was added to the x, y, and z components of $\boldsymbol{K}_x$ in anticipation of the contact task. Additional stiffness was added in one of the rotational components of $\boldsymbol{K}_x$ to brace for the rapid motions. More stiffness was added to the S1 and E0 joints in $\boldsymbol{K}_q$ to correct for joint-space drift due to imperfections in the gravity model (which become slightly more apparent due to the increased compliance at the endpoint). Additionally, more damping was added to the wrist joints in $\boldsymbol{B}_q$ to ensure smoother movements.

The equilibrium position of Controller I was then brought downwards, such that the cloth comes into contact with the shoe, and maintains an appropriate normal force (Figure 5-4e). Since there is added compliance under the new impedance parameters, larger errors in manipulator positioning can be tolerated. This allows us to forgo any detailed modeling of the cloth contact shape, and robot-cloth contact points. Finally, the robot begins moving the equilibrium trajectory, $\boldsymbol{x}_0$, such that it undergoes a 1.25 Hz rhythmic oscillation in the z-y plane (Figure 5-4f). This virtual trajectory motion has an amplitude of 0.07 m, at an angle of 85° relative to the robot's y axis. Once again, the built in compliance removes the need to model cloth dynamics such as nonlinear stretching, or contact friction forces. Relying on manipulator compliance to forego detailed modelling of the manipulated object is of great benefit for many flexible object manipulation tasks. Flexible objects such as paper, ropes, or cloths are very difficult to model, since their complete states may never be fully known, and finite element approximations of the dynamics tend to be computationally taxing [Yamakawa et al., 2011, Bai et al., 2016, Hopcroft et al., 1991, Miller et al., 2011, Balkcom and Mason, 2008].

Additionally, this controller is robust to large disturbances. Large forces can be applied to the robot arms, the cloth can be ripped out of the robot's grippers, and the

shoe can even be removed from the workspace without any loss of controller stability. These simple motions, all planned in endpoint-space, can yield intelligent, stable, and robust cooperation of 14 degrees of freedom to accomplish a high-speed closed-chain contact manipulation task.

One shortcoming of this implementation is that the manipulation is carried out without any higher order perception control loop. While the controller will remain stable in the face of disturbances, the robot cannot adjust for any changes in the manipulation task while it is underway. This framework could be combined with visual perception to close a higher-level task supervisory loop for more advanced applications.

Chapter 6

Nullspace Projections: An Impedance Based Interpretation

For redundant manipulators (i.e. when the the number of robot degrees of freedom, n, is greater than the number of endpoint-space coordinates, m), this work has emphasized using two impedance control laws in parallel, as in Eqns. 2.5 and 4.1. In these controllers, an endpoint-space impedance controller was specified for the task, and a second joint-space impedance controller with relatively small stiffness and damping specified to control the $n - m$ redundant degrees of freedom. There are two major problems with this approach. The first problem is that adding on this extra joint-space stiffness and damping will distort the desired endpoint-space stiffness and damping matrices. The net endpoint stiffness and damping matrices will have elevated diagonal values, and many introduced off-diagonal values, compared to the specified endpoint-space matrices. This can be observed by comparing Eqns. 4.11, 4.12, 4.13, and 4.15 to Eqns. A.1, A.2, A.3, and A.4. The second problem is that as the manipulator moves away from its starting equilibrium position (where $\boldsymbol{x} = \boldsymbol{x_0} = L(\boldsymbol{q}_0)$), the added joint-space stiffness, $\boldsymbol{K}_q$, will always work to drive $\boldsymbol{q}$ and $\boldsymbol{x}$ to their starting positions of $\boldsymbol{q}_0$ and $L(\boldsymbol{q}_0)$, respectively. This may add sizable and troublesome steady state errors to positioning tasks performed in a large portion of the manipulator's workspace.

In order to counteract these negative effects, the controller must delegate com-

plete control of all m endpoint degrees of freedom to the endpoint-space impedance controller, while also only delegating the $n - m$ redundant degrees of freedom in the joint-space to the joint-space controller. No torques arising from the added joint-space controller should interfere with tasks performed in the endpoint-space of the robot.

One should also note that a secondary controller operating in the $n-m$ redundant degrees of freedom does not need to be a simple impedance around a nominal set of joint angles. Examples of other secondary tasks that would control the redundant degrees of freedom in a robot might be managing elbow contact with the environment, collision avoidance, or endpoint mass matrix optimization. For this work, however, we will focus on the example of a simple additional joint space impedance controller.

6.1 The Traditional Approach

The traditional approach to preventing a secondary controller acting in the redundant degrees of the joint space from interfering in the endpoint degrees of freedom is to use a nullspace projection matrix [Dietrich et al., 2015, Khatib, 1987, Khatib, 1995, Siciliano and Slotine, 1991]. The general idea is as follows:

The relationship between the generalized forces applied to the manpulator's endpoint and the torques felt at its joints are given by Eqn. 1.14. We can non-uniquely invert this relationship by taking the generalized inverse of the $m \times n$ Jacobian:

$$\boldsymbol{f} = \boldsymbol{J}^{\#T}\boldsymbol{\tau} \tag{6.1}$$

where $\boldsymbol{J}^{\#T}$ represents the transpose of the generalized inverse of $\boldsymbol{J}$. One formula for this generalized inverse is:

$$\boldsymbol{J}^{\#} = \boldsymbol{W}\boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{W}\boldsymbol{J}^T)^{-1} \tag{6.2}$$

where $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ is an invertable weighting matrix. If $\boldsymbol{W}$ is equal to the identity matrix, $\boldsymbol{I}$, then $\boldsymbol{J}^{\#}$ is the Moore-Penrose pseudoinverse of the Jacobian, denoted by

$\boldsymbol{J}^+$.

If we take a vector of arbitrary joint torques and project them into the nullspace of $\boldsymbol{J}^{\#T}$, we can see from (6.1) that these projected torques will produce no net force at the manipulator endpoint. We can implement this nullspace projection by premultiplying our arbitrary torque vector by the following matrix [Dietrich et al., 2015]:

$$\boldsymbol{N} = \left(\boldsymbol{I} - \boldsymbol{J}^T(\boldsymbol{J}^{\#})^T\right) \tag{6.3}$$

where $\boldsymbol{N} \in \mathbb{R}^{n \times n}$.

For simplicity, let us consider our usual controller, but with only the stiffness terms (the damping terms can be incorporated into this framework trivially):

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\left[\boldsymbol{K}_x(\boldsymbol{x_0} - \boldsymbol{x})\right] + \boldsymbol{K}_q(\boldsymbol{q}_0 - \boldsymbol{q}) \tag{6.4}$$

If we apply the nullspace projection matrix to the joint-space impedance controller, we have:

$$\boldsymbol{\tau}_{act} = \boldsymbol{J}^T\left[\boldsymbol{K}_x(\boldsymbol{x_0} - \boldsymbol{x})\right] + \boldsymbol{N}\boldsymbol{K}_q(\boldsymbol{q}_0 - \boldsymbol{q}) \tag{6.5}$$

This controller will now exert a restoring stiffness on the $n - m$ degrees of freedom in the manipulator nullspace, without any impact at all on the endpoint-space impedance. For this controller, $\boldsymbol{K}_{x,net} = \boldsymbol{K}_x$. This is an extremely effective approach while the robot is well inside its workspace, and has a Jacobian of full row rank (i.e. the rank of $\boldsymbol{J}$ is m). However, the behavior of the controller at singularity must be analyzed. If we use the formula given in (6.2) to compute $\boldsymbol{J}^{\#}$, we can see that at singularity points, when the rank of $\boldsymbol{J}$ drops, the matrix $(\boldsymbol{J}\boldsymbol{W}\boldsymbol{J}^T)$ becomes rank deficient, which causes the inverse $(\boldsymbol{J}\boldsymbol{W}\boldsymbol{J}^T)^{-1}$ to become unbounded. For the following analysis, the variable r will be used to refer to the rank of $\boldsymbol{J}$.

A more sophisticated (and perhaps robust) way to take a generalized inverse of $\boldsymbol{J}$ is to use singular value decomposition (SVD) to obtain the Moore-Penrose pseudoinverse of $\boldsymbol{J}$. This algorithm is used in many numerical software packages such as MATLAB, NumPy, and GNU Octave [MathWorks, 2019b, The SciPy Community, 2018, Eaton,

2015].

First, $\boldsymbol{J}$ is factored using SVD:

$$\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{6.6}$$

For the case where $\boldsymbol{J}$ is of rank m, $\boldsymbol{U} \in \mathbb{R}^{m \times m}$ is the matrix whose columns are the eigenvectors of $\boldsymbol{J}\boldsymbol{J}^T$, and $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ is the matrix whose columns are the eigenvectors of $\boldsymbol{J}^T\boldsymbol{J}$. $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is a diagonal vector of singular values, which are the square roots of the eigenvalues of both $\boldsymbol{J}\boldsymbol{J}^T$ and $\boldsymbol{J}^T\boldsymbol{J}$. $\boldsymbol{\Sigma}$ also has $n - m$ columns of zeros. This structure can be visualized as follows:

$$\boldsymbol{J} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_1 & \boldsymbol{u}_2 & \dots & \boldsymbol{u}_m \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & & & 0 & \cdots & 0 \\ & \ddots & & & \vdots & \cdots & \vdots \\ & & \sigma_{m-1} & & \vdots & \cdots & \vdots \\ & & & \sigma_m & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{v}_2 & — \\ & \vdots & \\ — & \boldsymbol{v}_n & — \end{bmatrix} \tag{6.7}$$

The Moore-Penrose pseudoinverse of $\boldsymbol{J}$ can now be expressed as [Strang, 2016]:

$$\boldsymbol{J}^+ = \boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^T \tag{6.8}$$

The pseudoinverse of the diagonal matrix is defined as taking the reciprocal of all non-zero diagonal entries in the matrix. This equation can be visualized as:

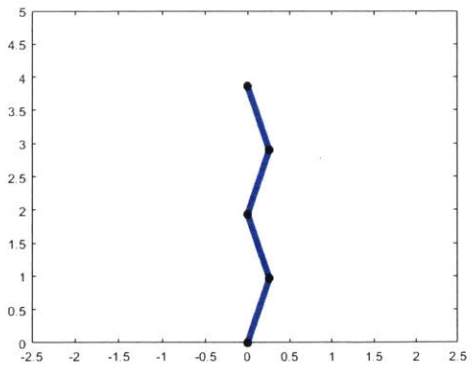
$$\boldsymbol{J}^+ = \begin{bmatrix} | & | & & | \\ \boldsymbol{v}_1 & \boldsymbol{v}_2 & \dots & \boldsymbol{v}_m \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1^{-1} & & & & 0 & \cdots & 0 \\ & \ddots & & & \vdots & \cdots & \vdots \\ & & \sigma_{m-1}^{-1} & & \vdots & \cdots & \vdots \\ & & & \sigma_m^{-1} & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} — & \boldsymbol{u}_1 & — \\ — & \boldsymbol{u}_2 & — \\ & \vdots & \\ — & \boldsymbol{u}_n & — \end{bmatrix} \tag{6.9}$$

When the robot reaches a singular configuration, $\boldsymbol{J}$ loses rank such that $r < m$. In this configuration, the last $m - r$ singular values on the main diagonal of $\boldsymbol{\Sigma}$ will go to zero, and the corresponding vectors in $\boldsymbol{U}$ and $\boldsymbol{V}$ will become nullspace basis vectors of $\boldsymbol{J}$. Now, when we take the Moore-Penrose pseudoinverse of $\boldsymbol{\Sigma}$ in Eqn. 6.8,

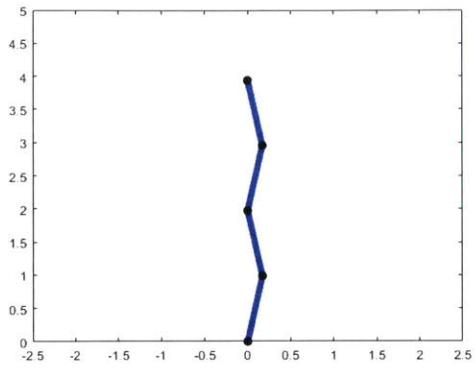
we ignore these zero values on the diagonal[1], while taking the reciprocals of all the non-zero diagonal entries. This leaves us with a pseudoinverse matrix that is bounded at manipulator singularities [Strang, 2016].

While the behavior of the pseudoinverse *at* manipulator singularities is well behaved, the Jacobian pseudoinverse when *approaching* singularities needs more analysis. When $\boldsymbol{J}$ approaches a singularity, and goes from rank m to, for instance, rank $m-1$, the singular value σ_m in $\boldsymbol{\Sigma}$ continuously approaches zero. When this value is close to zero, but not quite zero, the reciprocal, σ_m^{-1}, tends towards infinity. In fact, this reciprocal becomes by far the dominant singular value in $\boldsymbol{J}^+$, and thus, $\|\boldsymbol{J}^+\|$ tends towards infinity. However, once the singular value reaches zero, it is ignored, and $\|\boldsymbol{J}^+\|$ immediately drops down to a reasonable value. The way this behavior is practically implemented in floating-point computer arithmetic is that σ_m is checked against a threshold, and if it is below that threshold, it is set to zero [MathWorks, 2019b, The SciPy Community, 2018, Eaton, 2015]. In MATLAB for instance, this threshold, by default, is set to: $\max(m,n) \times \text{eps}(\sigma_{max})$, where $\text{eps}(\cdot)$ gives the distance from its argument to the next largest floating-point double-precision number [MathWorks, 2019a]. Essentially, this is the computer's double-precision floating-point resolution around a given number (also known as the machine epsilon). NumPy and Octave have similar zero-cutoff tolerance schemes. To get a sense of the typical magnitudes, in MATLAB, $\text{eps}(5.0) = 8.8816 \times 10^{-16}$. One should note that the SVD zero cutoff tolerance can be manually set higher for the purposes of pseudoinversion. This Jacobian pseudoinverse ill-conditioning is also the reason behind why motion control schemes based on inverse kinematics and Jacobian inversion fail to operate into singularities.

