

Copyright

by

Kyle Anthony Schroeder

2013

**The Dissertation Committee for Kyle Anthony Schroeder Certifies that this is the
approved version of the following dissertation:**

**Requirements for Effective Collision Detection on Industrial Serial
Manipulators**

Committee:

Sheldon Landsberger, Supervisor

Mitch Pryor, Co-Supervisor

Troy Harden

Benito Fernandez

Ronald Barr

Requirements for Effective Collision Detection on Industrial Serial Manipulators

by

Kyle Anthony Schroeder, B.S.M.E.; M.S.E.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

August, 2013

Dedication

This work is dedicated to my parents, Bruce and Susan Schroeder.

Acknowledgements

I would like to acknowledge and thank Dr. Mitch Pryor, Dr. Troy Harden, and Dr. Sheldon Landsberger for their support and guidance during my doctoral education and for this work in particular. I would like to acknowledge and thank Agile Planet for their support and cooperation. Thank you to Los Alamos National Laboratory for funding, support, and the use of facilities and equipment.

Requirements for Effective Collision Detection on Industrial Serial Manipulators

Kyle Anthony Schroeder, Ph.D

The University of Texas at Austin, 2013

Supervisor: Sheldon Landsberger

Co-Supervisor: Mitch Pryor

Human-robot interaction (HRI) is the future of robotics. It is essential in the expanding markets, such as surgical, medical, and therapy robots. However, existing industrial systems can also benefit from safe and effective HRI. Many robots are now being fitted with joint torque sensors to enable effective human-robot collision detection. Many existing and off-the-shelf industrial robotic systems are not equipped with these sensors. This work presents and demonstrates a method for effective collision detection on a system with motor current feedback instead of joint torque sensors. The effectiveness of this system is also evaluated by simulating collisions with human hands and arms.

Joint torques are estimated from the input motor currents. The joint friction and hysteresis losses are estimated for each joint of an SIA5D 7 Degree of Freedom (DOF) manipulator. The estimated joint torques are validated by comparing to joint torques predicted by the recursive application of Newton-Euler equations. During a pick and place motion, the estimation error in joint 2 is less than 10 Newton meters. Acceleration increased the estimation uncertainty resulting in estimation errors of 20 Newton meters over the entire workspace.

When the manipulator makes contact with the environment or a human, the same technique can be used to estimate contact torques from motor current. Current-estimated contact torque is validated against the calculated torque due to a measured force. The error in contact force is less than 10 Newtons. Collision detection is demonstrated on the SIA5D using estimated joint torques.

The effectiveness of the collision detection is explored through simulated collisions with the human hands and arms. Simulated collisions are performed both for a typical pick and place motion as well as trajectories that transverse the entire workspace. The simulated forces and pressures are compared to acceptable maximums for human hands and arms. During pick and place motions with vertical and lateral end effector motions at 10mm/s and 25mm/s, the maximum forces and pressures remained below acceptable levels. At and near singular configurations some collisions can be difficult to detect. Fortunately, these configurations are generally avoided for kinematic reasons.

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1. Research goals	8
1.2. Approach and Outline	9
2. BLACK BOX MODEL FOR ESTIMATING JOINT TORQUE	11
2.1. Joint Modeling Literature Review	13
2.2. Calculated Dynamics	17
2.3. Black Box Model	19
2.4. Parameter Estimation Setup	23
2.5. Parameter Estimation Results	32
3. VALIDATION OF TORQUE ESTIMATE	35
3.1. Estimated and Predicted Comparison Experiments	35
3.2. Torque Due to Contact Force Validation	44
4. COLLISION DETECTION	53
4.1. Point of Actuation Collision Detection Review	53
4.2. Demonstrated Method	55
4.3. Considerations in Joint Space	57
4.4. Considerations in Operational Space	59
4.4.1. Null space and condition number	60
4.4.2. System near singularity, force in ‘near-null’ space	63
4.4.3. Manipulator configuration and detection	64
4.5. Demonstration	67
4.5.1. Plexiglas	67
4.5.2. 5”x8” Fiberboard on foam	72
4.5.3. Glovebox glass	77

4.5.4. Banana.....	79
4.6. Summary	81
5. COLLISION SIMULATION	82
5.1. The simulator	83
5.2. Validation.....	88
5.3. Parameter comparison.....	90
5.4. Chapter Summary	97
6. EFFECTIVENESS OF COLLISION DETECTION	99
6.1. Introduction.....	99
6.1.1. Sharp/penetrating collisions.....	100
6.1.2. Blunt collisions	100
6.1.2.1. Unclamped collisions.....	100
6.1.2.2. Clamped collisions.....	101
6.1.3. Human Collision Response.....	101
6.2. Human-Robot Collision Simulations.....	104
6.2.1. Standard configuration, downward clamp	104
6.2.2. Standard configuration, sideways clamp	109
6.2.3. Random configurations at 10mm/s EEf velocity	110
6.2.4. Sensitivity prediction	115

6.3. Chapter Summary	118
7. CONCLUSIONS AND FUTURE WORK	120
7.1. Summary	120
7.2. Contributions and Conclusions	125
7.3. Future Work.....	127
APPENDIX	129
REFERENCES	153

LIST OF TABLES

Table 2.1: Friction parameter determination, repeated 15 times	25
Table 2.2: Black Box Model Coefficients	34
Table 3.1: Validation Motion Statistics	43
Table 4.1: Example collision torques required to detect collision.....	58
Table 4.2: Velocities and joint thresholds for Plexiglas collision tests	69
Table 4.3: Plexiglas tests with forces and torque.....	70
Table 4.4: Results of CD on 5x8 fiberboard on foam.....	75
Table 4.5: Glovebox glass collision detection results.....	78
Table 4.6: Banana collision detection results	80
Table 5.1: Simulator input and output parameters.....	85
Table 5.2: Parameters for control-group simulations	91
Table 5.3: Relationship between parameters and maximum force	96
Table 5.4: Instantaneous change in force for small change in given parameter	97
Table 6.1: Limits for human-robot collisions [translated from German by the author. source: BGIA 2011].....	102
Table 6.2: Range of random joint positions.....	111
Table 7.1: Validation Motion Statistics	121
Table 7.2: Relationship between parameters and maximum force	124

LIST OF FIGURES

Figure 1.1: (left) iRobot Roomba home vacuuming robot. [iRobot 2013] (right) Texai telepresence robot. [Willow Garage 2013]	1
Figure 1.2: KUKA/DLR Lightweight Arms (top left), Robonaut (top right), and Kawada Nextage (bottom left), and Rethink Robotics Baxter (bottom right)	2
Figure 1.3: Example Glovebox [http://bit.ly/x6TEmw] (left) and Glovebox Gloves [http://bit.ly/Aog9in] (right).....	6
Figure 1.4: Typical Robots in a Radiation Facility: MANTIS Teleoperated Robot (left) [http://bit.ly/x8UVav] and Aries Robot in Glovebox (right) [http://1.usa.gov/xgnUZ2] ...	7
Figure 1.5: 2 Motoman SIA5D 7 Degree of Freedom (DOF) Industrial Serial Manipulators	8
Figure 2.1: Example commercially-available position-controlled robot system	11
Figure 2.2: Yaskawa Motoman SIA5D 7 degree of freedom 5kg payload serial manipulator	12
Figure 2.3: Harmonic drive Gear Train	13
Figure 2.4: Joint Model.....	20
Figure 2.5: Commercial system with black-box model torque estimator	23
Figure 2.6: Joint 2 friction characterization motion, side view	26

Figure 2.7: Joint 2: Example predicted torque versus measured current for determining velocity dependent losses.....	27
Figure 2.8: Joint 2 friction current versus joint velocity.....	28
Figure 2.9: Joint currents for sample constant velocity motions	29
Figure 2.10: Joint 2 hysteresis example data	31
Figure 2.11: Example lumped inertia parameter graph	32
Figure 2.12: Friction model data, curves, fits, and uncertainty	33
Figure 3.1: Estimated joint torque is significant to manipulator interaction and human safety	35
Figure 3.2: Pick and place example motion for validation	36
Figure 3.3: Example linear EEF validation motion positions.....	37
Figure 3.4: Example linear EEF validation motion error and acceleration	38
Figure 3.5: Joint positions for workspace test 1 joint move	39
Figure 3.6: First workspace validation motion, joint 2 torque error and acceleration.....	40
Figure 3.7: Second workspace validation test, joint positions.....	41
Figure 3.8: Second workspace validation motion, joint 2 torque error and acceleration .	42
Figure 3.9: First workspace validation motion, all joints torque error	43
Figure 3.10: Contact test setup with foam (left), Plexiglas (center), and fiberboard on foam (right)	46
Figure 3.11: EEF pressing on foam at 25mm/s.....	47
Figure 3.12: Joint 2 force error, hysteresis, and acceleration for 25mm/s EEF velocity on foam	48
Figure 3.13: EEF pressing on foam at 50mm/s.....	49
Figure 3.14: EEF pressing on foam at 75mm/s.....	49
Figure 3.15: EEF pressing on Plexiglas at 25mm/s	50
Figure 3.16: EEF pressing on Plexiglas at 125mm/s	51
Figure 3.17: EEF pressing on 5x5 fiberboard on foam at 25mm/s.....	52
Figure 4.1: Illustrative collision detection with collision torque and sensor noise.....	55
Figure 4.2: Commercial system with black box torque estimation and collision detection	56
Figure 4.3: Illustration relating Joint, Operational, and Human Injury spaces.....	57
Figure 4.4: Illustration of worst case collision torque for detection	58
Figure 4.5: 2 DOF planar robot at a singularity.....	65
Figure 4.6: Illustration of effects of bandwidth on collision detection response.....	66
Figure 4.7: Setup for collision detection on Plexiglas target	68
Figure 4.8: Estimated joint 2 torque and measured EEF force at 25mm/s with Plexiglas	69
Figure 4.9: Estimated joint 2 torque and measured EEF force at 175mm/s with Plexiglas	71
Figure 4.10: Plexiglas collision detection at 200mm/s with 20Nm thresholds	72
Figure 4.11: EEF collision detection tests with fiberboard on foam	73
Figure 4.12: Collision detection at EEF 25mm/s on 5x8 fiberboard on foam	74
Figure 4.13: Collision detection at EEF 100mm/s on 5x8 fiberboard on foam.....	76
Figure 4.14: Acceleration at EEF 100mm/s on 5x8 fiberboard on foam.....	77