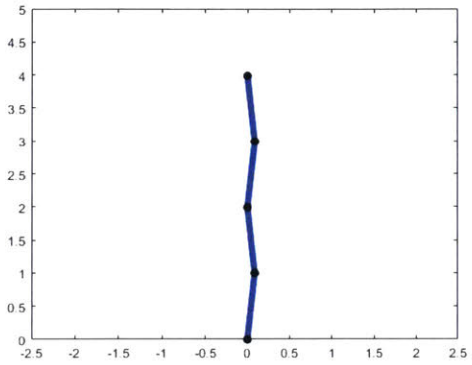
[1]If ignoring zero values on the diagonal of $\boldsymbol{\Sigma}$ bothers the reader, one can visualize truncating the original SVD matrices (Eqns. 6.6, 6.7) to get the same result. If we perform an SVD on a rank deficient Jacobian ($r < m$), the last $m - r$ rows of Σ will consist entirely of zeros. We can truncate these three matrices by removing the zero rows and columns in Σ, along with the corresponding nullspace basis vectors in $\boldsymbol{U}$ and $\boldsymbol{V}$. Now, $\boldsymbol{U} \in \mathbb{R}^{m \times r}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{r \times r}$, and $\boldsymbol{V}^T \in \mathbb{R}^{r \times n}$. If we multiply these truncated matrices, we will obtain the same $\boldsymbol{J}$ matrix as our product. All we have done is removed the rows and columns of the matrices that would have multiplied out to zero in the SVD. Now, we can perform the pseudoinversion in 6.8 on these truncated matrices without ignoring any diagonal values in Σ. See pp. 371-372 and pp. 395-396 of [Strang, 2016].



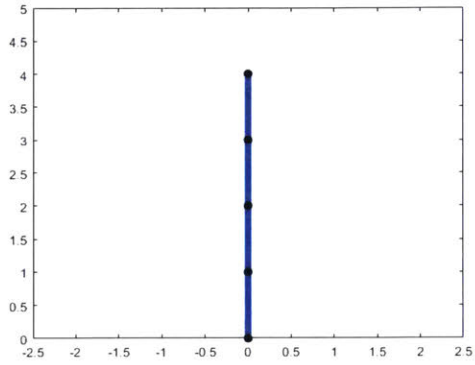
(a) $\theta = 30°$, $\boldsymbol{q} = [75, 30, -30, 30]^T$



(b) $\theta = 20°$, $\boldsymbol{q} = [80, 20, -20, 20]^T$



(c) $\theta = 10°$, $\boldsymbol{q} = [85, 10, -10, 10]^T$



(d) $\theta = 0°$, $\boldsymbol{q} = [90, 0, 0, 0]^T$

Figure 6-1: Four-link planar robot approaching a kinematic singularity.

To visualize this discontinuity in $\boldsymbol{J}^+$, a simple numerical experiment was performed in MATLAB. The kinematics of a four link planar manipulator, of unit link lengths was simulated. The links were assigned joint angles (in degrees) of $\boldsymbol{q} = \left[90 - \frac{\theta}{2}, \quad \theta, \quad -\theta, \quad \theta\right]^T$, and the parameter θ was varied form 30° to 0° with the resolution of a 16-bit encoder. The manipulator reaches singularity when θ reaches zero. This setup with a few sample values of θ is shown in Fig. 6-1.

As the manipulator reaches the singularity point, the Jacobian condition number becomes unbounded, and the norm of the pseudoinverse rapidly increases, as seen in Fig. 6-2 (note the semi-log scale). In Fig. 6-3, we can see that during this process, the norm of the Jacobian pseudoinverse also rapidly rises (note the semi-log scale). At $\theta = 30°$, $\|\boldsymbol{J}^+\| = 4.481$. With a 16-bit encoder angular resolution and the default pseudoinverse singular value zero cutoff tolerance on MATLAB, $\|\boldsymbol{J}^+\|$ peaks

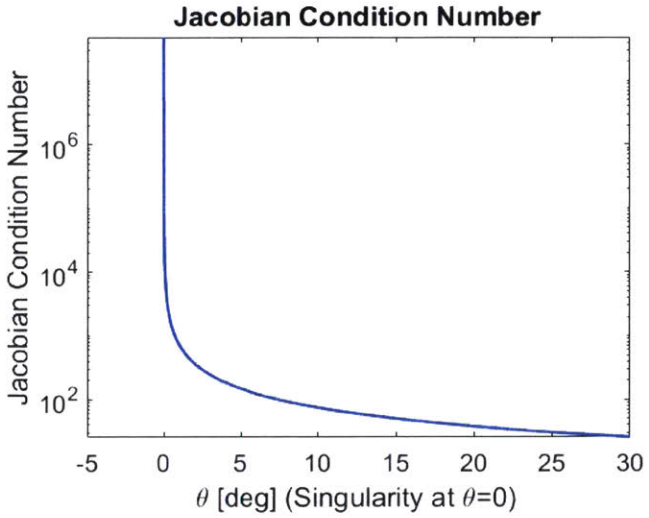


Figure 6-2: The Jacobian condition number. Note the semi-log scale.

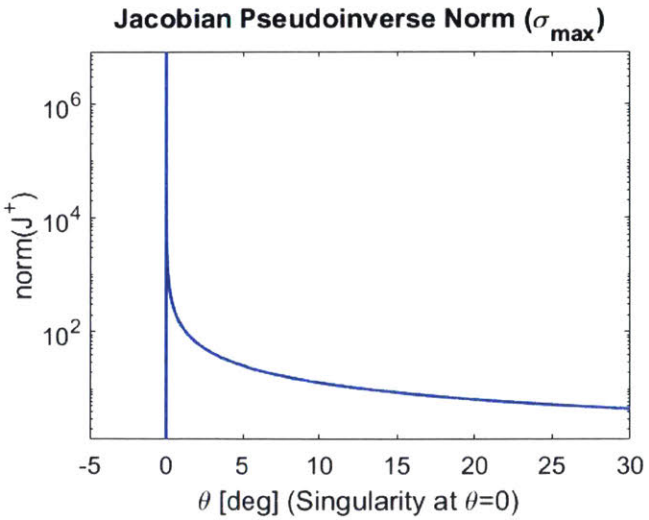


Figure 6-3: The norm of the Moore-Penrose pseudoinverse of $\boldsymbol{J}$, obtained from the SVD method on MATLAB, with default cutoff tolerances. Note the semi-log scale.

at 8.379×10^6. If we were to increase the angular resolution, $\|\boldsymbol{J}^+\|_{max}$ would increase correspondingly, since we would compute $\|\boldsymbol{J}^+\|$ with $\sigma_{min}(\boldsymbol{J})$ at values increasingly to machine epsilon. Once we reach a value of $\sigma_{min}(\boldsymbol{J}) = \max(m, n) \times \text{eps}(\sigma_{max}(\boldsymbol{J}))$, the computer stops taking the reciprocal of $\sigma_{min}(\boldsymbol{J})$, and simply sets it to zero. This causes $\|\boldsymbol{J}^+\|$ to instantaneously drop to a steady state singularity value of 1.292, which is well below the 4.481 value of $\|\boldsymbol{J}^+\|$ at $\theta = 30°$.

While the $\boldsymbol{J}^+$ matrix becomes very badly scaled while approaching singularity, these scaling issues do not appear in the output nullspace projection matrix given in Eqn. 6.3. Instead, the singularity behavior manifests itself only as a step change in the values of the nullspace projection matrix once singularity is reached. In order to understand this behavior better, we can perform an SVD of each term in Eqn. 6.3 by substituting in Eqns. 6.6 and 6.8 [Dietrich et al., 2012]:

$$\boldsymbol{N} = \left(\boldsymbol{I} - (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^T(\boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^T)^T\right) \tag{6.10}$$

$$\boldsymbol{N} = \left(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{\Sigma}^T\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{\Sigma}^{+T}\boldsymbol{V}^T\right) \tag{6.11}$$

Since U is orthogonal, we have:

$$\boldsymbol{N} = \left(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}^{+T}\boldsymbol{V}^T\right) \tag{6.12}$$

Since the diagonal values of $\boldsymbol{\Sigma}^{+T}$ are the reciprocal values of $\boldsymbol{\Sigma}^T$, the term $\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}^{+T}$ becomes a square diagonal matrix of size $n \times n$ containing entries of ones and zeros [Dietrich et al., 2012]. We can call this matrix $\boldsymbol{S}$:

$$\boldsymbol{N} = \left(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{S}\boldsymbol{V}^T\right) \tag{6.13}$$

Essentially, each singular value is divided by itself to yield unity, no matter how badly scaled each singular value is. If a particular singular value nears the machine epsilon and is ignored by the pseudoinversion algorithm, the corresponding diagonal value in $\boldsymbol{S}$ instantaneously becomes zero. This leads to a discontinuous step in actuator

torques once singularity is reached in a robot. In terms of the usual implementation of $\boldsymbol{N}$ as given in Eqn. 6.3, we can see that all very large entries in $\boldsymbol{J}^{\#}$ are multiplied by a correspondingly small value in $\boldsymbol{J}$ to yield a well-scaled output matrix $\boldsymbol{N}$. Several techniques have been implemented to limit the sharp step change in torques that come about from ones flipping to zeros in $\boldsymbol{S}$ at singularity [Dietrich et al., 2012, Dietrich, 2016, Chiaverini, 1997, Deo and Walker, 1995].

6.2 An Impedance Based Approach

The fundamental problem that led us to use an additional joint-space stiffness matrix, $\boldsymbol{K}_q$, is that $\boldsymbol{K}_x$ was insufficient to apply a restoring stiffness to all n degrees of freedom in the manipulator. We can see this more clearly by directly reflecting this endpoint-space stiffness matrix into the joint-space. Ignoring any kinematic stiffness terms, like that in Eqn. 2.7 (i.e. assuming zero endpoint force), the equivalent joint-space stiffness matrix arising from an endpoint stiffness matrix is given by [Mussa-Ivaldi and Hogan, 1991]:

$$\boldsymbol{K}_\theta = \boldsymbol{J}^T \boldsymbol{K}_x \boldsymbol{J} \tag{6.14}$$

Here, $\boldsymbol{J}$ is of rank $r \leq m < n$. Since $\boldsymbol{K}_x \in \mathbb{R}^{m \times m}$ with rank m, we know $\boldsymbol{K}_\theta \in \mathbb{R}^{n \times n}$ with at most rank $r < n$. This means that the joint-space reflection of our desired endpoint stiffness matrix has an $n-r$ dimensional nullspace. We previously attempted to control these nullspace motions by superimposing a full rank n joint-space stiffness matrix. A superior approach might be to craft a rank $n-r$ joint-space stiffness matrix which spans the nullspace of $\boldsymbol{K}_x$ reflected into the joint-space. This would control the redundant degrees of freedom without impacting the endpoint-space interactive properties.

First, an eigenvalue decomposition is performed on the reflection our endpoint stiffness matrix into joint-space ($\boldsymbol{K}_\theta$):

$$\boldsymbol{K}_\theta = \boldsymbol{V} \boldsymbol{\Lambda} \boldsymbol{V}^{-1} \tag{6.15}$$

Here, $\boldsymbol{V}$ is a matrix whose columns are the eigenvectors of $\boldsymbol{K}_\theta$, and $\boldsymbol{\Lambda}$ is a diagonal matrix of eigenvalues of $\boldsymbol{K}_\theta$. If we chose $\boldsymbol{K}_x$ to be symmetric, $\boldsymbol{K}_\theta$ is also symmetric, which means that $\boldsymbol{V}$ is orthogonal, and all the eigenvalues are real. Therefore, $\boldsymbol{V}^{-1} = \boldsymbol{V}^T$, and we are left with:

$$\boldsymbol{K}_\theta = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T \tag{6.16}$$

In the case where the Jacobian is full row rank (corresponding to conditions where the Jacobian is well within its dexterous workspace), we have $r = m$. The structure of these matrices in this case are as follows:

$$\boldsymbol{K}_\theta = \begin{bmatrix} | & | & & | \\ \boldsymbol{v}_1 & \boldsymbol{v}_2 & \dots & \boldsymbol{v}_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} 0 & & & 0 & \cdots & 0 \\ & \ddots & & & \ddots & \vdots \\ & & 0 & & & 0 \\ 0 & & & \lambda_1 & & \\ \vdots & \ddots & & & \ddots & \\ 0 & \cdots & 0 & & & \lambda_m \end{bmatrix} \begin{bmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{v}_2 & — \\ & \vdots & \\ — & \boldsymbol{v}_n & — \end{bmatrix} \tag{6.17}$$

In the diagonal matrix, we have $(n-m)$ diagonal zero entries (which always remain zero), and m non-zero eigenvalues. In this decomposition, the nullspace properties become apparent. The nullspace dimension is $(n-m)$, and the first $(n-m)$ eigenvectors are the nullspace basis vectors. The m non-zero eigenvalues and eigenvectors multiply to give us our original $\boldsymbol{K}_\theta$ matrix.

From Eqn. 6.17, we can see how to fill in the nullspace. We can replace all $(n-m)$ zeros on the diagonal with arbitrary real constants, k_1 ... k_{n-m}. We can multiply this out to obtain a full rank version $\boldsymbol{K}_\theta$, which will be called $\boldsymbol{K}_{\theta,Full}$. The eigenstructure

is now:

$$\boldsymbol{K}_{\theta,Full} = \begin{bmatrix} | & | & & | \\ \boldsymbol{v}_1 & \boldsymbol{v}_2 & \dots & \boldsymbol{v}_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} k_1 & & & 0 & \cdots & 0 \\ & \ddots & & & \ddots & \vdots \\ & & k_{n-m} & & & 0 \\ 0 & & & \lambda_1 & & \\ \vdots & \ddots & & & \ddots & \\ 0 & \cdots & 0 & & & \lambda_m \end{bmatrix} \begin{bmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{v}_2 & — \\ & \vdots & \\ — & \boldsymbol{v}_n & — \end{bmatrix} \tag{6.18}$$

These arbitrary constants, k_i, correspond to stiffnesses that will provide a restoring force to any perturbations in the manipulator nullspace. This ensures that $\boldsymbol{K}_{\theta,Full}$ is full rank, and thus, the robot will not have any uncontrolled nullspace motions. Now, we can break up this full-rank joint space stiffness matrix into a rank m endpoint stiffness matrix, and a rank $n-m$ joint-space stiffness matrix that operates in the nullspace of the joint-space stiffness matrix.

To examine how this might be done, we could treat the matrix multiplication in Eqn. 6.18 as a sum of n rank 1 matrices of size $n \times n$. This allows us to break down $\boldsymbol{K}_{\theta,Full}$ into an original $\boldsymbol{K}_\theta$ component, and a nullspace-filling $\boldsymbol{K}_{\theta,Null}$ component:

$$\begin{aligned} \boldsymbol{K}_{\theta,Full} = \boldsymbol{K}_{\theta,Null} + \boldsymbol{K}_\theta = & \begin{bmatrix} | \\ \boldsymbol{v}_1 \\ | \end{bmatrix} k_1 \begin{bmatrix} — & \boldsymbol{v}_1 & — \end{bmatrix} + \dots + \begin{bmatrix} | \\ \boldsymbol{v}_{n-m} \\ | \end{bmatrix} k_{n-m} \begin{bmatrix} — & \boldsymbol{v}_{n-m} & — \end{bmatrix} \\ & + \begin{bmatrix} | \\ \boldsymbol{v}_{n-m+1} \\ | \end{bmatrix} \lambda_1 \begin{bmatrix} — & \boldsymbol{v}_{n-m+1} & — \end{bmatrix} + \dots + \begin{bmatrix} | \\ \boldsymbol{v}_n \\ | \end{bmatrix} \lambda_m \begin{bmatrix} — & \boldsymbol{v}_n & — \end{bmatrix} \end{aligned} \tag{6.19}$$

This summation can be written more compactly as:

$$\boldsymbol{K}_{\theta,Full} = \boldsymbol{K}_{\theta,Null} + \boldsymbol{K}_\theta = \sum_{i=1}^{n-m} \boldsymbol{v}_i k_i \boldsymbol{v}_i^T + \sum_{i=(n-m+1)}^{n} \boldsymbol{v}_i \lambda_i \boldsymbol{v}_i^T \tag{6.20}$$

Thus, the first $n-m$ terms in the sum encode the filled nullspace portion of the

matrix, and the last m terms in the sum represent the original $\boldsymbol{K}_\theta$ matrix, which encodes the original $\boldsymbol{K}_x$ matrix. Thus, we can implement the endpoint-space portion of the original controller (Eqn. 6.4), and add on an additional nullspace filling matrix:

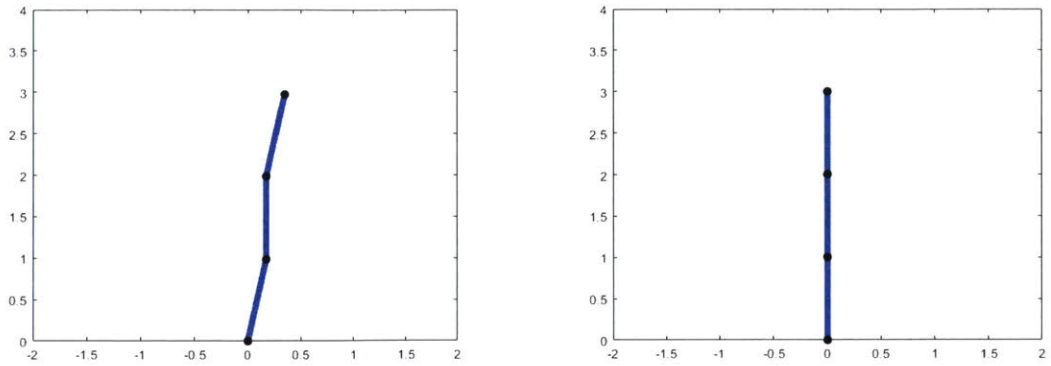
$$\boldsymbol{\tau} = \boldsymbol{J}^T \boldsymbol{K}_x (\boldsymbol{x}_0 - \boldsymbol{x}) + \boldsymbol{K}_{\theta,Null} (\boldsymbol{q}_0 - \boldsymbol{q}) \tag{6.21}$$

This controller gives us the proper restoring torques (parametrized by $k_1 \ldots k_n$) without interfering with the desired endpoint behavior. As a simple verification, in the case of a well conditioned Jacobian, if one computes $\boldsymbol{K}_{\theta,Null}$ from Eqn. 6.21 with setting all arbitrary k values to unity, we will obtain the same matrix as $\boldsymbol{N}\boldsymbol{K}_q$ from Eqn. 6.5 if we set $\boldsymbol{W} = \boldsymbol{I}$ (corresponding to $\boldsymbol{J}^\# = \boldsymbol{J}^+$) and $\boldsymbol{K}_q = \boldsymbol{I}$.

6.3 Comparison of Projection Behavior at Singularity

In order to visualize the differences between the traditional and impedance based nullspace projection approaches, a series of numerical experiments were performed on a three link planar robot, with unit link lengths. The enpoint coordinates of this robot were considered to be $\boldsymbol{x} = \begin{bmatrix} x, & y \end{bmatrix}^T$, which meant that the robot had one degree of redundancy, since $m = 2$ and $n = 3$. The robot was assigned joint angles $\boldsymbol{q} = \begin{bmatrix} 90 - \theta, & \theta, & -2\theta \end{bmatrix}^T$, and the parameter θ was varied form 10° to 0°. This manipulator is visualized in Figure 6-4 in its initial configuration, and at a singular configuration.

For the purposes of computing the projected nullspace torques using the traditional approach, a joint space stiffness of $\boldsymbol{K}_q = \boldsymbol{I}$ was assigned to the manipulator. This stiffness was associated with a nominal position $\boldsymbol{q}_0 = [80, 10, -20]^T$. This corresponded to the initial state of the parameter θ. For the impedance based approach, an endpoint stiffness of $\boldsymbol{K}_x = \text{diag}(\begin{bmatrix} 10, & 10 \end{bmatrix})$ was assigned to the manipulator.



(a) $\theta = 10°$, $\boldsymbol{q} = [80, 10, -20]^T$ (b) $\theta = 0°$, $\boldsymbol{q} = [90, 0, 0]^T$

Figure 6-4: Three link planar robot in its initial configuration, and at singularity.

6.3.1 Traditional Approach

The nullspace projected torques given by the second term of Equation 6.5, with the generalized inverse in $\boldsymbol{N}$ (Equation 6.3) being the Moore-Penrose pseudoinverse carried out with MATLAB's default zero singular value threshold (which in this case was 1.33×10^{-15}). These nullspace projected torques are plotted in Figure 6-5.

This plot illustrates the step discontinuity in torques that occurs at the kinematic singularity, due to ones becoming zeros on the main diagonal of $\boldsymbol{S}$. If the zero singular value threshold for computing the Moore-Penrose psuedoinverse of $\boldsymbol{J}$ is increased, the discontinuity will still occur, but will be shifted away from $\theta = 0$. An example of this, with the threshold set at 0.005 is shown in Figure 6-6. Here, the final torque values were identical to the plot with the default threshold, but the discontinuity was shifted to $\theta = 0.297$. As mentioned previously, there are many methods in the literature for limiting the sharp step change in torques that arise from this approach, but a detailed comparison with these methods is outside the scope of the present work [Dietrich et al., 2012, Dietrich, 2016, Chiaverini, 1997, Deo and Walker, 1995].

6.3.2 Impedance Based Approach

In the impedance based approach, we first examine the eigenvalues of $\boldsymbol{K}_\theta$, which is the reflection of the endpoint-space stiffness matrix, $\boldsymbol{K}_x$, into the joint-space. For a

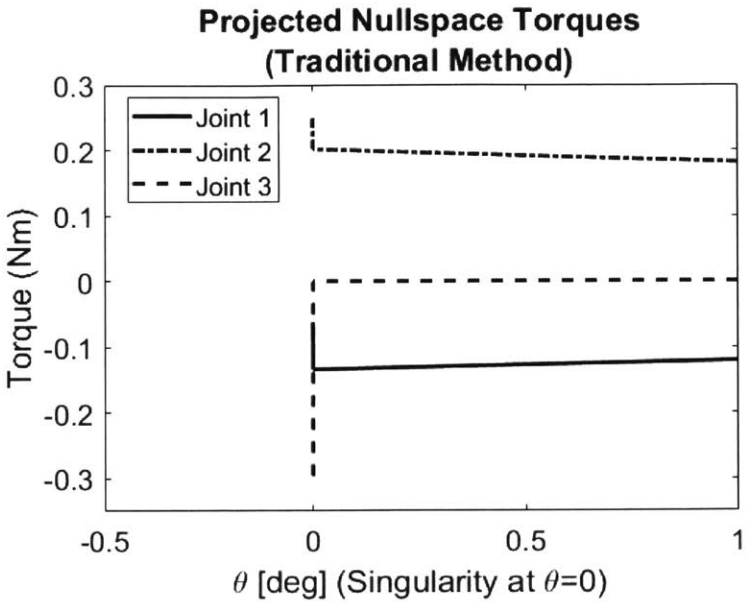


Figure 6-5: Nullspace projected torques along each joint, generated from the traditional method (the second term of Equation 6.5). Only the last degree of θ is plotted. While a vertical line connects the singularity point to the non-singularity curves in this plot, this step is discontinuous. The data points were plotted with solid lines for visual clarity.

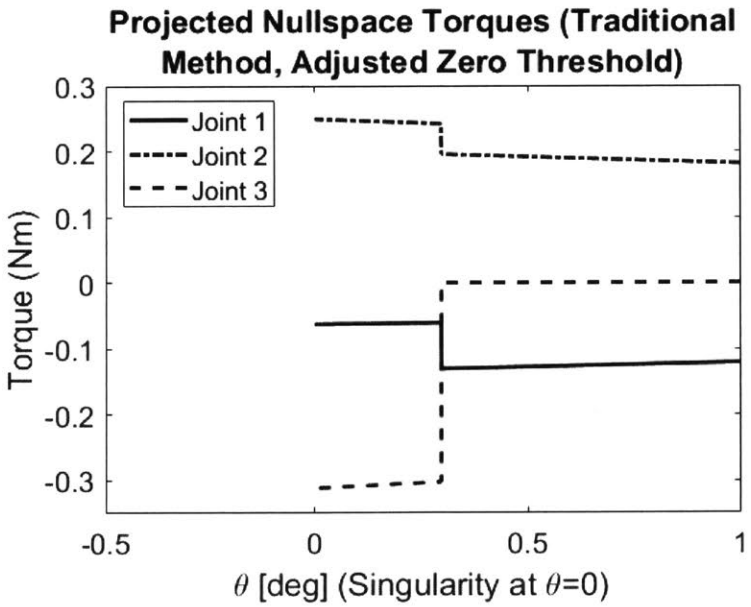


Figure 6-6: Nullspace projected torques along each joint, generated from the traditional method, with the zero singular value threshold adjusted to 0.005.

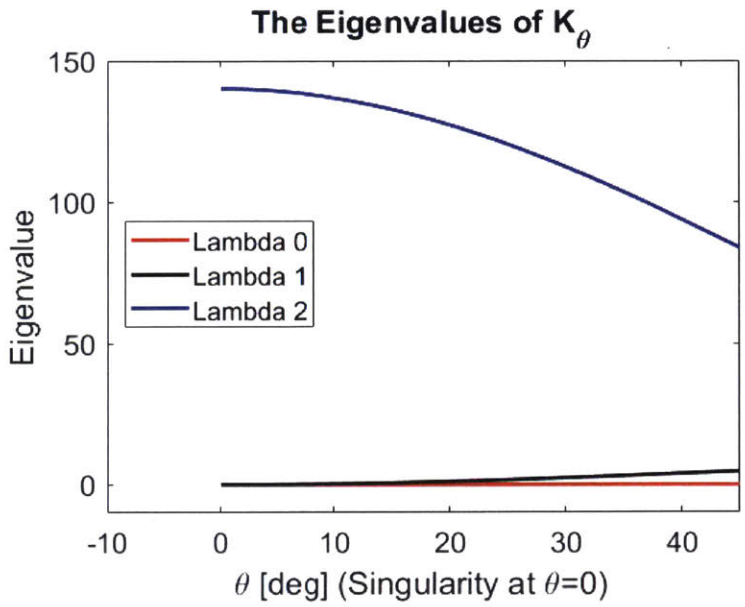


Figure 6-7: The eigenvalues of $\boldsymbol{K}_\theta$. The values are plotted starting at $\theta = 45°$, in order to better visualize λ_1

general robot, the eigendecomposition of $\boldsymbol{K}_\theta$ is given by Equation 6.17. As a robot moves towards a singularity, λ_1 (and perhaps more eigenvalues) begin to steadily tend towards zero. While this is happening, all computations performed with this control law remain well bounded and continuous. When the robot does reach singularity, λ_1 becomes zero. As this happens, all of the eigenvectors appear to rapidly shift directions.