Figure 4.15: Setup of collision detection tests with glovebox glass	78
Figure 4.16: Setup for collision detection with banana	79
Figure 4.17: Collision detection at EEF 5mm/s with banana	80
Figure 4.18: Collision detection at EEF 17.5mm/s with banana	81
Figure 5.1: Robot collision simulator diagram	86
Figure 5.2: Simulator block diagram	87
Figure 5.3: Example collision simulation data	88
Figure 5.4: Example Plexiglas stiffness estimation	89
Figure 5.5: Simulated ($K=19100N/m$) and experimental results for 50mm/s collisions with Plexiglas.....	90
Figure 5.6: Distribution of maximum force for 1000 samples at different EEF velocities	91
Figure 5.7: Collision force dependence on eef velocity	92
Figure 5.8: Collision force dependence on stiffness.....	93
Figure 5.9: Collision force dependence on detection threshold.....	94
Figure 5.10: Collision force dependence on controller feedback rate	95
Figure 5.11: Collision force dependence on maximum joint acceleration	96
Figure 6.1: Cranium/Forehead force and stress limits	103
Figure 6.2: Standard configuration downward clamping motion	105
Figure 6.3: Simulated collisions with human hand at different EEF velocities	106
Figure 6.4: Simulated collisions with human lower arm/wrist at different EEF velocities	107
Figure 6.5: Simulated collisions with human upper arm/elbow at different EEF velocities	108
Figure 6.6: Trigger joints for lateral hand motions.....	109
Figure 6.7: Results of simulated collisions with hand during lateral EEF motions.....	110
Figure 6.8: EEF velocity distribution.....	112
Figure 6.9: Maximum force distribution with random positions and velocities with hand collisions	113
Figure 6.10: EEF locations at detection (black X indicates force exceeded acceptable level, color of dot is distance from origin for acceptable-force collisions)	114
Figure 6.11: 2 DOF planar robot at a singularity.....	115
Figure 6.12: Maximum force versus condition number frequency map. Color and cell number indicate the frequency out of 25,000 simulations. Red are most frequent, green are least.	116
Figure 6.13: Maximum force versus minimum singular value frequency map.....	117
Figure 7.1: Joint 2 force error, hysteresis, and acceleration for 25mm/s EEF velocity on foam	122
Figure 7.2: Estimated joint 2 torque and measured EEF force at 25mm/s with Plexiglas	123
Figure 7.3: Simulated collisions with human hand at different EEF velocities	124
Figure 7.4: Maximum force distribution with random positions and velocities with hand collisions	125

Figure 7.5: Example lumped inertia parameter graph 128

1. INTRODUCTION

Safe Human-Robot Interaction (HRI) will be needed for residential, medical, co-manufacturing and in many other applications. Robots are sweeping our floors [iRobot, 2013] and providing more functional tele-presence [Willow Garage, 2013]. As mobile platforms such as these become more reliable, they will provide a means for introducing manipulation to residential and commercial settings. An aging world populace is leading researchers to investigate elderly care robots [Mukhopadhyay and Gupta, 2007][Hansen et al., 2010][Meng and Lee, 2004]. Physical and spatial boundaries have protected humans in the past, but the trend toward human-robot interaction requires new means for ensuring safety.



Figure 1.1: (left) iRobot Roomba home vacuuming robot. [iRobot 2013] (right) Texai telepresence robot. [Willow Garage 2013]

Even though they are becoming more popular, high performance robots designed for safe human-robot interaction have not found widespread use. Compliant robots and stiff robots with joint torque sensors for collision detection, etc., are found mostly in research laboratories. These robots are expensive and often custom built for research (Figure 1.2, top left) [Albu-Schaeffer et al., 2007] or special applications, such as space

exploration (Figure 1.2, top right) [Diftler et al., 2011]. The Kawada Nextage, a commercial robot designed to share human workspaces uses visual collision prevention (Figure 1.2, bottom left). [Saenz, 2011][Kawada, 2012] Another industrial manipulator designed for affordable HRI is the Baxter by Rethink Robotics (Figure 1.2, bottom right). [Rethink Robotics 2013] Technology must become more robust and the hardware more affordable for widespread industrial application.

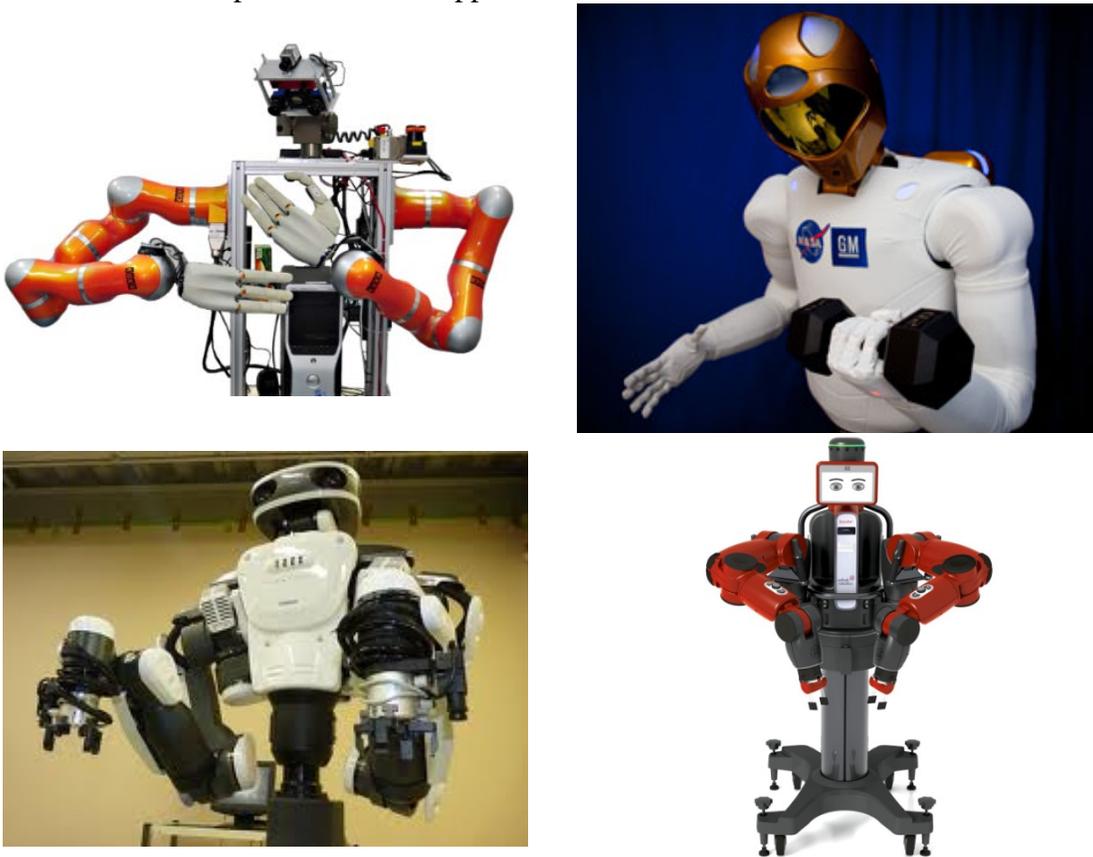


Figure 1.2: KUKA/DLR Lightweight Arms (top left), Robonaut (top right), and Kawada Nextage (bottom left), and Rethink Robotics Baxter (bottom right)

Commercially-viable industrial robots rarely have the capability to perform collision detection and are not compliant. They generally lack torque sensors to avoid additional costs and are built stiff to maintain desired trajectories under load. Despite current standards that largely prohibit physical human-robot interaction, industrial robots

still injure workers; even fatally crushing employees. [NIOSH, 1984, 1999, and 2001] Fortunately, since the first robot fatality in 1979, [Kravets, 2010] only a small number of people have died in accidents involving robots¹. However, people continue to be injured by industrial machines and robots. [Local, 2009]

Researchers have focused on several ways to ensure human-robot interaction is safe. Hardware solutions have included new sensors for detecting collisions such as skins [Marks, 2010] or whiskers [Jung and Zelinsky, 1996] and mechanically compliant joints to limit injury in case of collision. [Bicchi and Tonietti, 2004] Software solutions include collision detection and avoidance algorithms as well as compliant control algorithms that use force or torque feedback to effectively decrease robot stiffness. [Zollo et al., 2002] Both collision avoidance and detection have been demonstrated through several different means including model-based collision prevention [Harden, 2002][Swint, 2004][Spencer, et al. 2008], redundancy resolution for obstacle avoidance [Maciejewski & Klein, 1985][Duguleana et al., 2012], sensor-based collision detection, [Zheng and Sias, 1986][Lu and Chung, 2005][Ralph and Pai, 1995], etc.

Current safety standards for robotics are prescribed by The International Organization for Standardization (ISO) and by The American National Standards Institute (ANSI) in conjunction with The Robotics Industries Association (RIA). The ANSI/RIA standards are published in ANSI/RIA R15.06-1999 and are being updated for 2011. [Design Safety Engineering, Inc., 2011] The ISO Standards are published in ISO 10218-1:2006.

Both the ISO and ANSI/RIA standards dictate that humans should not be in the robot's workspace when operating in automatic mode. There is a provision for testing

¹ Only one other documented industrial robot death could be found. The death of Kenji Urada was caused by a robot on 4 July 1981. [Desert News 1981]

automated motions while the operator is in the workspace, but full operation is disallowed. The standards include much information on protecting the worker and preventing operation should a human enter the workspace. The safety of the human is – correctly – the utmost priority.

When the configurations of the environment can be restricted and always known, the robot knows *a priori* the current state of the environment. When a human enters the space, this state is disrupted. Humans can adapt to changes in the environment at a nearly-continuous rate, but robots cannot. To assist robots in changing environments, methods of collision avoidance, prevention, and detection have been demonstrated. Research has been done to prevent collisions through workspace models and vision feedback. Also, researchers have demonstrated the ability to sense collisions after they occur. Misleadingly, both of these are sometimes called “collision detection”. Some have taken the next step in planning motions after the detection of the collision while completing a goal or objective. [De Luca et al., 2006] To prevent confusion, the following definitions are used in this work.

Collision avoidance refers to algorithms that move the robot into configurations that avoid collisions. Obstacles are known or detected before contact is made between the robot and the object (or the robot itself). Collision avoidance is a preventative technique. Collision avoidance methods generally use vision or models of the robot and the environment to identify when the robot’s proposed motion will lead to a collision. Whiskers [Jung & Zelinsky, 1996] can often be considered non-interfering and thus considered collision avoidance techniques. In some cases, collision avoidance is integrated with motion planning. An in-depth coverage of collision avoidance techniques (including world-model and visual feedback) is beyond the scope of this work but is

discussed by Harden [2002], Swint, [2004], Ebert and Henrich [2002], and Eitner et al. [2008].

Collision prevention refers to methods that prevent actual collisions by stopping the robot. Unlike avoidance, prevention techniques do not attempt to complete the task. The goal is to safely stop and inform the operator who reassesses the situation and takes appropriate actions. Light curtains, deterministic model minimum distance algorithms [Harden 2002], and other techniques that indicate a potential collision fall into this category.

Collision detection refers specifically to methods that recognize when a collision has occurred. Many ways of detecting collisions have been demonstrated and can be roughly categorized as surface coverings, end-effector (EEF) force/torque (F/T) sensors, or point-of-actuation sensors. The method demonstrated and studied in this work is a point-of-actuation technique. Surface coverings are expensive and hamper dexterity. F/T sensors mounted at the wrist only detect collisions with the EEF. Lu and Chung [2005] mounted F/T sensors both at the base and the EEF permitting detection of collisions along the entire length of the manipulator. More information about surface coverings and F/T sensor collision detection can be found in Zheng and Sias [1986], Uchiyama and Kitagaki [1989], and Marks [2010].

Injury avoidance refers to methods which attempt to prevent injuries in case of collision. These techniques may control manipulator velocity based on robot mass and inertia to prevent injury in case of collision [Heinzmann and Zelinsky, 2003] or compare collision parameters, such as force, to safe limits [Ikuta and Nokata, 1999][Ikuta et al., 2001][Ikuta et al., 2003]. In themselves, Injury avoidance techniques do not detect or prevent collisions, they only maintain a state of operation intended to minimize injury should a collision occur.

At Department of Energy national laboratories like Los Alamos National Lab, radioactive materials must be handled, manipulated, mixed, separated, etc. Shielding and barriers (e.g. gloveboxes) protect workers from radiation and radioactive contamination during these processes. Gloveboxes contain radioactive contamination and help reduce radiation dose but introduce ergonomic risks. The glovebox is a steel structure that may be built in any size or shape. (Figure 1.3, left) It is designed to keep radiation and contaminants from harming workers. A box is fitted with windows for individuals to see into the box. Lead can be added to the box and the windows for additional shielding as necessary. Workers can reach in and manipulate the contents through glove ports; holes in the glovebox fitted with thick protective gloves. (Figure 1.3, right) In some cases, it is necessary to wear additional gloves over the glovebox gloves to prevent punctures and cuts. Gloveboxes permit work that would otherwise be too hazardous but the worker is still exposed to radioactive dose and introduced to new ergonomic risks.

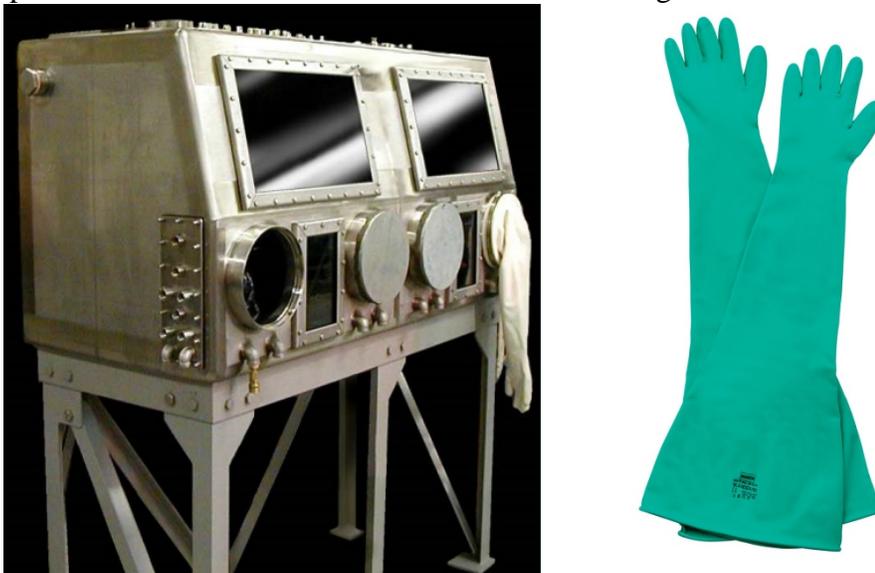


Figure 1.3: Example Glovebox [<http://bit.ly/x6TEmw>] (left) and Glovebox Gloves [<http://bit.ly/Aog9in>] (right)

Tasks in gloveboxes are sometimes automated to further remove humans from the radiation and ergonomic risks. The simplest and earliest examples of robotics in

radioactive environments are teleoperated manipulators (Figure 1.4, left) that can be controlled from outside the glovebox or from the safe side of thick concrete walls and shielding. Robots also work inside gloveboxes (Figure 1.4, right) to ease the ergonomic and radiation burdens on the human worker. [Foster et al., 2001] In these cases, human workers operate in spaces distinct from those of the robot. [Pittman et al., 2001] When a space must be accessed by man and machine, switches and sensors are used to prevent operation of the robot while a human is in the space, in accordance with ANSI standards. [ANSI/RIA, 1999] Work in gloveboxes is ideally suited for automation but automated work is hampered by the requirement that robot and human tasks be separated; glovebox automation could benefit from Human-Robot Interaction (HRI).



Figure 1.4: Typical Robots in a Radiation Facility: MANTIS Teleoperated Robot (left) [http://bit.ly/x8UVav] and Aries Robot in Glovebox (right) [http://1.usa.gov/xgnUZ2]

The Yaskawa Motoman SIA5D, like many similar industrial robots of all sizes, has motor current limiters which provide emergency collision detection. These limiters are insufficient to permit co-robotic interactions. Collision detection algorithms such as those mentioned above can provide an additional level of safety. The SIA5D does not have the additional sensors used in the examples from Marks [2010], Jung & Zelinsky [1996], Bicchi & Tonietti [2004], Zollo et al. [2002], Zheng & Sias [1986], Lu & Chung [2005], or Ralph and Pai [1995] above. However, joint positions and motor currents are available to the system controller and can potentially be used for collision detection.

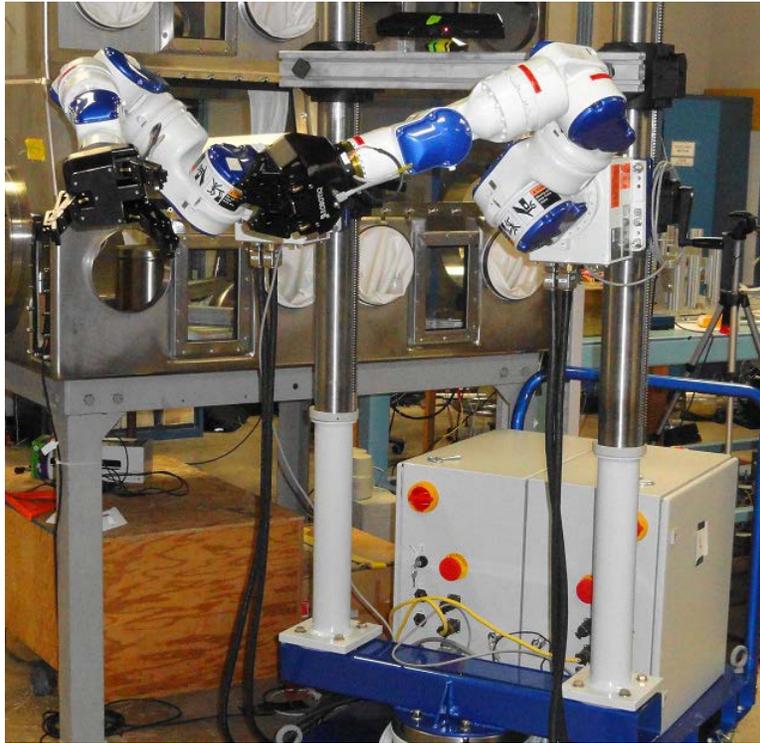


Figure 1.5: 2 Motoman SIA5D 7 Degree of Freedom (DOF) Industrial Serial Manipulators

1.1. Research goals

Collision detection (CD) provides a useful technology for all robotics applications. There are two major hurdles to implementing collision detection on real-world industrial systems performing manufacturing tasks (e.g. handling hazardous materials, in a glovebox, etc):

- Implementing CD algorithms on available and affordable hardware without expensive torque sensors.
- Understanding of the effectiveness of the CD algorithm to accurately detect collisions and prevent injuries.

These two points are addressed from engineering and research perspectives as a part of this effort. In order to address these issues, the following research objectives have been met.

- Map motor “current” measurements to joint torque for an industrial manipulator that is compatible with glovebox manufacturing.
- Analytically identify critical points where manipulators are most likely to cause damage.
- Calculate collision response time (time from initial contact until removal of contact) as function of manipulator parameters.
- Model compression and force imparted to relevant body parts during collision.
- Experimentally compare the collision response time, compression distance, and force to model.
- Quantify and/or empirically evaluate manipulator parameters (inertia, actuator power, etc.) and controller parameters (bandwidth, detection sensitivity, etc.) that impact safety during robot-human collision in a glovebox.