For the example three link system, there were three eigenvalues associated with $\boldsymbol{K}_\theta$: λ_0, λ_1, and λ_2. Here, λ_0 was always zero, due to the single degree of redundancy. In order to form the null-space stiffness matrix, $\boldsymbol{K}_{\theta,Null}$, this eigenvalue was always replaced by an arbitrary constant, k_1 (which was set to unity). λ_1 in this experiment was strictly positive in non-singular configurations, but gradually became zero as the manipulator reached kinematic singularity. λ_2 in this experiment remained strictly positive. These eigenvalues are plotted in Figure 6-7.

For the first numerical experiment with this approach, λ_0 was always replaced with $k_1 = 1$, and λ_1 was allowed to become zero without replacement. Thus, the

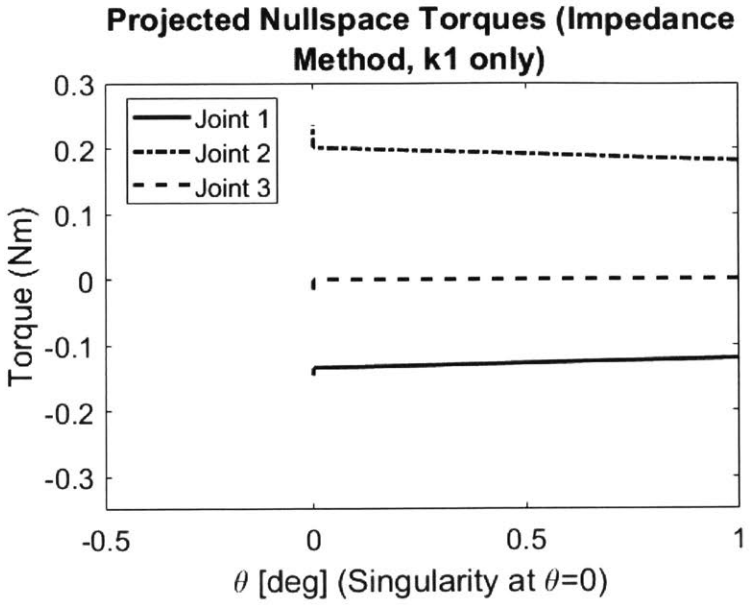


Figure 6-8: The nullspace torques arising from replacing λ_0 with unity, but not replacing λ_1.

nullspace stiffness was always given by:

$$\boldsymbol{K}_{\theta,Null} = \boldsymbol{v}_1 k_1 \boldsymbol{v}_1^T \tag{6.22}$$

Which, under the control law in Equation 6.21, gave rise to the nullspace torques:

$$\boldsymbol{\tau}_{null} = \boldsymbol{v}_1 k_1 \boldsymbol{v}_1^T (\boldsymbol{q}_0 - \boldsymbol{q}) \tag{6.23}$$

These torques are plotted in Figure 6-8. At singularity, there was an apparent step in the manipulator torques. This step occurred as a result of the rapid changes in the direction of $\boldsymbol{v}_1$ when the manipulator reached singularity. While these changes are in principle continuous, they happen extremely rapidly, so that for all practical purposes, they will be experienced as torque discontinuities. The magnitude of these torque increases depends on the magnitude of k_1

The above experiment differentiated between the zero eigenvalues arising from the redundancy, and the zero eigenvalues that arise at singularity. Eigenvalues of the second type were then allowed to remain at zero. In real robotic systems, however,

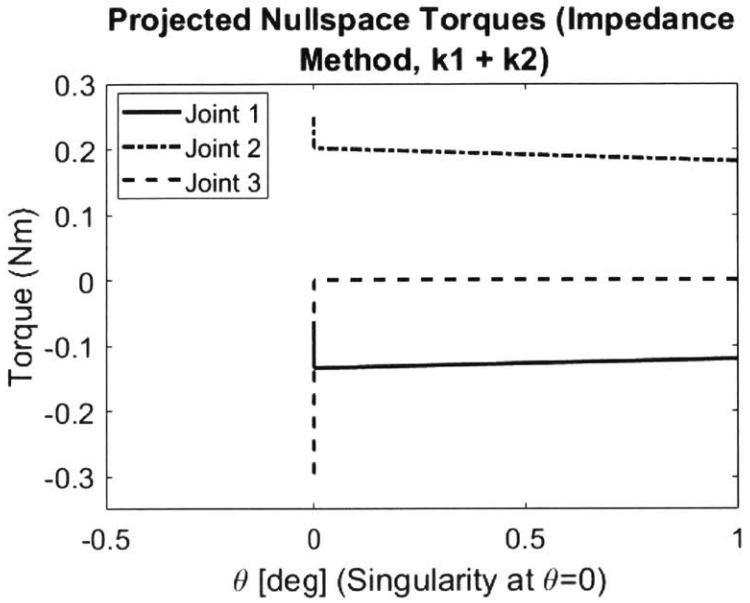


Figure 6-9: The nullspace torques arising from replacing λ_0 with unity, and λ_1 with unity at singularity. Note the similarity to Figure 6-5.

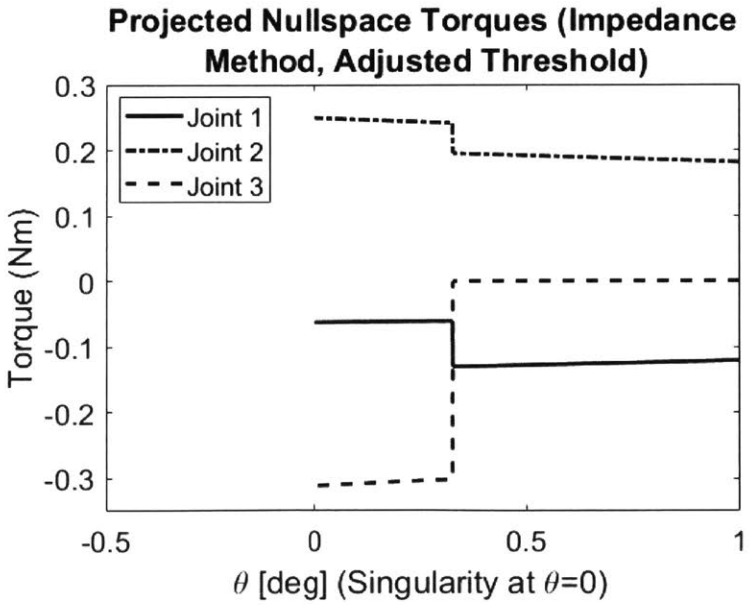


Figure 6-10: The nullspace torques arising from replacing λ_0 with unity, and λ_1 with unity at a reduced threshold of 0.0003. Note the similarity to Figure 6-6.

it may not be trivial to track the origins of all the eigenvalues that are zero at singularity. Furthermore, allowing eigenvectors to remain zero at singularities in high-DOF systems may lead to uncorrected drift in the manipulator's nullspace, if the manipulator is held at singularity for any extended periods of time. A more practical control scheme would be to automatically replace any zero eigenvalue with a constant, k. This will lead to an actual step discontinuity in the nullspace projected torques.

Furthermore, since this controller is implemented in floating point arithmetic, we must specify a threshold for regarding a specific eigenvalue as zero, in order to replace it with the constant nullspace stiffness parameter. An experiment was performed where λ_1 was replaced by unity once it crossed a threshold of 1×10^{-13}. Due to the selected sampling resolution of θ this threshold was only crossed at the singularity point. Thus, the nullspace projection torques were given by Equation 6.23 when the manipulator was not at singularity. The torques at singularity were given by:

$$\boldsymbol{\tau}_{null} = (\boldsymbol{v}_1 k_1 \boldsymbol{v}_1^T + \boldsymbol{v}_2 k_2 \boldsymbol{v}_2^T)(\boldsymbol{q}_0 - \boldsymbol{q}) \tag{6.24}$$

where $k_1 = k_2 = 1$. These torques are plotted in Figure 6-9. At singularity, there is a superposition of the continuous (but extremely rapid) torque changes from the k_1 term, and the discontinuous step, which arises from adding the k_2 term at singularity. These torque curves are identical to the torques generated by the traditional method (Figure 6-5).

Now that the source of the discontinuity is isolated in the k_2 parameter, one potential way to eliminate the added discontinuity is to ramp up and ramp down the value of k_2 whenever singularity is entered or exited. However, this requires advanced knowledge of the manipulator state, and is therefore not suitable for unpredictable, or poorly modeled interaction tasks. Furthermore, this method requires a way to reliably separate the redundancy zero eigenvalues from the singularity zero eigenvalues.

In the traditional nullspace projection, the discontinuity could be moved away from the singularity point by increasing the threshold that set the singular values in

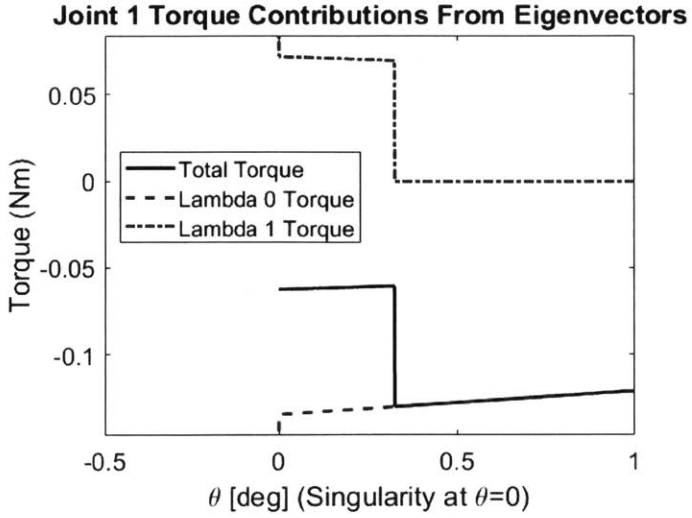
Jacobian pseudoinversion to zero. In the impedance based approach, similar behavior was obtained when the threshold for replacing λ_1 with k_2 was increased. Figure 6-10 shows the resulting torques when the threshold was set to 3×10^{-4}. These torques appeared nearly identical to the torques plotted in 6-6. One item of interest was that the additional increase in torques at singularity that was observed in Figure 6-8 was absent from these torques.

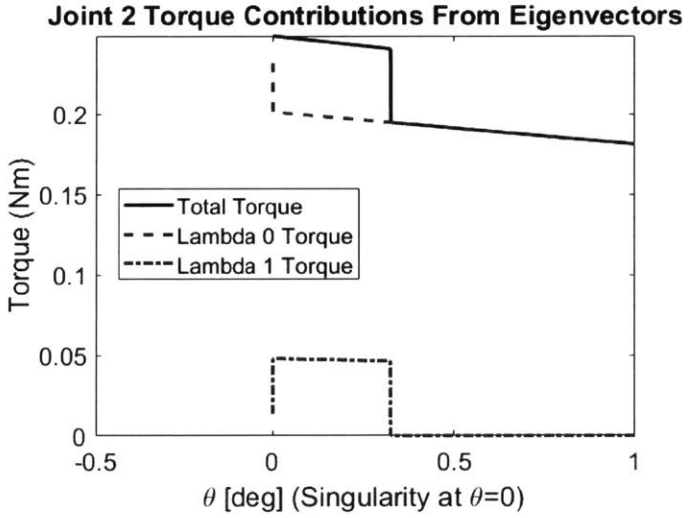
This phenomenon was further explored by plotting the contributing torques from each eigenvector and k pair for each joint individually. These contributions are plotted in Figure 6-11. From these plots, several patterns were observed.

Before λ_1 crosses the arbitrary zero threshold, its associated eigenvectors do not appear in $\boldsymbol{K}_{\theta,Null}$. Thus, the total torque precisely tracks the λ_0 torque, and the λ_1 torque is zero. Once λ_1 crosses the zero threshold, and gets replaced by k_2, there is a discontinuous jump in the λ_1 torque (but not the λ_0 torque). This causes to total torque to increase discontinuously as well. Finally, at singularity, both of the eigenvectors change direction extremely rapidly. However, the sudden torque contribution from the λ_0 eigenvector is exactly canceled out by the sudden torque contribution from the λ_1 eigenvector. Thus, the total torque remains unchanged. From this we see that the the discontinuity in Figure 6-8 was caused by rapid changes in the λ_0 eigenvector torque contribution not being canceled by an equal and opposite rapid change in the λ_1 eigenvector torque contribution.

This suggests that it might be possible to eliminate all discontinuities by ramping up k_2 before singularity is reached, such that at singularity, the sudden torque changes from the two eigenvectors would cancel each other. This strategy is visualized in Figure 6-12.

In this experiment, the threshold for substituting λ_1 with k_2 was set at 0.0003. When this threshold was reached, k_2 was linearly increased, such that it would reach its maximum value of unity before the manipulator reached singularity. Once the manipulator reached singularity, the sudden torque increases from each set of eigenvectors cancelled each other. This led to perfectly smooth and continuous torques into singularity.





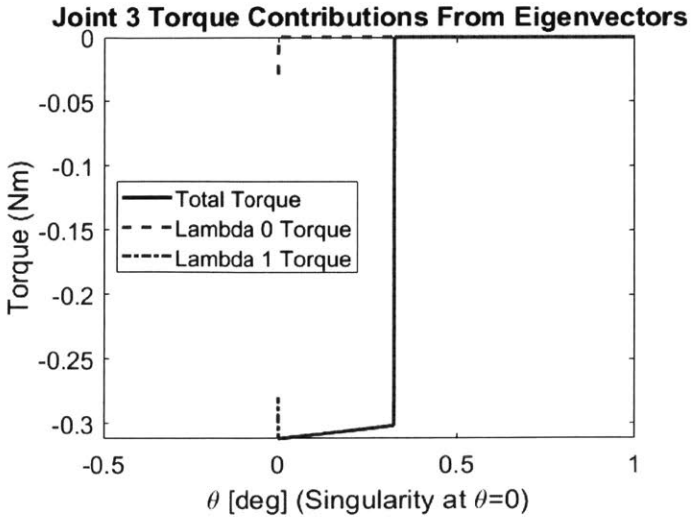


Figure 6-11: The torque contribution from each eigenvector on each of the three links. The total torque is shown as a black dotted line, and the two eigenvector component torques are depicted as dotted lines.

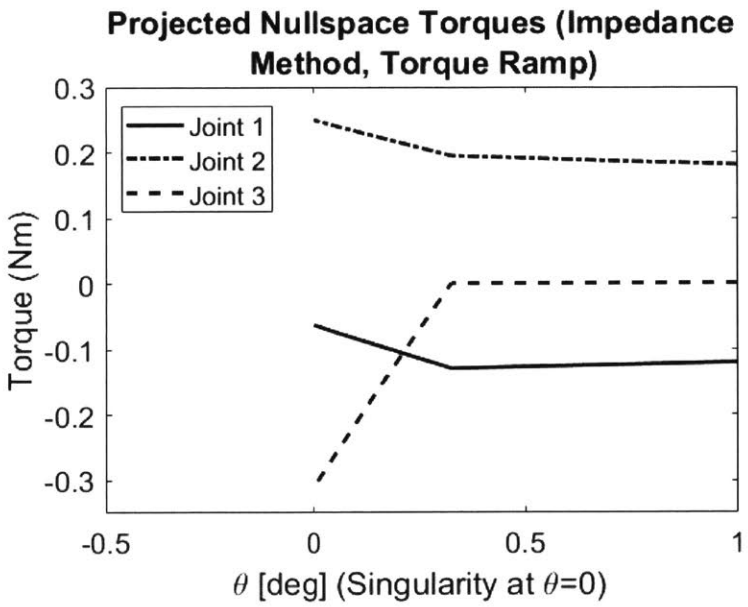


Figure 6-12: Eliminating all discontinuities on a predetermined trajectory by ramping up k_2 before singularity is reached.

This approach is not without its drawbacks. In addition to the aforementioned limitations of requiring advance knowledge of the manipulator states (and thereby not being able to apply the technique to unpredictable or poorly modeled tasks), prematurely introducing k_2 and linearly ramping it before the manipulator reaches singularity will introduce errors in the desired endpoint behavior, which is the very thing we sought to avoid by introducing nullspace projections methods. It should be noted, however, that many of the methods meant to soften the torque jumps into singularity for the traditional nullspace projection approach also introduce errors in the desired endpoint behavior around singularities [Dietrich et al., 2012, Dietrich, 2016, Chiaverini, 1997, Deo and Walker, 1995].

In summary, the new controller given in Eqn. 6.21 provides identical performance to that in Eqn. 6.5 inside the dexterous workspace, but with a reduced, and more intuitive parameter space. At singularity, while the methods may appear to behave similarly, the impedance based method can identify sources of discontinuity in a more explicit and intuitive manner.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this work, the idea of reducing the human motor controller to an equivalent modulated impedance, and nominal position command was first discussed. This formed the biologically inspired basis for impedance control in robots. An impedance control scheme for single and multi-DOF robots was derived, with and without inertial compensation.

Chapter 2 discussed the linear superposition property of nonlinear impedance controllers, along with applications to redundancy resolution in overactuated robots, and simplifying the control of closed kinematic chain systems and collaborative manipulation. The idea of breaking down complex tasks into simpler sub-tasks with individual impedance controllers was explored. Finally, the potential to translate this idea to very high-DOF humanoid systems was discussed.

Chapter 3 discussed the particulars of implementing this control scheme on a Baxter Research Robot, including hardware and control software details. The issue of numerical singularities with minimal coordinate representations of rotations in impedance controllers was also touched upon.

Chapter 4 discussed the theory and methodology behind a set of system ID experiments designed to evaluate the impedance controller performance in Baxter, and to verify the compositionality of impedances in a closed kinematic chain configuration.

It was found that a commanded static stiffness could be delivered by each individual arm to within a 2% error. The static stiffness linear superposition was verified to within a 3% error.

In terms of the dynamic damping and inertial parameters, step responses carried out at a nominal stiffness of either 400 N/m or 500 N/m were found to yield satisfactory second order response, while a driving nominal stiffness of 300 N/m was insufficient to overshadow inherent nonlinear manipulator frictional effects. Endpoint damping on the right arm was verified to be within 17% of the predicted value, while damping on the right arm was found to be within 57% of the specified value. Since it is known that the Baxter Robot is a low-cost manipulator with a large manufacturing variability, this is not particularly surprising. Damping composition, however, was verified to within 7% error.

Estimates for the endpoint x inertia were quite poor, but this may be due to a variety of reasons. Since the inertia matrix has relatively sizable off-diagonal elements, compared to the main diagonal elements, the upper right hand value might not be particularly representative of the inertial behavior about the x coordinate. During the step responses, displacements were observed along the y direction in the single arm trials, but not the linkage trials. This may have impacted the accuracy of the individual arm inertia measurements, in addition to causing the measured closed chain inertia to deviate from the sum of the single arm inertia measurements. In any event, it is widely acknowledged that obtaining accurate inertia measurements for a robotic manipulator is a challenging task [Dietrich et al., 2015, Nakanishi et al., 2008].

Chapter 5 provided some sample manipulation tasks, including stable closed chain manipulation into a singularity, along with an implementation of a shoe buffing task. This is an example of a seemingly complex task which can be broken down into a number of smaller tasks, each of which can be addressed with a separate impedance controller.

Chapter 6 provided an overview of traditional nullspace projection methods and a discussion of the projection's performance at manipulator singularities. A new, impedance based implementation of the nullspace projection method was provided

which may have improved properties over the traditional projection method for torque controlled robots under impedance control.

7.2 Future Work

Many of the experiments, derivations, and discussions in this thesis have prompted new avenues for potential exploration in future work. Several of these research topics are listed here.

7.2.1 System Identification of Back-Driving Impedances

In Section 4.1, the difference between the forward driving dynamics (Eqn. 4.8) and back-driving dynamics (Eqn. 4.9) was discussed. Between the two dynamic equations, while the inertia and stiffness would probably be the same, there might be significant nonlinear frictional effects, which would make the fitted forward driving damping differ from the back-driving damping. For the purposes of an interaction controller, identifying the parameters on the back-driving side potentially matter far more than those on the forward driving side.

Since we have a two-armed robot platform available, one potential way to perform this system identification is to have one robot arm in position control mode interact with a second robot arm with an impedance controller running. If all interactions are limited to a single coordinate, an approximation to these dynamics might be two 1-DOF robot manipulators, as shown in Figure 7-1. The robots have effective masses M_1 and M_2, and are actuated by actuator forces F_{act1} and F_{act2}. The external forces on each robot are given by F_{ext1} and F_{ext2}. Let us also assume the existence of some possibly non-linear (coulomb or viscous) joint friction forces (F_f) acting on both robots.

Since the Baxter Robot has relatively high bandwidth torque control on the output side of the gearbox, it is assumed that F_{act1} and F_{act2} are known quantities. If we clamp both robot end effectors together, we are left with the system depicted in Figure 7-2. Let us place robot 1 under a second order impedance controller, and robot 2 in

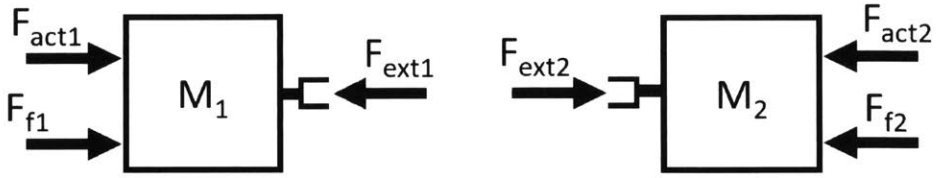


Figure 7-1: Two 1-DOF robots to approximate the behavior of two 7-DOF manipulators interacting along a single coordinate.

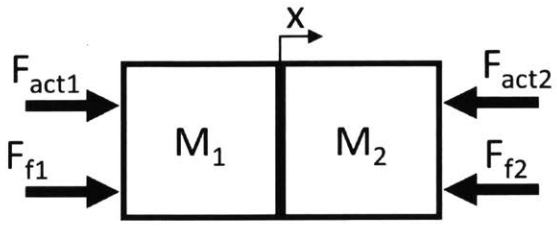


Figure 7-2: Two 1-DOF robots clamped together.

a very rigid, high-bandwidth position control mode.

F_{act1} is given by our impedance control law: $F_{act1} = k(x_0 - x) + b(\dot{x}_0 - \dot{x})$. F_{act2} is given by an unknown control law which is presumably some very high gain PD or PID controller, perhaps with a superimposed inverse dynamics feedforward controller to compensate for inertial forces from M_2. We can model this as a controlled flow source (S_f), and therefore, a system input. The closed loop system is depicted in Figure 7-3. For the purposes of evaluating the interactive dynamics (rather than the forward path dynamics), x_0 will be held constant at zero.

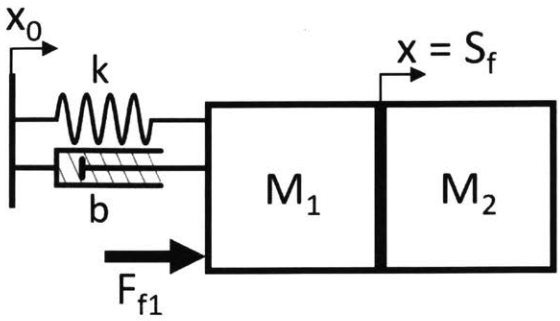


Figure 7-3: Two clamped 1-DOF robots interacting one another. Robot 1 is under second order impedance control, and robot 2 is in a very rigid, high-bandwidth position control mode.

One potential avenue for this system ID is to examine the dynamics of the entire system. Assuming the second robot acts as a perfect flow source, and that the combined robot system mass is $M_1 + M_2$, the system behaves as follows, for a twice differentiable commanded $x(t)$:

$$F_{act2} = kx + b\dot{x} + F_{f1} + F_{f2} + (M_1 + M_2)\ddot{x} \tag{7.1}$$

This model isn't very workable in its current state, since it involves two (hopefully identical) friction models. If we assume, as we did earlier, that the joint friction for each arm can be modeled as viscous damping with a known value, we can fold both friction terms into b and compensate for them in the fits. Alternatively, we can ignore the friction terms and be cognizant that our fit for b will include un-modeled nonlinear friction effects from two arms, rather than a single arm. It should be noted that F_{f1} is friction felt upon back-driving, and F_{f2} is friction felt in the forward path. If we assume known viscous friction and fold both frictions into b to get b_{tnet}, we are left with:

$$F_{act2} = kx + b_{net}\dot{x} + (M_1 + M_2)\ddot{x} \tag{7.2}$$

Fitting the interaction behavior to this model presupposes the linear summing of the masses, which was shown to be questionable in the aforementioned system identification experiments. Additionally, if our actual $x(t)$ doesn't track our commanded $x(t)$ very well (i.e. if our assumption about robot 2 being a perfect position source is invalid), then in order to force it into our framework, we are left with double differentiating our actual $x(t)$ signal, which is impractical.

Alternatively, we can change the causality of the system (consider robot 2 to be an effort source, rather than a flow source) to get a proper system ODE. Now, we can treat the measured F_{act2} as the system's input function, and the measured $x(t)$ as the system's output function. We can integrate the ODE to get $X_{int}(t)$. This integration can be nested within a non-linear optimizer to yield estimates for k, b_{net}, M_1, and M_2 by driving the error between the measured $x(t)$ and the integrated $x_{int}(t)$ to zero.

For any of these methods, all inertial estimates would depend on getting sizable values for $\ddot{x}$, which might be difficult to obtain without having the Jacobian matrix change significantly. Alternatively, if we are only after damping measurements, we could use a ramp function for $x(t)$, thereby yielding a zero second derivative.

This is only one possible method to yield a measurement of the back-driving dynamics of a robot. Alternative methods might involve placing a force-torque sensor between the two robot arms.