1.2. Approach and Outline

This work addresses the goals presented above using commercially-available hardware available at Los Alamos National Laboratory and The University of Texas at Austin. A black box model for estimating joint torque in a serial manipulator will be developed and demonstrated. The collision observer will use the estimated torques from this model. The estimated torques will be validated against the robot dynamics predicted torques and the torque due to a measured contact force.

Experimental results of collision detection with various challenging or dramatic objects (e.g. a glovebox window, a banana) will further demonstrate the application of the black box model to collision detection. Results of these tests will be used to develop and verify a simulator of the collision system. Collisions with human body parts will be simulated to evaluate the improvements to safety.

Finally, the effectiveness of the system for safely detecting and reacting to human-robot collisions will be evaluated. The simulator will be used to simulate collisions with the physical system. The critical parameters of the system, such as the end effector velocity, maximum joint acceleration, system bandwidth, etc, will be evaluated to identify the requirements to effectively detect robot-human collisions without serious injury.

Chapter 1 summarizes the specific problem addressed as well as the broader application of the research.

Chapter 2 reviews research models for common robotic joint components and develops the black box model for joint torque estimation based on motor current and joint position feedback. The technique and results of black box model parameter determination are presented.

Chapter 3 validates the black box estimated joint torques on the SIA5D manipulator during 6 DOF motion and while making contact with the environment.

Chapter 4 reviews collision detection methods from the literature. A collision detection method using black box estimated torques is demonstrated and evaluated.

Chapter 5 develops a collision simulator and validates it against data taken with the SIA5D.

Chapter 6 examines the effectiveness of the collision detection with regards to human safety. The simulator is used to examine relative gains to effectiveness given changes to system parameters, i.e. operating conditions or hardware and software capabilities.

Chapter 7 presents conclusions of this work and ideas for future work.

2. BLACK BOX MODEL FOR ESTIMATING JOINT TORQUE

Joint torque feedback is important in many robotic control techniques for human robot interaction, e.g. collision detection, compliant control. Joint torque sensors are frequently not present on industrial serial manipulators. Many systems employed in industry are position controlled as shown in Figure 2.1. A typical system has a low-level motor control that operates at high frequencies, at least an order of magnitude greater than the high level position control. The commercial system then has a higher-level controller that allows the user to set desired joint or Cartesian positions. Feedback at this level is slower but still quite fast. Control, position, and current feedback frequencies as high as 250Hz and 1000Hz are achievable on the 3rd party Agile Planet controllers used in this work. [Agile Planet, 2013]

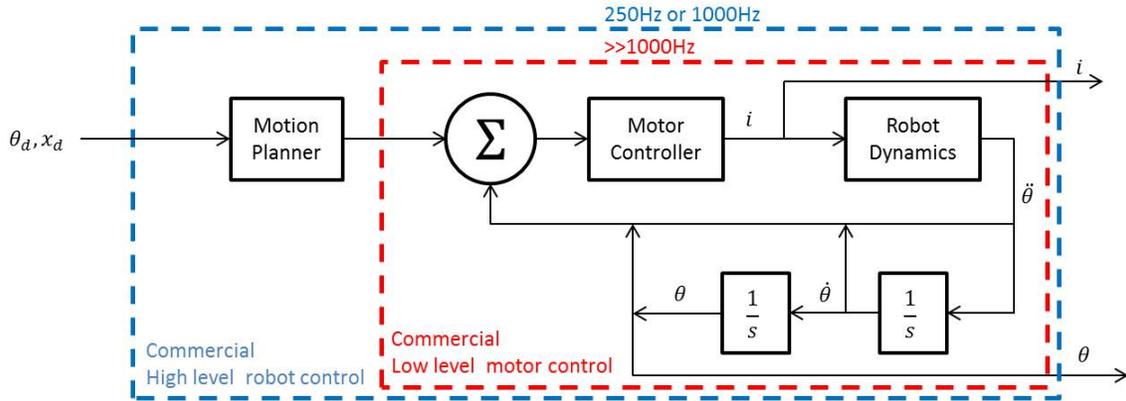


Figure 2.1: Example commercially-available position-controlled robot system

To implement torque feedback, a black box model for estimating joint torque is proposed and demonstrated here. The model is designed for a commercially available, position controlled serial manipulator for which detailed information about the joint (i.e. gear ratios, motor constants, etc) is proprietary. To address this issue, a literature review of harmonic drive and friction modeling is presented. The modeling techniques for these

major components are considered when creating the black box model for joint torque estimation.

The objective of this chapter is to develop a model assuming realistic, limited access to model parameters and feedback due to the closed, proprietary nature of the system. Of course additional data could be acquired by additional sensors but this requires an unacceptable amount of robot disassembly. Several models for harmonic drive gears and joint friction are discussed in the next section. Measurements for these models require unacceptable disruption of the closed, commercial system. They cannot be duplicated here but they will be used as a basis for the development of the black box model for joint torque estimation.

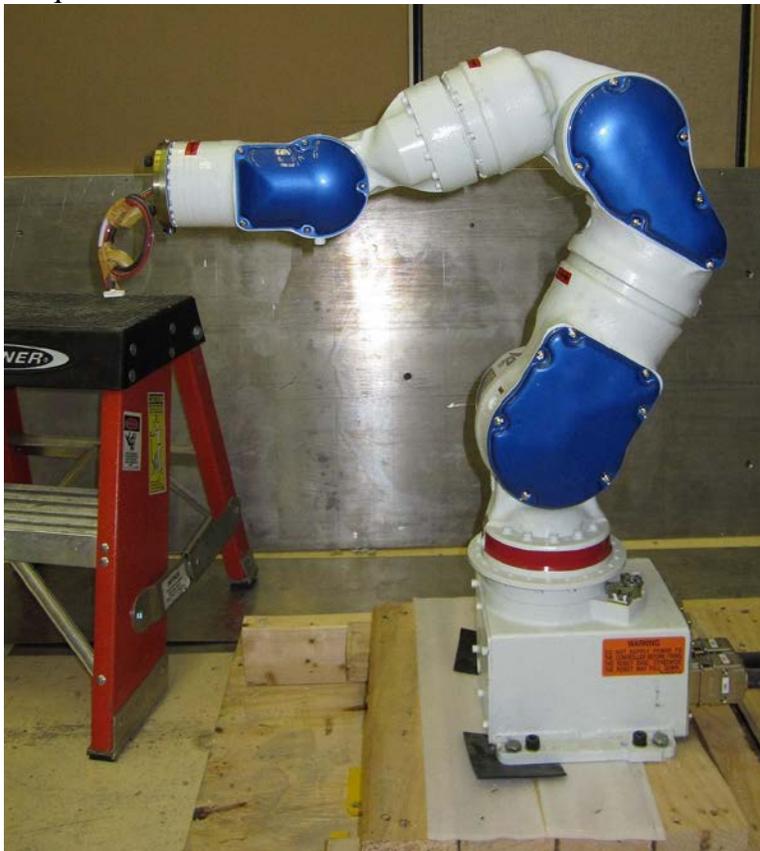


Figure 2.2: Yaskawa Motoman SIA5D 7 degree of freedom 5kg payload serial manipulator

The next two sections are literature review. In the first of these, methods of joint modeling are examined as a basis for *estimating* joint torque. The second section reviews accepted methods for *predicting* the joint torque. In the third section, the black box model for estimating joint torque is developed. The fourth section describes the experimental procedure for estimating the black box model parameters. Lastly, the results of parameter determination are presented.

2.1. Joint Modeling Literature Review

The harmonic drive (Figure 2.3) is widely used in industrial serial manipulators due to negligible backlash, compact design, and a high torque-to-weight ratio. The drive's key components are the wave generator, the circular spline, and the flexspline. The configuration most advantageous to robotics (high torque/low speed output) uses the flexspline as output and the wave generator as input. The circular spline has a rigid shape and is fixed in position relative to the reference link of the joint. The joint motor drives the wave generator. The wave generator deforms the flexible flexspline. As the wave generator rotates, it forces the slightly misaligned teeth to mesh. A tangential force is generated causing the output to rotate.

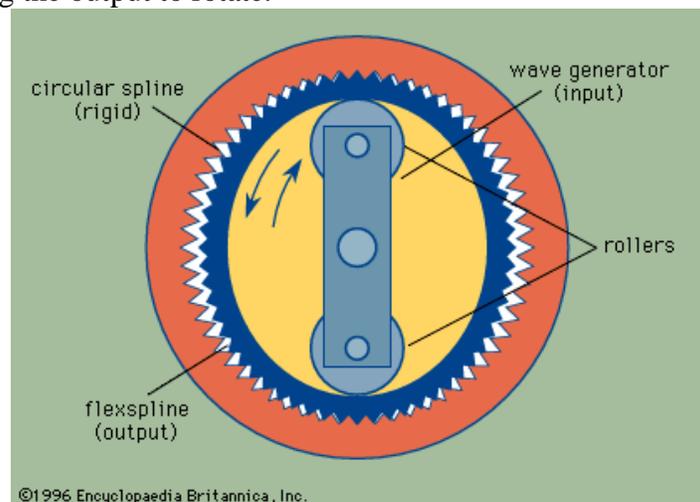


Figure 2.3: Harmonic drive Gear Train

Several researchers have developed non-linear models and parameter estimation techniques for harmonic drives. The model developed by Seyfferth and Angeles includes friction (dynamic and startup), compliance, and hysteresis. Model parameters were identified by a least-squares fit of torque-torsion data. The experimental setup differs from that available in the SIA5D. They measured the angle and torque at both the input and the output instead of only the output angle and input motor current as is done in this work. They do calculate the drive motor torque from measured input current instead of using a torque sensor. This is possible in their experiments because they use a motor with known parameters. In this work, the motor parameters are unknown.

Seyfferth & Angeles [1995] and Seyfferth, Maghazi, and Angeles [1995] model the gear train similar to any other gear train; torque is required to accelerate each gear and to overcome friction losses. The remainder is transmitted between gears and then to the robot joint. The relationships (torque in/torque out, inertial torque/acceleration, etc) are all linear except the friction loss. They model the friction loss as

$$B_m = B_{m0} \text{sgn}(\dot{\theta}) + b_1 \dot{\theta} + b_2 \dot{\theta}^2 \quad (2.1)$$

where B_m is the total torque due to friction losses, B_{m0} is the constant friction torque, b_1 is the linear coefficient, and b_2 is the quadratic term. All terms are dependent on the joint velocity. When considering conversion from the input current to output torque, some torque is required to accelerate the motor, input gear, and output gear. These are all linearly dependent on the joint acceleration.

Tuttle and Seering [1996] model a harmonic drive and include kinematic error, compliance, and geometric and Coulomb friction at the gear interface. Kinematic error is a function of the gear input and output angles. Since the output angle is controlled by the low-level controller, the influence of kinematic error on the black box model is expected to be negligible. In addition to velocity dependent friction modeled by Seyfferth and

Angeles, this model also includes position-dependent friction. Vibrations and resonances in the harmonic drive are also included. Output velocities “hang” at resonances and then suddenly jump out of them. In their system, resonances were experienced when applying a step input. For position controlled systems, the input is adjusted to achieve the desired result.

The two equations below [Tuttle and Seering, 1996] model the angular acceleration of the wobble gear (wave generator) and the flexspline of the harmonic drive. In both of these equations, the losses are linearly dependent on the velocity of the wave generator and flexspline.

$$\frac{d}{dt}\dot{\theta}_{wg} = \frac{1}{J_{in}}(K_t i_m - b_{in}\dot{\theta}_{wg} - T_{wg}) \quad (2.2)$$

$$\frac{d}{dt}\dot{\theta}_{fs} = \frac{1}{J_{out}}(-b_{out}\dot{\theta}_{fs} - T_{fs}) \quad (2.3)$$

Tuttle [1996] found that when in motion, the friction has constant, velocity-dependent, position-dependent, and resonance vibration terms. Velocity-dependent and constant terms are modeled with a cubic (instead of quadratic as in Seyfferth et al. [1995][1995]) function. The position-dependent terms are modeled as a sinusoid with the same period as the flexspline (i.e. output). This friction model is considerably more complex than the Seyfferth, et al. models which included only constant and velocity-dependent terms.

Taghirad and Belanger [1998] use simple models for compliance, hysteresis, and friction. Unknown parameters are estimated from least square approximations applied to experimental data. They report better results using linear stiffness and velocity-dependent damping than the more complicated models. Friction in the harmonic drive is modeled having a constant term and a linear dependence on joint velocity. Unique parameters are identified for motions in positive and negative directions.

Vakil, Fotouhi, & Nikiforuk [2010] present a method for determining the friction parameters for a robot joint. They review several general friction models but settle on the following model for robot torque modeling. The friction torque, $\tau_{friction}$, is modeled as

$$\tau_{friction} = \left(\tau_c + (\tau_{st} - \tau_c) e^{-(\dot{\theta}/v)^2} \right) sgn(\dot{\theta}) + \sigma \dot{\theta} \quad (2.4)$$

where $\dot{\theta}$ is the rotational joint velocity, τ_c , τ_{st} , v , and σ are the Coulomb friction, static friction, Stribeck velocity constant, and viscous damping, respectively. In this model, frictional memory and rising static friction are assumed negligible. They end up with terms for Coulomb and static friction, the Stribeck velocity constant, and viscous damping. The friction components are identified from the work done by the input torque in a motion between rest positions. The technique uses a strictly positive or negative torque input to move the joint. The commercial off-the-shelf system does not immediately provide this information but, as with the harmonic drive models, fundamentals from the model will be adapted to the black box model developed in this work. Of particular note: the terms of the Vakil model are velocity-dependent or constant. The Coulomb friction term is constant while the other terms have an exponential and linear relationship to the velocity.

In all these efforts, researchers placed sensors before and after the gear train. In this work, we will not have access to the same data. The objective here is not to accurately model the gear train but to estimate the effects of the harmonic drive, friction, hysteresis, motor constant, etc. as a black box. We will estimate joint torque from input current on a commercially available serial manipulator without disrupting its construction. Of particular note from the literature is that friction is modeled with a nonlinear dependence on the joint velocity.

2.2. Calculated Dynamics

In the next section, the black box model for estimating joint torques will be developed. The determination of the model's parameters utilizes predicted torques as calculated by one of the methods presented in this section. The predicted joint torques are also essential to the model validation in the next chapter and to collision detection.

The predicted joint torques can be expressed as

$$\tau = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) + \tau_{contact} \quad (2.5)$$

where $M(\theta)$ represents the inertia matrix, i.e. the dependence on joint acceleration, $V(\theta, \dot{\theta})$ contains the Coriolis terms, and $G(\theta)$ contains the gravity terms. There are several methods for calculating the joint torques required for a particular kinematic state (joint position, velocity, and acceleration) and contact forces of a manipulator. Two approaches are commonly used for calculating the torque required to achieve the desired kinematic state: 1) a recursive Newton-Euler algorithm and 2) an energy-based Lagrangian method. More detailed discussion of both algorithms can be found in Craig [2005].

The Lagrangian method can be used to calculate the joint torques required for the desired kinematic state. The kinetic and potential energy of the manipulator are equal to the sum of energies for the links. The link kinetic (k) and potential energies (u) are

$$k(\theta, \dot{\theta}) = \frac{1}{2}mv_C^T v_C + \frac{1}{2}\omega^T {}^C I \omega \quad (2.6)$$

and

$$u(\theta) = -mg^T P_C + u_{ref} \quad (2.7)$$

where m is the link mass, v_C is the velocity of the center of mass, ω is the angular velocity, ${}^C I$ is the inertia tensor, g is the gravity vector, P_C is the vector from the link

origin to the center of mass, and u_{ref} is a reference potential energy such that u is always positive.

The Lagrangian is

$$\mathcal{L}(\theta, \dot{\theta}) = k(\theta, \dot{\theta}) - u(\theta) \quad (2.8)$$

and the torque is shown below.

$$\tau = \frac{d}{dt} \frac{\partial k}{\partial \dot{\theta}} - \frac{\partial k}{\partial \theta} + \frac{\partial u}{\partial \theta} \quad (2.9)$$

The Lagrangian method requires the derivatives of the energy functions be taken with respect to each of the joint position and velocity variables. The energy functions must be written with linear and angular velocities in a common frame. These velocities are commonly found by outward (base to EEF) iterations. However, this makes it difficult to write the energy equations in easily-differentiable forms.

Alternatively, the joint torques due to gravity, velocity, and acceleration can be calculated by the Newton and Euler equations of motion.

$$F = m\dot{v} \quad (2.10)$$

$$N = {}^c I \dot{\omega} + \omega \times {}^c I \omega \quad (2.11)$$

These equations, applied recursively, identify the forces and moments at each joint. The velocity, v , and angular velocity, ω , are the linear and rotational velocities of the link. The force and moment calculated are each 3 component vectors. In the robot, the joint torque is the moment about the joint axis. Like the Lagrangian method, the velocities are calculated by outward iterations. However, the Newton-Euler method does not require differentiation. An inward (EEF to base) iterative method is employed to calculate the forces and moments.

The torques due to contact forces, $\tau_{contact}$, can be estimated as

$$\tau_{contact} = J^T F_{contact} \quad (2.12)$$

where J^T is the transpose of the Jacobian at the point of contact and $F_{contact}$ is the contact force in the same frame as the Jacobian. Given a known contact force and manipulator configuration, the torque in each of the joints due to that contact force can be calculated.

The robotics software, Operational Software Components for Advanced Robotics (OSCAR) was developed at The University of Texas at Austin. [Kapoor, 1998] OSCAR can perform joint torque calculations using the Newton-Euler method. Joint torques due to contact forces at the end effector can also be calculated with OSCAR. These libraries are used in this work to calculate the predicted gravity, Coriolis, and acceleration torques.

2.3. Black Box Model

Based on the harmonic drive and joint friction literature a model is developed here. A few assumptions about the joint are made before developing the model.

- Motors with nearly linear current-to-torque characteristics
- Harmonic drive gears with velocity-dependent friction characteristics
- When no current is supplied to the motor, it exerts no torque

The first assumption is used as a basis for developing the model but the nature of the black box model does not necessarily require it. The last assumption is fundamental to the model – a motor drawing no current should generate no torque.

The exact design of the robot joint is unknown; it is proprietary information. However, based on common robot design practices, it is expected the joint consists of a DC motor, a harmonic drive gear train, and an output to the joint. (Figure 2.4)

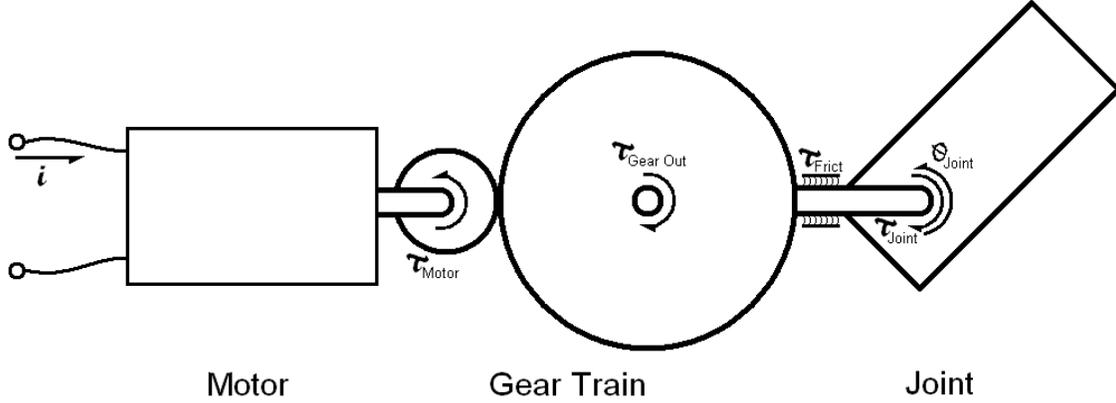


Figure 2.4: Joint Model

Only the motor current, i , and the joint angle, θ_{joint} , are accessible, from industrial manipulator controllers such as the one shown in Figure 2.1.