7.2.2 Tackling the Optimization Scale Up Issue

Another direction for future research is exploring the extent to which a compositional impedance control framework could mesh with an optimization-based approach. The approach with many trajectory optimization/MPC based approaches is to feed a high-DOF nonlinear (or linearized) system with many constraints into a commercial solver with a quadratic cost function, and hope for rapid convergence into a workable local minimum [Tedrake, 2019, Kuindersma et al., 2014, Betts, 1998].

If the idea of splitting up a complex, high-DOF system into modular components is viable, it would be worth examining how this could potentially assist with the scale-up problem in optimization approaches. Instead of optimizing a large array of joint torques over time, incorporating constraints from the entire system in one large quadratic program (QP) or nonlinear program (NLP), one could solve several smaller NLPs for desired impedance parameters and nominal trajectories over time. Having many smaller, independent optimization problems running in parallel might show improvements when compared to running a single, massive optimization.

One starting point for this might be to formulate a simple manipulation problem, such as a planar robot interacting with a spring. Automatically generating impedance parameters and a nominal trajectory via optimization to satisfy a cost function would be the first step towards this idea. Next, a high-DOF closed chain system might be considered (consisting of two planar robot arms, connected via a linkage, interacting with a spring). A comparison could be made between a naive MPC approach, one large joint-space impedance control optimization, and finally

two smaller superimposed endpoint-space impedance controllers with two additional joint-space controllers to manage redundancy, all generated via optimization.

7.2.3 Impedance Based Nullspace Projection Behavior at Singularity

One final question generated by this work is how the impedance based nullspace projection method behaves at singularity for robots with more degrees of freedom and redundancy. Implementing the impedance based method in a simulator and on real hardware to compare with the traditional method are important next steps. Another question that may be worth pursuing is to develop a method to separate out the $n - m$ ever-present nullspace dimensions from the $m - r$ additional nullspace dimensions gained at singularity. For parts of the dexterous workspace where the eigenvectors tend to change very little in direction from one control loop iteration to the next, this should be a straightforward problem to implement. However at locations near singular configurations, the eigenvalues and eigenvectors may to change very rapidly over a short distance, which may pose a challenge for tracking them from one time step to the next.

Finally, it is important to acknowledge that the literature on nullspace projections and smoothing torque discontinuities at singular configurations is vast. Therefore, it is highly likely that improvements in methodology can be sparked by surveying this literature, and performing comparisons of various projection techniques to the impedance based method.

Appendix A

Expected Endpoint Stiffness, Damping, and Inertia Matrices for System ID Experiments

The following are the expected net endpoint stiffness ($\boldsymbol{K}_{x,total}\big|_{\boldsymbol{f}=0}$), damping ($\boldsymbol{B}_{x,total}$), and inertia matrices (with and without reflected actuator inertias, $\boldsymbol{M}_x$ and $\boldsymbol{M}_{x,total}$, respectively) for each individual right and left robot arm endpoint. These are generated using Jacobian matrices evaluated at ideal mirrored joint angles (The "Nominal Angles" row of Table 4.2). This was done for simplicity, since the actual joint angles deviated negligibly from these nominal angles. When a $\pm$ sign is given in the matrix, the value is positive for the left arm, and negative for the right arm, and vice versa for the $\mp$ sign. All of the matrices are symmetric, and the values were rounded to two decimal places (zero entries may represent small values).

A.1 Net Endpoint Stiffnesses

These matrices are generated using Eqn. 4.2.

$$\boldsymbol{K}_{x,total,500} = \begin{bmatrix} 504.00 & \mp 0.91 & 8.45 & \mp 0.01 & 1.86 & \pm 0.18 \\ \cdot & 500.25 & \mp 1.95 & 0.00 & \mp 0.43 & -0.06 \\ \cdot & \cdot & 518.38 & 0.00 & 4.04 & \pm 0.42 \\ \cdot & \cdot & \cdot & 5.00 & 0.00 & 0.00 \\ \cdot & \cdot & \cdot & \cdot & 5.90 & \pm 0.09 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 5.02 \end{bmatrix} \tag{A.1}$$

$$\boldsymbol{K}_{x,total,400} = \begin{bmatrix} 404.00 & \mp 0.91 & 8.45 & \mp 0.01 & 1.86 & \pm 0.18 \\ \cdot & 400.25 & \mp 1.95 & 0.00 & \mp 0.43 & -0.06 \\ \cdot & \cdot & 418.38 & 0.00 & 4.04 & \pm 0.42 \\ \cdot & \cdot & \cdot & 5.00 & 0.00 & 0.00 \\ \cdot & \cdot & \cdot & \cdot & 5.90 & \pm 0.09 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 5.02 \end{bmatrix} \tag{A.2}$$

$$\boldsymbol{K}_{x,total,300} = \begin{bmatrix} 304.00 & \mp 0.91 & 8.45 & \mp 0.01 & 1.86 & \pm 0.18 \\ \cdot & 300.25 & \mp 1.95 & 0.00 & \mp 0.43 & -0.06 \\ \cdot & \cdot & 318.38 & 0.00 & 4.04 & \pm 0.42 \\ \cdot & \cdot & \cdot & 5.00 & 0.00 & 0.00 \\ \cdot & \cdot & \cdot & \cdot & 5.90 & \pm 0.09 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 5.02 \end{bmatrix} \tag{A.3}$$

A.2 Net Endpoint Damping

This matrix generated using Eqn. 4.4:

$$\boldsymbol{B}_{x,total} = \begin{bmatrix} 31.44 & \mp 1.37 & -2.62 & \mp 0.47 & -0.20 & \mp 1.12 \\ \cdot & 35.14 & \mp 0.28 & -0.30 & \mp 0.21 & -2.12 \\ \cdot & \cdot & 28.04 & \pm 1.12 & 3.74 & \pm 0.64 \\ \cdot & \cdot & \cdot & 0.80 & \pm 0.59 & 0.22 \\ \cdot & \cdot & \cdot & \cdot & 2.76 & \pm 0.21 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 2.17 \end{bmatrix} \tag{A.4}$$

A.3 Net Endpoint Inertia

This matrix is generated using Eqn. 4.5:

$$\boldsymbol{M}_x = \begin{bmatrix} 4.40 & \mp 2.43 & -0.38 & 0.00 & -0.07 & \pm 0.57 \\ \cdot & 8.77 & \pm 1.07 & -0.01 & \pm 0.25 & -1.83 \\ \cdot & \cdot & 3.76 & 0.00 & 0.67 & \mp 0.25 \\ \cdot & \cdot & \cdot & 0.00 & 0.00 & 0.00 \\ \cdot & \cdot & \cdot & \cdot & 0.14 & \mp 0.06 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0.40 \end{bmatrix} \tag{A.5}$$

This matrix is generated using Eqn. 4.7:

$$\boldsymbol{M}_{x,total} = \begin{bmatrix} 11.53 & \mp 5.29 & -1.73 & \mp 0.01 & -0.34 & \pm 1.14 \\ \cdot & 12.16 & \pm 1.47 & -0.04 & \pm 0.33 & -2.66 \\ \cdot & \cdot & 7.38 & -0.09 & 1.58 & \mp 0.32 \\ \cdot & \cdot & \cdot & 0.04 & \pm 0.04 & 0.01 \\ \cdot & \cdot & \cdot & \cdot & 0.43 & \mp 0.07 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0.65 \end{bmatrix} \tag{A.6}$$

Appendix B

Motions Along y and z Coordinates During Step Responses

While the step responses performed in Section 4.4 were only along the endpoint x coordinate, there were still some small residual oscillations and accelerations that occurred in the orthogonal endpoint coordinates. Plots of the endpoint displacements along the y and z coordinates are given below. All of the plots have axes which are scaled identically to those in Figures 4-7, 4-8, and 4-9, and are therefore directly comparable. All step commands occur at 1 second.

For the 500 N/m responses, there were some residual motions along the y coordinate in arms for the individual arm step responses. These motions peaked at about 1.5 cm relative to the nominal positions. There were very few residual motions along the z coordinate for the individual arm step responses. There were nearly no residual motions in the linkage step response.

The residual y motions for the 400 N/m responses were of slightly lower amplitude, peaking at about 1.5 cm relative to the nominal y position. Results for the z coordinates were similar to the 500 N/m case. The residual y motions for the 300 N/m case were of even lower amplitude, peaking at about 0.8 - 1 cm, and were slightly slower than the 500 N/m and 400 N/m cases.

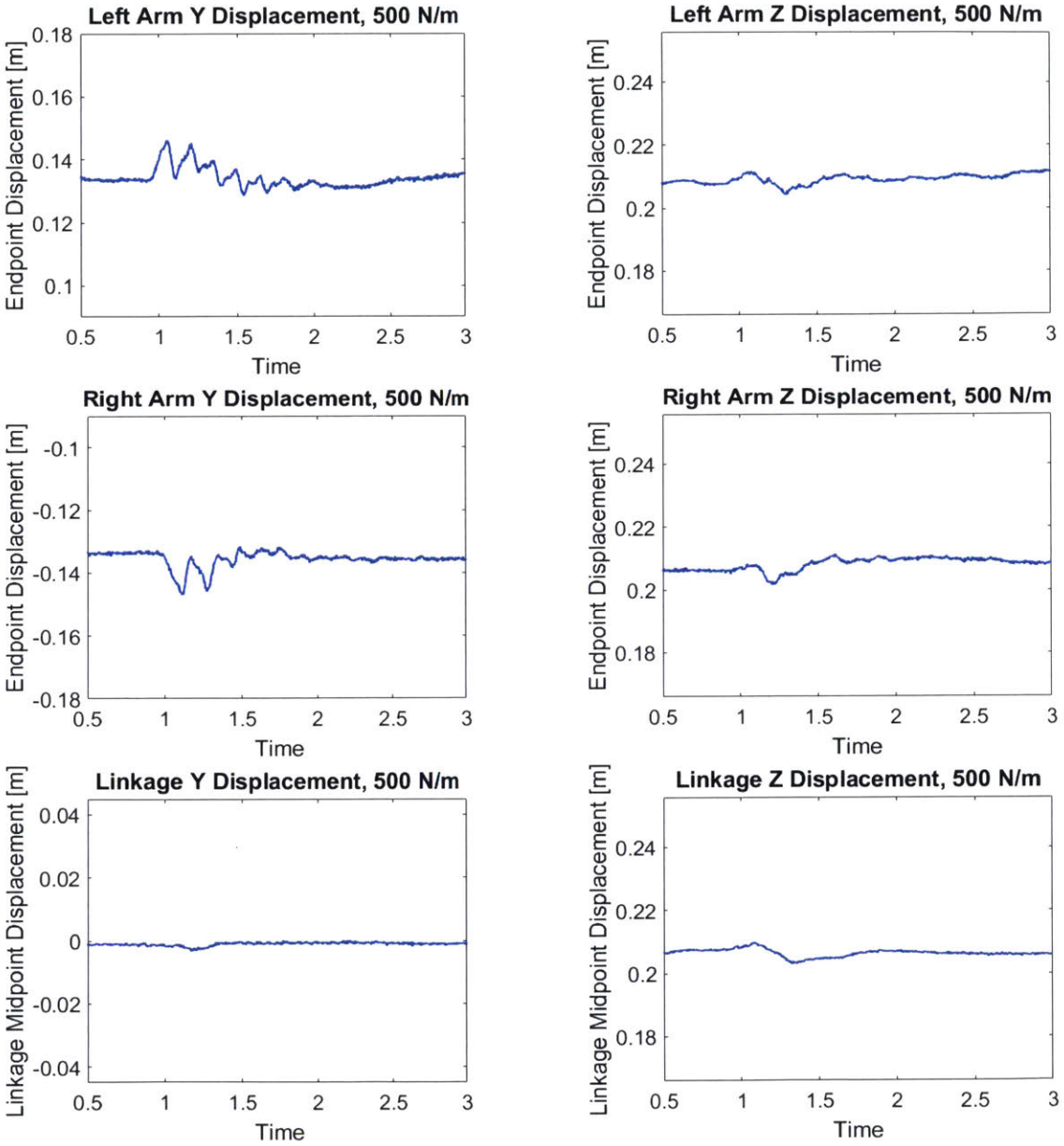


Figure B-1: Displacements along the y and z coordinates for the 500 N/m step responses

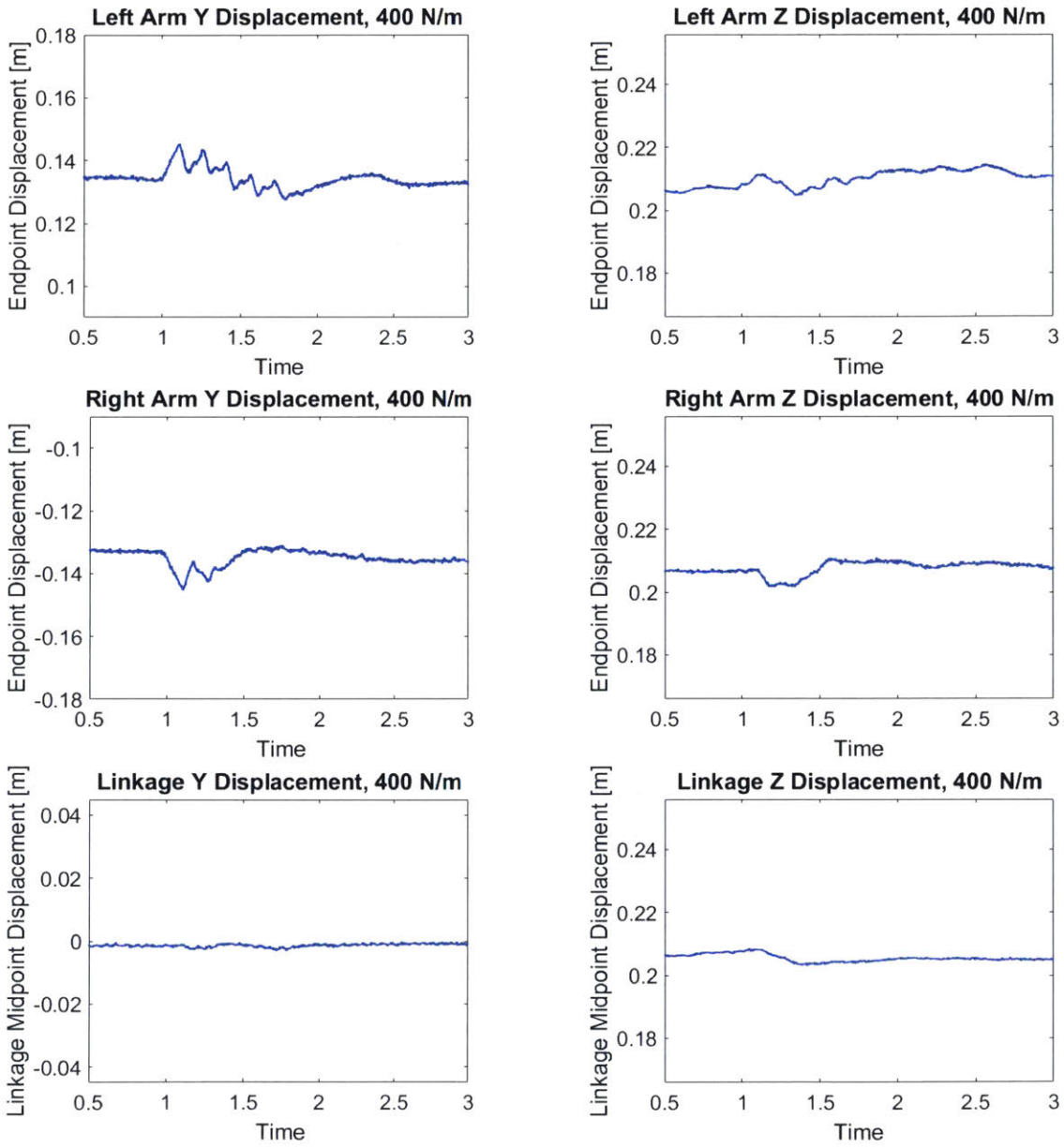


Figure B-2: Displacements along the y and z coordinates for the 400 N/m step responses

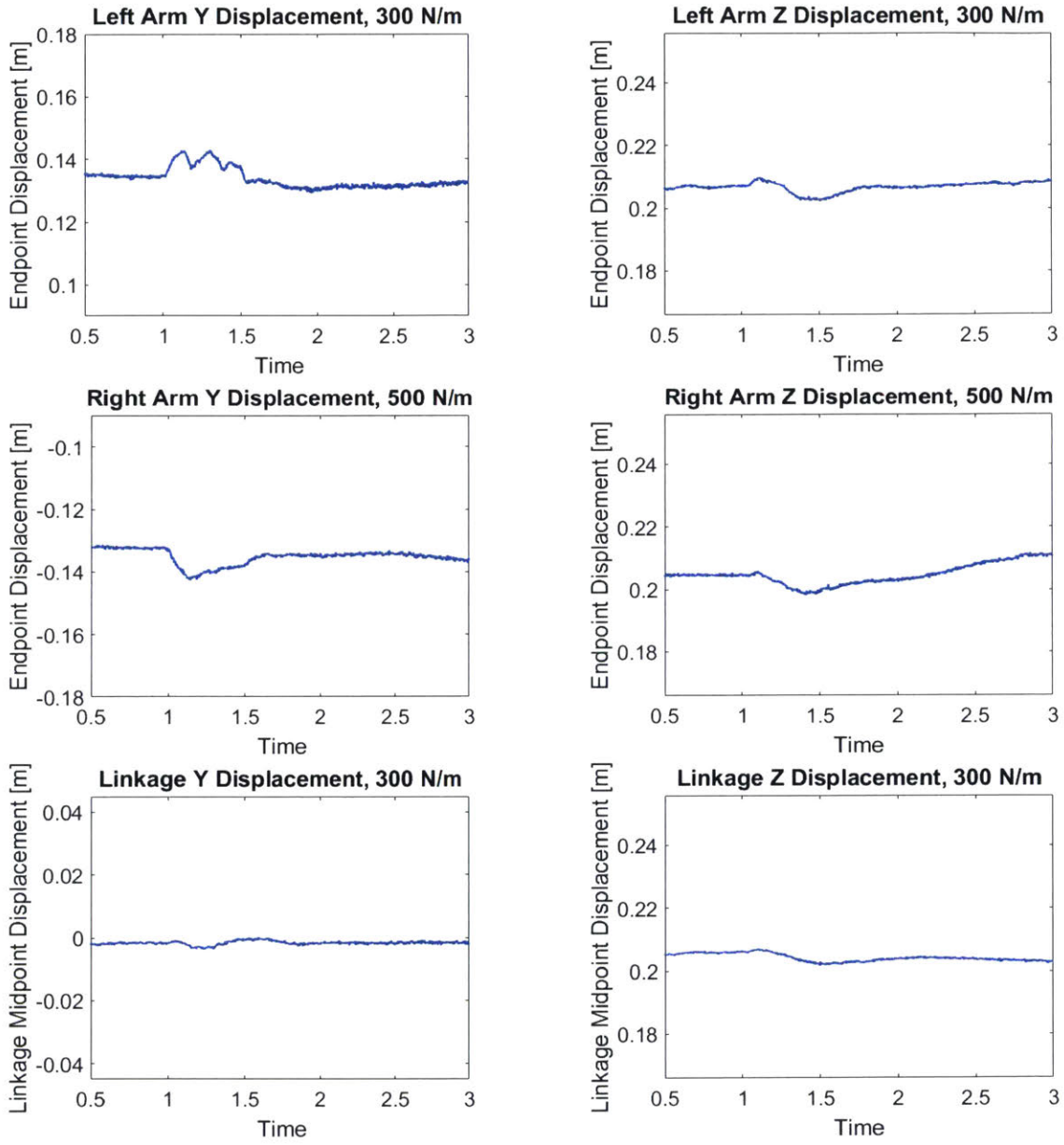


Figure B-3: Displacements along the y and z coordinates for the 300 N/m step responses

Bibliography

[Atkeson et al., 2018] Atkeson, C. G., Benzun, P. W. B., Banerjee, N., Berenson, D., Bove, C. P., Cui, X., DeDonato, M., Du, R., Feng, S., Franklin, P., Gennert, M., Graff, J. P., He, P., Jaeger, A., Kim, J., Knoedler, K., Li, L., Liu, C., Long, X., Padir, T., Polido, F., Tighe, G. G., and Xinjilefu, X. (2018). What Happened at the DARPA Robotics Challenge Finals. In Spenko, M., Buerger, S., and Iagnemma, K., editors, *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*, pages 667–684. Springer International Publishing, Cham.

[Bai et al., 2016] Bai, Y., Yu, W., and Liu, C. K. (2016). Dexterous Manipulation of Cloth. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics*, EG '16, pages 523–532, Goslar Germany, Germany. Eurographics Association.

[Balkcom and Mason, 2008] Balkcom, D. J. and Mason, M. T. (2008). Robotic origami folding. *The International Journal of Robotics Research*, 27(5):613–627.

[Betts, 1998] Betts, J. T. (1998). Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207.

[Bizzi et al., 1982] Bizzi, E., Accornero, N., Chapple, W., and Hogan, N. (1982). Arm trajectory formation in monkeys. *Experimental Brain Research*, 46(1):139–143.

[Bizzi et al., 1984] Bizzi, E., Accornero, N., Chapple, W., and Hogan, N. (1984). Posture control and trajectory formation during arm movement. *The Journal of Neuroscience*, 4(11):2738 LP – 2744.

[Boston Dynamics, 2019] Boston Dynamics (2019). Atlas. (Online) https://www.bostondynamics.com/atlas.

[Brown, 2001] Brown, F. T. (2001). *Engineering System Dynamics: A Unified Graph-Centered Approach*. CRC Press, Boca Raton, 2nd edition.

[Burdet et al., 2001] Burdet, E., Osu, R., Franklin, D. W., Milner, T. E., and Kawato, M. (2001). The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414:446.

[Caccavale et al., 2008] Caccavale, F., Chiacchio, P., Marino, A., and Villani, L. (2008). Six-DOF Impedance Control of Dual-Arm Cooperative Manipulators. *IEEE/ASME Transactions on Mechatronics*, 13(5):576–586.

[Caccavale et al., 1999] Caccavale, F., Natale, C., Siciliano, B., and Villani, L. (1999). Six-DOF impedance control based on angle/axis representations. *IEEE Transactions on Robotics and Automation*, 15(2):289–300.

[Caccavale and Uchiyama, 2016] Caccavale, F. and Uchiyama, M. (2016). Cooperative Manipulation. In Siciliano, B. and Khatib, O., editors, *Springer Handbook of Robotics*, pages 989–1006. Springer International Publishing, Cham.

[Chiaverini, 1997] Chiaverini, S. (1997). Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410.

[Colgate and Hogan, 1989a] Colgate, E. and Hogan, N. (1989a). An analysis of contact instability in terms of passive physical equivalents. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 404–409 vol.1.

[Colgate and Hogan, 1989b] Colgate, E. and Hogan, N. (1989b). The Interaction of Robots with Passive Environments: Application to Force Feedback Control. In *Advanced Robotics: 1989*, pages 465–474. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Craig and Raibert, 1979] Craig, J. and Raibert, M. (1979). A systematic method of hybrid position/force control of a manipulator. *COMPSAC 79. Proceedings. Computer Software and The IEEE Computer Society's Third International Applications Conference, 1979.*, pages 446–451.

[Damm and McIntyre, 2008] Damm, L. and McIntyre, J. (2008). Physiological Basis of Limb-Impedance Modulation During Free and Constrained Movements. *Journal of Neurophysiology*, 100(5):2577–2588.

[Deo and Walker, 1995] Deo, A. S. and Walker, I. D. (1995). Overview of damped least-squares methods for inverse kinematics of robot manipulators. *Journal of Intelligent and Robotic Systems*, 14(1):43–68.

[Diebel, 2006] Diebel, J. (2006). Representing attitude: Euler angles, unit quaternions, and rotation vectors. *Matrix*, 58(15-16):1–35.

[Dietrich, 2016] Dietrich, A. (2016). *Whole-Body Impedance Control of Wheeled Humanoid Robots*. Springer International Publishing, Basel.

[Dietrich et al., 2015] Dietrich, A., Ott, C., and Albu-Schäffer, A. (2015). An overview of null space projections for redundant, torque-controlled robots. *The International Journal of Robotics Research*, 34(11):1385–1400.

[Dietrich et al., 2012] Dietrich, A., Wimbock, T., Albu-Schaffer, A., and Hirzinger, G. (2012). Integration of Reactive, Torque-Based Self-Collision Avoidance Into a Task Hierarchy. *IEEE Transactions on Robotics*, 28(6):1278–1293.

[Doeringer, 1999] Doeringer, J. A. (1999). *An Investigation into the Discrete Nature of Human Arm Movements.* Ph.d. thesis, Massachusetts Institute of Technology.

[Eaton, 2015] Eaton, J. (2015). GNU Octave: libinterp/corefcn/pinv.cc Source File. (Online) http://octave.org/doxygen/4.0/db/da0/pinv_8cc_source.html.

[Feldman, 1966] Feldman, A. G. (1966). Functional Tuning of the Nervous System With Control of Movement or Maintenance of a Steady Posture– II. Controllable Parameters of the Muscles. *Biofizika*, 11(3):498–508.

[Franklin et al., 2007] Franklin, D. W., Liaw, G., Milner, T. E., Osu, R., Burdet, E., and Kawato, M. (2007). Endpoint Stiffness of the Arm Is Directionally Tuned to Instability in the Environment. *The Journal of Neuroscience*, 27(29):7705 – 7716.

[Franklin et al., 2015] Franklin, G. F., Powell, J. D., and Emami-Naeini, A. (2015). *Feedback control of dynamic systems.* Pearson, Upper Saddle River, 7th edition.

[Gasparetto et al., 2012] Gasparetto, A., Vidoni, R., Pillan, D., and Saccavini, E. (2012). Automatic Path and Trajectory Planning for Robotic Spray Painting. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6.

[Goodwine, 2004] Goodwine, B. (2004). Inverse Kinematics. In Kurfess, T., editor, *Handbook of Robotics and Automation*, chapter 3. CRC Press, Boca Raton.

[Hanson, 2006] Hanson, A. J. (2006). *Visualizing Quaternions.* San Francisco: The Morgan Kaufmann Series in Interactive 3D Technology.

[Hayati, 1986] Hayati, S. (1986). Hybrid position/Force control of multi-arm cooperating robots. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, pages 82–89.

[Hermus, 2018] Hermus, J. R. (2018). *Human Physical Interaction with a Circular Constraint.* S.m. thesis, Massachusetts Institute of Technology.