Based on the first assumption, the motor output torque, τ_m , may be estimated as

$$\tau_m = Ki - I_m \ddot{\theta}_m - T_{fm} \quad (2.13)$$

where K is the motor constant, i is the input current, I_m is the motor inertia, $\ddot{\theta}_m$ is the motor acceleration, and T_{fm} is the motor friction. The gear output torque, τ_{GO} , would then be

$$\tau_{GO} = N\tau_m - T_{fg} \quad (2.14)$$

given a gear ratio of N and gear friction loss of T_{fg} . Assuming negligible kinematic error, the motor acceleration can be calculated from the output acceleration which is the same as the joint acceleration, $\ddot{\theta}_{joint}$.

$$\ddot{\theta}_m = N\ddot{\theta}_{joint} \quad (2.15)$$

The gear output torque can be rewritten.

$$\tau_{GO} = NKi - N^2 I_m \ddot{\theta}_{joint} - T_{fm} N - T_{fg} - I_{GI} N \ddot{\theta}_{joint} - I_{GO} \ddot{\theta}_{joint} \quad (2.16)$$

Combining like terms and accounting for any friction between the gear output and joint, T_{fj} . The gear output torque is the link joint torque.

$$\tau_{joint} = NKi - \ddot{\theta}_{joint} (N^2 I_m - N I_{GI} - I_{GO}) - T_{fm} - T_{fg} - T_{fj} \quad (2.17)$$

Based on the models for harmonic drives and robot joint friction from the literature, the major component of friction losses, $\tau_{friction}(\dot{\theta}_{joint})$, is dependent on the joint velocity. The inertias can be combined into a lumped inertia, I_{lumped} .

$$\tau_{joint} = KNi - \ddot{\theta}_{joint}I_{lumped} - \tau_{friction}(\dot{\theta}_{joint}) \quad (2.18)$$

The initial model indicates the “loss”, the torque generated by the motor which isn’t present at the joint, is dependent on the joint velocity and acceleration.

The Newton-Euler method can be used to predict the joint torque from the joint position, velocity, and acceleration. This predicted torque does not account for torque due to external forces, so it cannot be used to directly estimate the torque during HRI. It will be used to estimate the model parameters for the black box model and to predict the uncollided torque during collision detection.

The Newton-Euler predicted torque, τ_{pred} , is compared to the black box model estimated torque, τ_{est} .

$$\tau_{pred} \approx \tau_{est} = KNi - \ddot{\theta}_{joint}I_{lumped} - \tau_{friction}(\dot{\theta}) \quad (2.19)$$

If the joint is moving at a constant velocity, the acceleration term is zero. If the predicted torque is zero, any motor torque is “consumed” in the black box model. So when the predicted torque is zero, the motor current represents the system losses, i.e. friction and inertia. Because the joint is not accelerating the inertia related term is zero.

$$0 = KNi(\tau_{pred} = 0) - i_{friction}KN \quad (2.20)$$

The motor current, less the current lost to friction, is converted by the motor into the estimated joint torque at that constant velocity. The torque during a constant-velocity motion is then estimated as

$$\tau_{est} = f(i - i_{friction}) \quad (2.21)$$

In the models reviewed, there is a constant friction term that is dependent on the joint velocity direction. In the black box model, this term is estimated, in terms of joint torque, as a hysteresis term. It is dependent on the last non-zero joint velocity, $\dot{\theta}_H$.

$$\tau_H = \begin{cases} T_H \text{sgn}(\dot{\theta}_H) & |\dot{\theta}| = 0 \\ 0 & |\dot{\theta}| > 0 \end{cases} \quad (2.22)$$

The hysteresis parameter, T_H , is determined by moving the joint to a position, noting the direction of approach, then moving another joint and observing the difference between the estimated and predicted torque. In practice, a threshold is used instead of an absolute zero.

$$\tau_H = \begin{cases} T_H \text{sgn}(\dot{\theta}_H) & |\dot{\theta}| \leq \dot{\theta}_{thresh} \\ 0 & |\dot{\theta}| > \dot{\theta}_{thresh} \end{cases} \quad (2.23)$$

Finally, the estimated torque can be written as follows.

$$\tau_{est} = f(i - i_{friction}) - \ddot{\theta}_{joint} I_{lumped} - \tau_H \quad (2.24)$$

The torque estimation model takes the input current and the joint position as input from the commercial controller at a rate of 250 or 1000Hz but the motor controller is operating at a much faster rate. The commercial system and the black box estimator are shown in Figure 2.5.

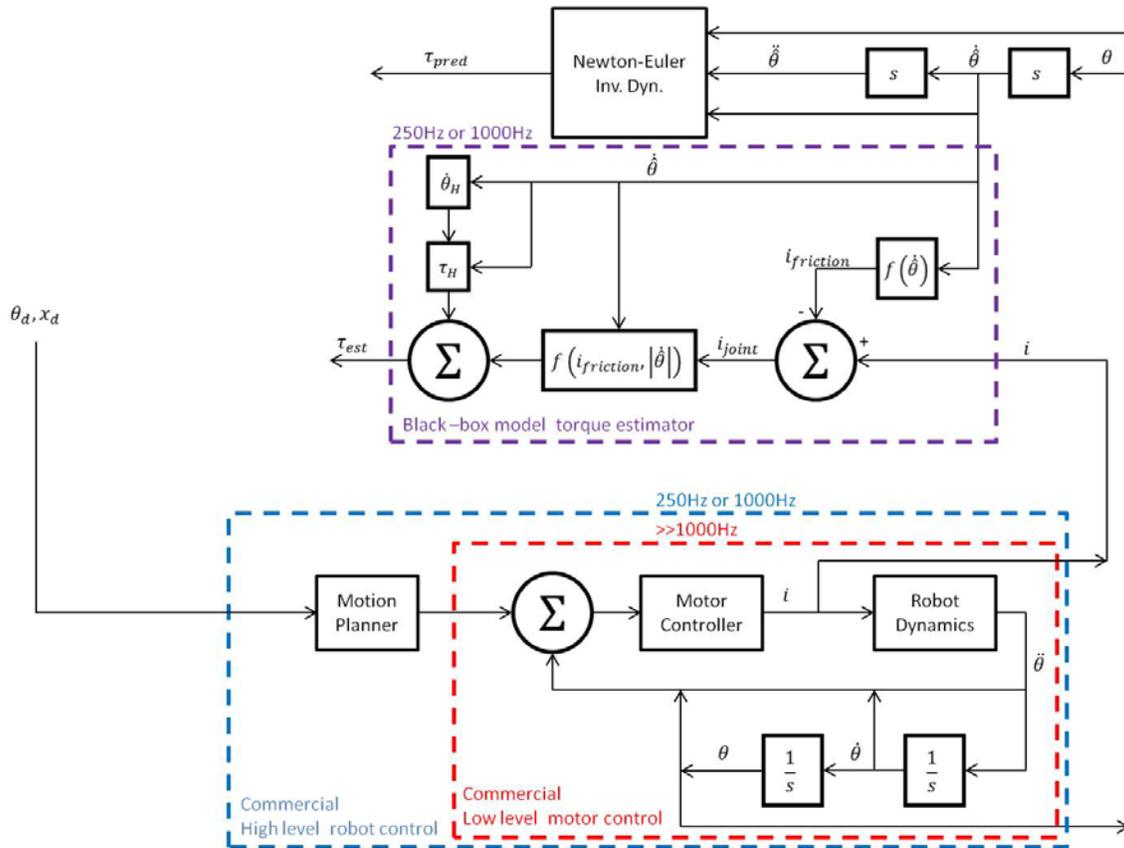


Figure 2.5: Commercial system with black-box model torque estimator

The system also estimates the joint velocities and accelerations from the joint positions for black box estimation as well as the Newton-Euler torque prediction. The procedure for estimating the black box model parameters is described in the next section.

2.4. Parameter Estimation Setup

The model presented above was verified using Yaskawa Motoman SIA5D 7 DoF serial manipulators. The model was verified on two different SIA5D manipulators, one at The University of Texas at Austin, the other at Los Alamos National Laboratory. Agile Planet controllers for the manipulators operated at 250Hz in Austin and 1000Hz in

Los Alamos. Parameters for joints 1 through 6 are estimated. Joint 7 parameters are not estimated because no suitable end effector² is present.

In this section, the method for estimating the parameters of the black box torque estimating model is presented. The procedure is described by detailing the parameter estimation technique for joint 2. The parameters for the other joints were found by the same means. The parameters for all joints will be presented in the next section. The black box model estimated torques will be validated in the next chapter.

The parameters of the black box model (2.24) are estimated one at a time. When the joint is moved at constant velocity, the acceleration is zero and the hysteresis is zero so only the friction loss and the relationship between current and torque are unknown.

$$\tau_{est} = f(i - i_{friction}) - \ddot{\theta}_{joint} I_{lumped} - \tau_H \quad (2.24)$$

First the friction parameters are estimated. The procedure is briefly summarized in this paragraph then the details are presented. The joint is moved several times at constant velocity (Table 2.1, Figure 2.6). The friction at each joint velocity is estimated by fitting a function to the measured and predicted data. Each friction estimate and the associated velocity form a new data set. The friction modeling function is fit to these points.

² The same method could be used to estimate joint 7 parameters given an end effector with a center of mass which does not lie on the joint axis. While it is possible to devise an experiment with a test fixture on the end-effector, such efforts are not necessary since a black box model for the final joint was not used as part of the collision detection algorithm for the system.