[Hoffer and Andreassen, 1981] Hoffer, J. A. and Andreassen, S. (1981). Regulation of soleus muscle stiffness in premammillary cats: intrinsic and reflex components. *Journal of Neurophysiology*, 45(2):267–285.

[Hogan, 1984] Hogan, N. (1984). Adaptive control of mechanical impedance by coactivation of antagonist muscles. *IEEE Transactions on Automatic Control*, 29(8):681–690.

[Hogan, 1985a] Hogan, N. (1985a). Impedance Control: An Approach to Manipulation: Part I - Theory. *Journal of Dynamic Systems, Measurement, and Control*, 107(1):1–7.

[Hogan, 1985b] Hogan, N. (1985b). Impedance Control: An Approach to Manipulation: Part II - Implementation. *Journal of Dynamic Systems, Measurement, and Control*, 107(1):8–16.

[Hogan, 1985c] Hogan, N. (1985c). Impedance Control: An Approach to Manipulation: Part III - Applications. *Journal of Dynamic Systems, Measurement, and Control*, 107(1):17–24.

[Hogan, 1987] Hogan, N. (1987). Stable execution of contact tasks using impedance control. In *1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 1047–1054.

[Hogan, 1990] Hogan, N. (1990). Mechanical Impedance of Single- and Multi-Articular Systems. In Winters, J. M. and Woo, S. L.-Y., editors, *Multiple Muscle Systems: Biomechanics and Movement Organization*, chapter 9, pages 149–164. Springer New York, New York, NY.

[Hogan, 2014] Hogan, N. (2014). A General Actuator Model Based on Nonlinear Equivalent Networks. *IEEE/ASME Transactions on Mechatronics*, 19(6):1929–1939.

[Hogan, 2017] Hogan, N. (2017). Physical Interaction via Dynamic Primitives. In Laumond, J.-P., Mansard, N., and Lasserre, J.-B., editors, *Geometric and Numerical Foundations of Movements*, pages 269–299. Springer International Publishing, Cham.

[Hogan and Breedveld, 2002] Hogan, N. and Breedveld, P. (2002). The Physical Basis of Analogies in Network Models of Physical System Dynamics. In Bishop, R. H., editor, *The Mechatronics Handbook*, chapter 15. CRC Press, Boca Raton.

[Hopcroft et al., 1991] Hopcroft, J. E., Kearney, J. K., and Krafft, D. B. (1991). A Case Study of Flexible Object Manipulation. *The International Journal of Robotics Research*, 10(1):41–50.

[Horowitz and Hill, 2015] Horowitz, P. and Hill, W. (2015). *The Art of Electronics*. Cambridge University Press, New York, 3rd edition.

[Hosford, 2016] Hosford, L. A. (2016). *Development and Testing of an Impedance Controller on an Anthropomorphic Robot for Extreme Environment Operations*. S.m. thesis, Massachusetts Institute of Technology.

[Humphrey and Reed, 1983] Humphrey, D. and Reed, D. J. (1983). Separate cortical systems for control of joint movement and joint stiffness: reciprocal activation and coactivation of antagonist muscles. In Desmedt, J. E., editor, *Motor Control Mechanisms in Health and Disease (Advances in Neurology)*, volume 39, pages 347–372. Raven Press, New York.

[Kandel et al., 2013] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S. A., and Hudspeth, A. J. (2013). *Principles of Neural Science*. McGraw Hill, New York, 5th edition.

[Khatib, 1986] Khatib, O. (1986). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research*, 5(1):90–98.

[Khatib, 1987] Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43–53.

[Khatib, 1995] Khatib, O. (1995). Inertial Properties in Robotic Manipulation: An Object-Level Framework. *The International Journal of Robotics Research*, 14(1):19–36.

[Knight, 2013] Knight, W. (2013). Baxter: The Blue-Collar Robot. *MIT Technology Review*.

[Koga et al., 1992] Koga, M., Kosuge, K., Furuta, K., and Nosaki, K. (1992). Coordinated motion control of robot arms based on the virtual internal model. *IEEE Transactions on Robotics and Automation*, 8(1):77–85.

[Kosuge and Hirata, 2004] Kosuge, K. and Hirata, Y. (2004). Coordinated Motion Control of Multiple Manipulators. In Kurfess, T. R., editor, *Robotics and Automation Handbook*, chapter 20. CRC Press, Boca Raton.

[Krotkov et al., 2018] Krotkov, E., Hackett, D., Jackel, L., Perschbacher, M., Pippine, J., Strauss, J., Pratt, G., and Orlowski, C. (2018). The DARPA Robotics Challenge Finals: Results and Perspectives. In Spenko, M., Buerger, S., and Iagnemma, K., editors, *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*, pages 1–26. Springer International Publishing, Cham.

[Kuindersma et al., 2014] Kuindersma, S., Permenter, F., and Tedrake, R. (2014). An efficiently solvable quadratic program for stabilizing dynamic locomotion. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2589–2594.

[Luh and Zheng, 1987] Luh, J. Y. S. and Zheng, Y. F. (1987). Constrained Relations between Two Coordinated Industrial Robots for Motion Control. *The International Journal of Robotics Research*, 6(3):60–70.

[Maroney, 2014] Maroney, K. (2014). Weighing from Joint Torque output - Baxter Research Robot Community - Google Groups. (Online) https://groups.google.com/a/rethinkrobotics.com/forum/#!topic/brr-users/8Zs_eC8k5N4.

[Mason, 1981] Mason, M. T. (1981). Compliance and Force Control for Computer Controlled Manipulators. *IEEE Transactions on Systems, Man and Cybernetics*, 11(6):418–432.

[MathWorks, 2019a] MathWorks (2019a). Floating-point relative accuracy - MATLAB eps. (Online) https://www.mathworks.com/help/matlab/ref/eps.html.

[MathWorks, 2019b] MathWorks (2019b). Moore-Penrose pseudoinverse - MATLAB pinv. (Online) https://www.mathworks.com/help/matlab/ref/pinv.html.

[Miller et al., 2011] Miller, S., van den Berg, J., Fritz, M., Darrell, T., Goldberg, K., and Abbeel, P. (2011). A geometric approach to robotic laundry folding. *The International Journal of Robotics Research*, 31(2):249–267.

[Miomir et al., 2008] Miomir, V., Dragoljub, S., Yury, E., and Katic, D. (2008). *Dynamics and Robust Control of Robot-environment Interaction.* World Scientific Publishing Co., Inc., River Edge, NJ, USA.

[Mussa-Ivaldi and Hogan, 1991] Mussa-Ivaldi, F. A. and Hogan, N. (1991). Integrable Solutions of Kinematic Redundancy via Impedance Control. *International Journal of Robotics Research*, 10(5):481–491.

[Nakanishi et al., 2008] Nakanishi, J., Cory, R., Mistry, M., Peters, J., and Schaal, S. (2008). Operational Space Control: A Theoretical and Empirical Comparison. *The International Journal of Robotics Research*, 27(6):737–757.

[Nakano et al., 1974] Nakano, E., Ozaki, S., Ishida, T., and Kato, I. (1974). Cooperational Control of the Anthropomorphous Manipulator MELARM. In *Proceedings of the 4th International Symposium on Industrial Robots.*

[NASA, 2014] NASA (2014). Robonaut 2 Fact Sheet FS-2014-02-002-JSC. Technical report, NASA Johnson Space Center, Houstion, TX.

[Netter, 2014] Netter, F. H. (2014). *Atlas of Human Anatomy.* Saunders Elsevier, Philadelphia, 6th edition.

[Newman and Hogan, 1987] Newman, W. and Hogan, N. (1987). High speed robot control and obstacle avoidance using dynamic potential functions. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 14–24.

[Newman, 1992] Newman, W. S. (1992). Stability and Performance Limits of Interaction Controllers. *Journal of Dynamic Systems, Measurement, and Control*, 114(4):563–570.

[Nichols and Houk, 1976] Nichols, T. R. and Houk, J. C. (1976). Improvement in linearity and regulation of stiffness that results from actions of stretch reflex. *Journal of Neurophysiology*, 39(1):119–142.

[Open Source Robotics Foundation, 2019] Open Source Robotics Foundation (2019). ROS Core Components. (Online) http://www.ros.org/core-components/.

[Ortenzi et al., 2017] Ortenzi, V., Stolkin, R., Kuo, J., and Mistry, M. (2017). Hybrid motion/force control: a review. *Advanced Robotics*, 31(19-20):1102–1113.

[Paine et al., 2015] Paine, N., Mehling, J. S., Holley, J., Radford, N. A., Johnson, G., Fok, C.-L., and Sentis, L. (2015). Actuator Control for the NASA-JSC Valkyrie Humanoid Robot: A Decoupled Dynamics Approach for Torque Control of Series Elastic Robots. *J. Field Robot.*, 32(3):378–396.

[Phillips and Parr, 2011] Phillips, C. L. and Parr, J. M. (2011). *Feedback Control Systems.* Pearson, Upper Saddle River, 5th edition.

[Pieper, 1968] Pieper, D. L. (1968). *The Kinematics of Manipulators Under Computer Control.* Ph.d. thesis, Stanford University.

[Raibert and Craig, 1981] Raibert, M. H. and Craig, J. J. (1981). Hybrid Position/Force Control of Manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 103(2):126.

[Rethink Robotics, 2015a] Rethink Robotics (2015a). Arm Calibration - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Arm_Calibration.

[Rethink Robotics, 2015b] Rethink Robotics (2015b). Arm Control Modes - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Arm_Control_Modes.

[Rethink Robotics, 2015c] Rethink Robotics (2015c). Arm Control Overview - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Arm_Control_Overview.

[Rethink Robotics, 2015d] Rethink Robotics (2015d). Arms - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Arms.

[Rethink Robotics, 2015e] Rethink Robotics (2015e). Custom IKFast For Your Baxter - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Custom_IKFast_for_your_Baxter.

[Rethink Robotics, 2015f] Rethink Robotics (2015f). Gravity Compensation - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Gravity_Compensation.

[Rethink Robotics, 2015g] Rethink Robotics (2015g). Hardware Specifications - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Hardware_Specifications.

[Rethink Robotics, 2015h] Rethink Robotics (2015h). URDF - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/URDF.

[Rethink Robotics, 2015i] Rethink Robotics (2015i). Workspace Guidelines - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Workspace_Guidelines.

[Rethink Robotics and Maroney, 2015] Rethink Robotics and Maroney, K. (2015). Baxter PyKDL - SDK - Wiki. (Online) http://sdk.rethinkrobotics.com/wiki/Baxter_PyKDL.

[Salehian et al., 2018] Salehian, S. S. M., Lin, H.-C., Fernandez, N. B. F., Smith, J., Mistry, M., and Billard, A. (2018). Transitioning with confidence during contact/non-contact scenarios. In *IROS 2018 Workshop on Towards Robots that Exhibit Manipulation Intelligence.*

[Schabowsky et al., 2007] Schabowsky, C. N., Hidler, J. M., and Lum, P. S. (2007). Greater reliance on impedance control in the nondominant arm compared with the dominant arm when adapting to a novel dynamic environment. *Experimental Brain Research*, 182(4):567 – 577.

[Selen et al., 2009] Selen, L. P. J., Franklin, D. W., and Wolpert, D. M. (2009). Impedance Control Reduces Instability That Arises from Motor Noise. *Journal of Neuroscience*, 29(40):12606–12616.

[Siciliano et al., 2010] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2010). *Robotics: modelling, planning and control*. Springer Science & Business Media, London.

[Siciliano and Slotine, 1991] Siciliano, B. and Slotine, J.-J. E. (1991). A general framework for managing multiple tasks in highly redundant robotic systems. In *Fifth International Conference on Advanced Robotics: Robots in Unstructured Environments*, pages 1211–1216 vol.2.

[Spong and Vidyasagar, 1989] Spong, M. and Vidyasagar, M. (1989). *Robot Dynamics and Control*. John Wiley & Sons, New York, 1st edition.

[Stepien et al., 1987] Stepien, T., Sweet, L., Good, M., and Tomizuka, M. (1987). Control of tool/workpiece contact force with application to robotic deburring. *IEEE Journal on Robotics and Automation*, 3(1):7–18.

[Strang, 2016] Strang, G. (2016). *Introduction to Linear Algebra*. Wellesley-Cambridge Press, Wellesley, MA, 5th edition.

[Tedrake, 2019] Tedrake, R. L. (2019). Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832). (Online) http://www.ros.org/core-components/.

[The SciPy Community, 2018] The SciPy Community (2018). numpy.linalg.pinv. (Online) https://docs.scipy.org/doc/numpy-1.15.1/reference/generated/numpy.linalg.pinv.html.

[Uchiyama and Dauchez, 1988] Uchiyama, M. and Dauchez, P. (1988). A symmetric hybrid position/force control scheme for the coordination of two robots. In *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, pages 350–356 vol.1.

[Valenzuela, 2016] Valenzuela, A. K. (2016). *Mixed-Integer Convex Optimization for Planning Aggressive Motions of Legged Robots Over Rough Terrain*. Ph.d. thesis, Massachusetts Institute of Technology.

[Whitney, 1977] Whitney, D. E. (1977). Force Feedback Control of Manipulator Fine Motions. *Journal of Dynamic Systems, Measurement, and Control*, 99(2):91–97.

[Yamakawa et al., 2011] Yamakawa, Y., Namiki, A., and Ishikawa, M. (2011). Motion planning for dynamic folding of a cloth with two high-speed robot hands and two high-speed sliders. In *2011 IEEE International Conference on Robotics and Automation*, pages 5486–5491.