Table 2.1: Friction parameter determination, repeated 15 times

Start position	Stop position	Speed
90°	-90°	-10%
-90°	90°	10%
90°	-90°	-20%
-90°	90°	20%
90°	-90°	-30%
-90°	90°	30%
.	.	.
.	.	.
.	.	.
90°	-90°	-90%
-90°	90°	90%
90°	-90°	-100%
-90°	90°	100%

Joint 2 was positioned so that the axis was perpendicular to the gravity vector and such that the range of expected torques passed through zero. (Figure 2.6) This is necessary to determine the current lost to friction as described above. The joint is moved at constant velocity from +90° to -90° at 10% of the maximum joint velocity. The motion is repeated with the equal and opposite joint velocity. The +90° to -90° then -90° to +90° motions are repeated at each 10% velocity increment until the motions are completed at 100% joint velocity. (Table 2.1) This process is repeated 15 times for a total of 300 constant velocity data sets for each joint. Figure 2.6 shows several frames of one such constant velocity motion.

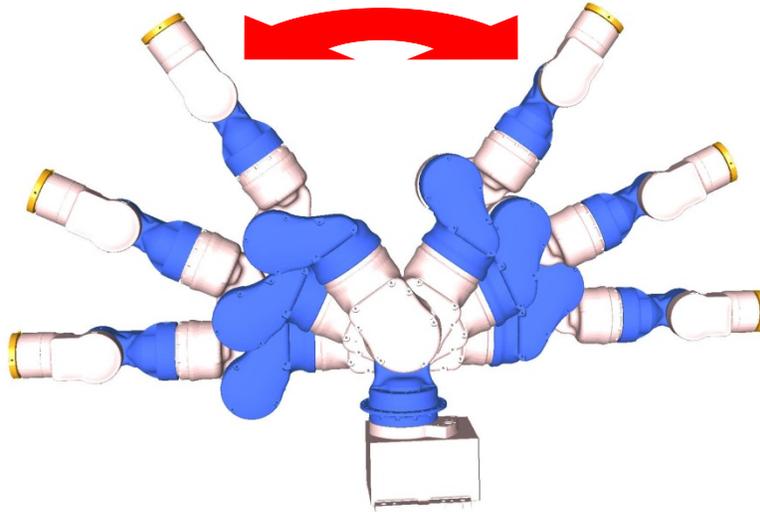


Figure 2.6: Joint 2 friction characterization motion, side view

The current was measured and recorded during the motions. The joint position was used to estimate the joint velocity and acceleration. The iterative Newton-Euler method was employed via OSCAR to calculate the predicted joint torques. For each constant velocity motion, a curve was fitted to the predicted torque/measured current data. Eight examples are shown in Figure 2.7.

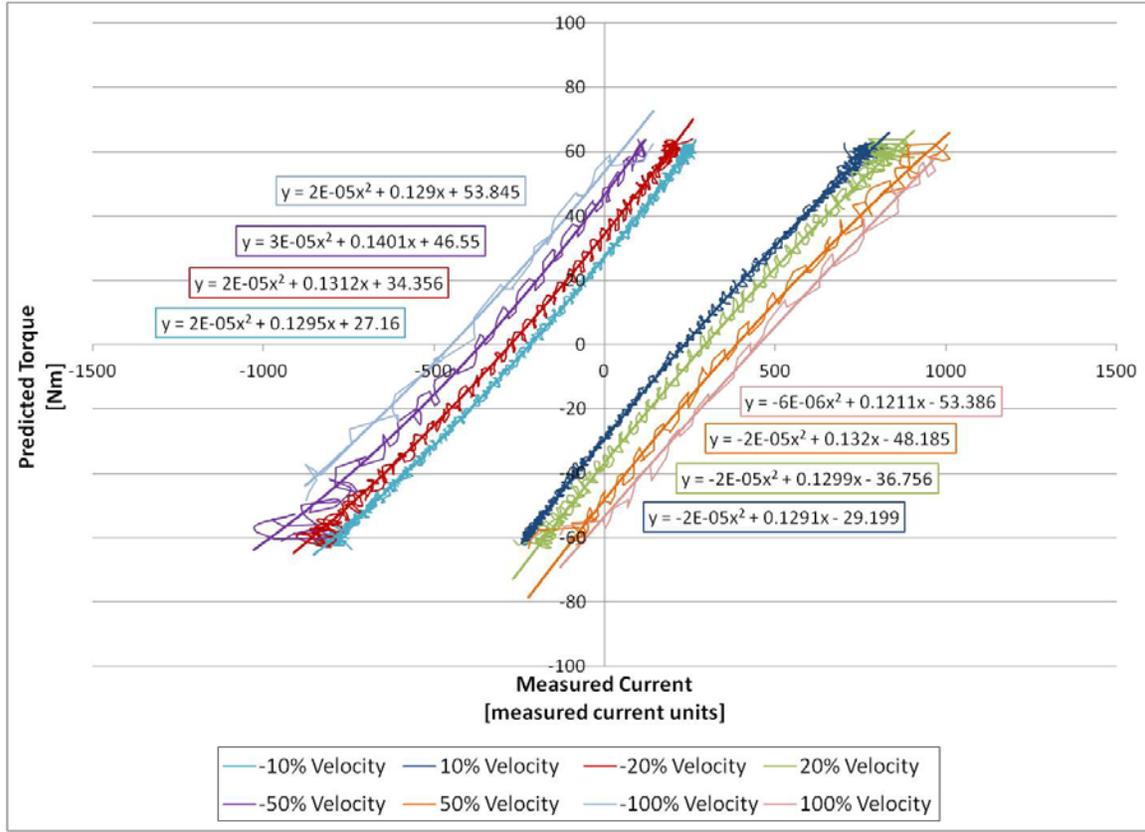


Figure 2.7: Joint 2: Example predicted torque versus measured current for determining velocity dependent losses

As mentioned in the previous section, the current lost to friction is the measured current when the predicted torque is zero, i.e. the x-axis crossing point. The relationship between the input current and the torque is quadratic.

$$\tau_{pred} = -sgn(\dot{\theta}_{joint})A_{CV}i^2 + B_{CV}i + C_{CV} \quad (2.25)$$

This equation is fitted to each of the constant velocity motion data sets. Eight examples of (2.25) are shown with the constant velocity data in Figure 2.7.

The friction current can be found from the quadratic formula applied to the coefficients of equation (2.25).

$$i_{friction}[\dot{\theta}_{joint}] = \frac{-B_{CV} \pm \sqrt{B_{CV}^2 - 4A_{CV}C_{CV}}}{2A_{CV}} \quad (2.26)$$

Equation (2.26) is applied to each of the three hundred fits of (2.25) to a constant velocity motion. Each solution to (2.26) is an estimate of the friction loss at a particular joint velocity. Each friction estimate is plotted against its corresponding joint velocity as a green point in Figure 2.8. Applying (2.26) to each of the example curves in Figure 2.7 yields one of the green points in Figure 2.8. The model for the velocity dependent friction loss is found by fitting a curve to the data.

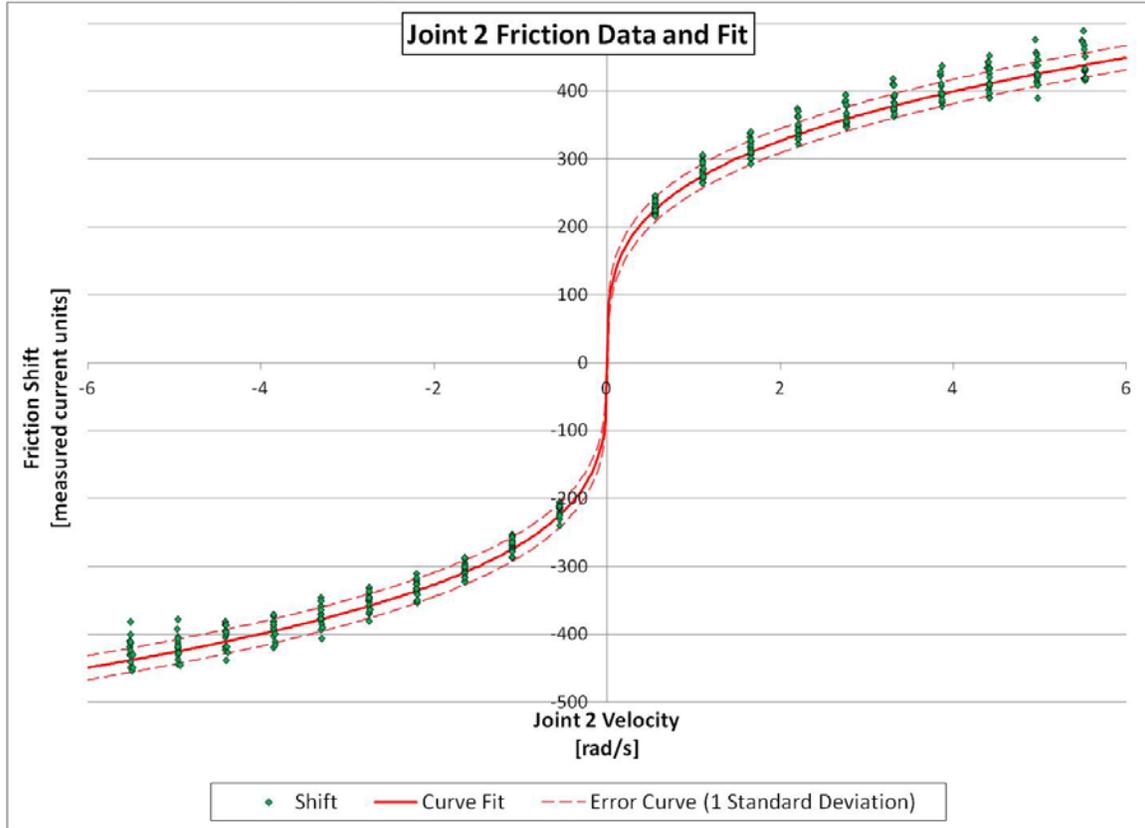


Figure 2.8: Joint 2 friction current versus joint velocity

The characteristic equation for the data in Figure 2.8 is found to be of the following form³.

$$i_{friction}(\dot{\theta}_{joint}) = sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \quad (2.27)$$

³ In practice, the absolute values of friction and velocity were used to determine the friction parameters because of the sign function.

The friction parameters, D and F , for joint 2 were found to be 267.345 and 0.2896, respectively.

The joint current, i_{joint} , for each constant velocity motion was determined by subtracting the friction current, as estimated by equation (2.27), from the measured current, i .

$$i_{joint} = i - i_{friction} \quad (2.28)$$

The *joint* currents for the same 8 constant velocity data sets shown in Figure 2.7 (*measured* current) are shown in Figure 2.9.

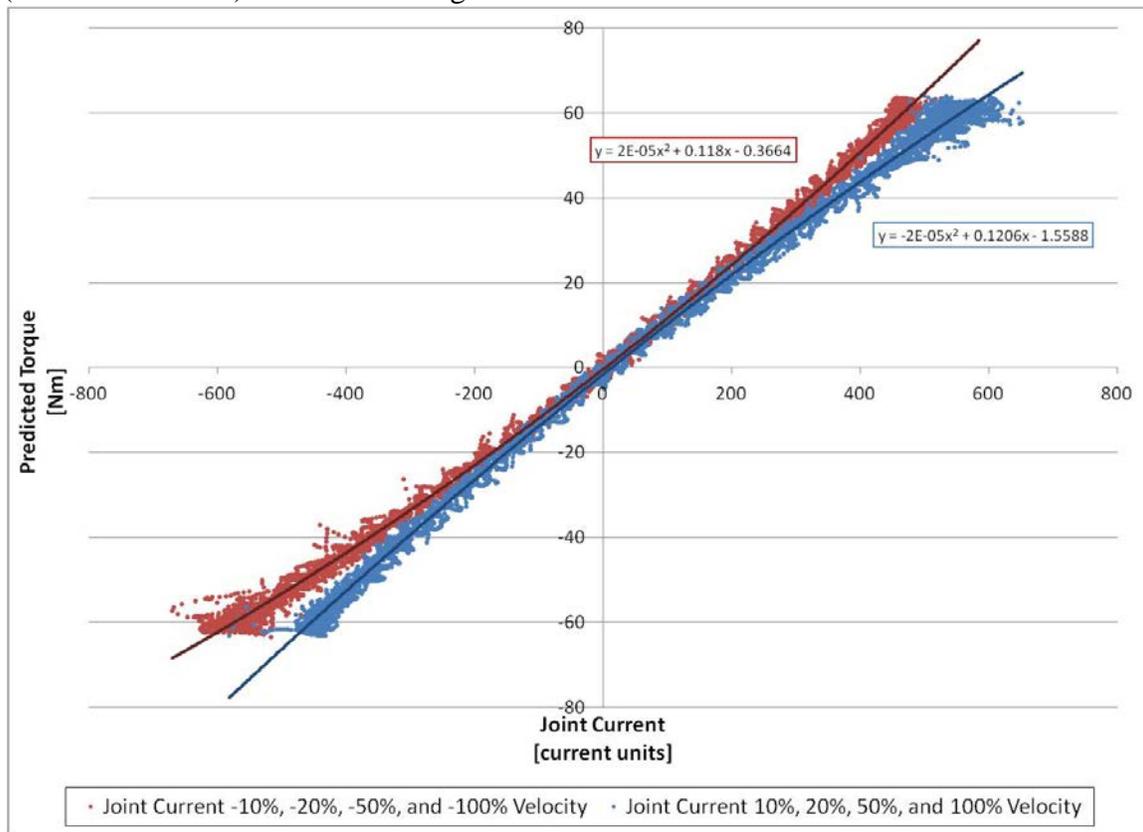


Figure 2.9: Joint currents for sample constant velocity motions

A curve was fitted to the joint currents to determine the relationship between joint current and joint torque. The relationship between joint current and joint torque is quadratic, as in equation (2.25).

$$\tau_{est, \ddot{\theta}=0} = f(i - i_{friction}) = -sgn(\dot{\theta}_{joint})Ai_{joint}^2 + Bi_{joint} + C \quad (2.29)$$

The model parameters A , B , and C are estimated for constant velocity motions in each direction (example in Figure 2.9). There was too much data to fit a single curve to all the positive or negative velocity data. The final parameters are determined by averaging the parameters from each of the data sets. The parameter A is dependent on the direction of motion but only in sign; the magnitude is the same. The parameter B is the same regardless of direction. The constant, C , is small and is dropped because of the third assumption; when all the current is “consumed” by friction ($i_{joint} = 0$), the joint torque is zero.

The estimated joint torque without the restriction of constant, non-zero velocity becomes.

$$\begin{aligned} \tau_{est} = & A \left(i - sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \right)^2 \\ & + B \left(i - sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \right) - \ddot{\theta}_{joint}I_{lumped} \\ & - \tau_H \end{aligned} \quad (2.30)$$

The remaining unknown terms are the lumped inertia and the hysteresis term. The hysteresis term is estimated using data when the joint velocity is zero. Joint acceleration is also zero. The hysteresis term can be identified in terms of the predicted torque and estimated model parameters.

$$\begin{aligned} -\tau_H = & \tau_{pred} - A \left(i - sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \right)^2 \\ & - B \left(i - sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \right) \end{aligned} \quad (2.31)$$

The hysteresis term is estimated by moving the joint to a position then measuring the estimation error while moving another joint. For joint 2, joint 7 was moved because it affected almost zero change in the predicted torque of joint 2. The manipulator was moved into a vertical position before starting the measurements. Then joint 7 was oscillated for 50 seconds. The hysteresis term, using eq. (2.31) is plotted in Figure 2.10

for 5 tests in each direction. “Decreasing Angle” tests moved joint 2 to 0° from a positive angle. “Increasing Angle” tests approached from a negative angle.

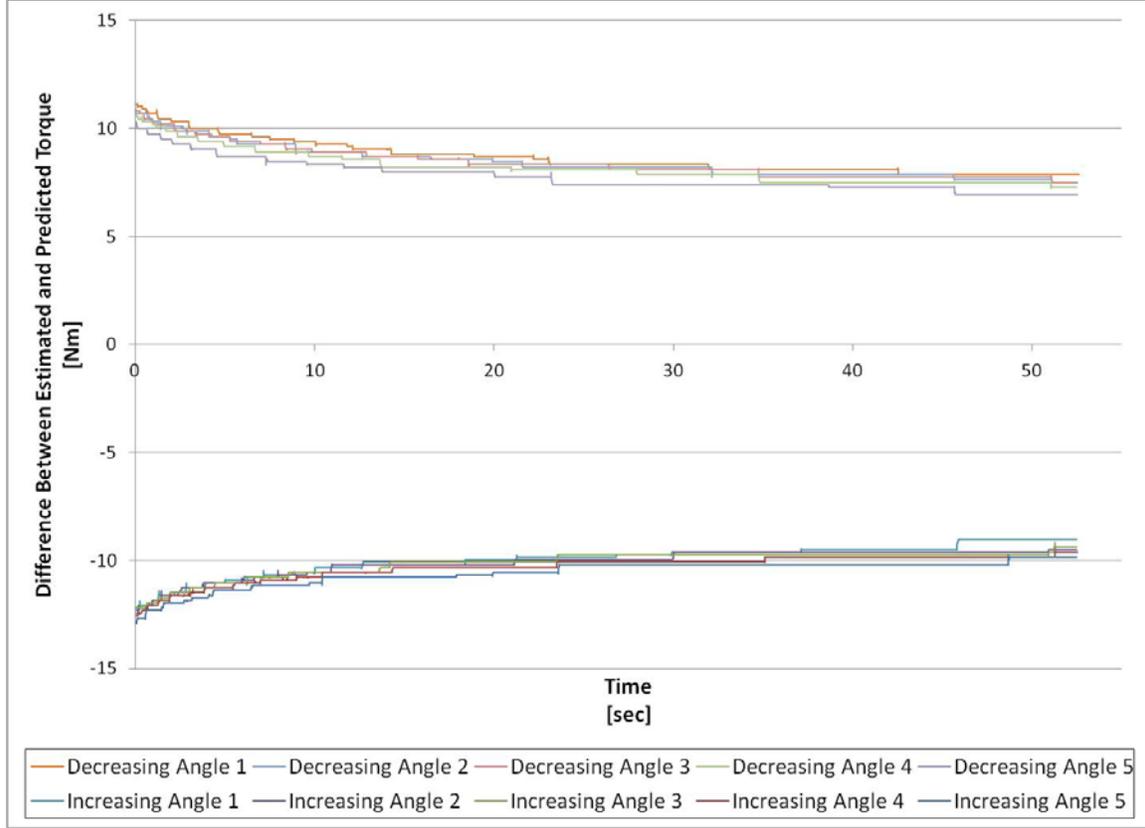


Figure 2.10: Joint 2 hysteresis example data

The hysteresis term depends only on the direction from which the stopped position is reached. The hysteresis term is identified in the plot above as the asymptote of the data set. Combining the data from each test yields an estimated hysteresis term, T_H , of 8Nm.

Lastly, the acceleration term is evaluated. The model equation can be rearranged.

$$\begin{aligned}
 \tau_{est} - A \left(i - \text{sgn}(\dot{\theta}_{joint}) D |\dot{\theta}_{joint}|^F \right)^2 \\
 - B \left(i - \text{sgn}(\dot{\theta}_{joint}) D |\dot{\theta}_{joint}|^F \right) + \tau_H \\
 = \ddot{\theta}_{joint} I_{lumped}
 \end{aligned} \tag{2.32}$$

The only unknown is the lumped inertia. The equation has the form

$$y = mx \tag{2.33}$$

The left hand side of equation (2.23) can be evaluated and plotted as y . The joint acceleration is x . A line fitted to the data for joint 2 is shown in Figure 2.11.

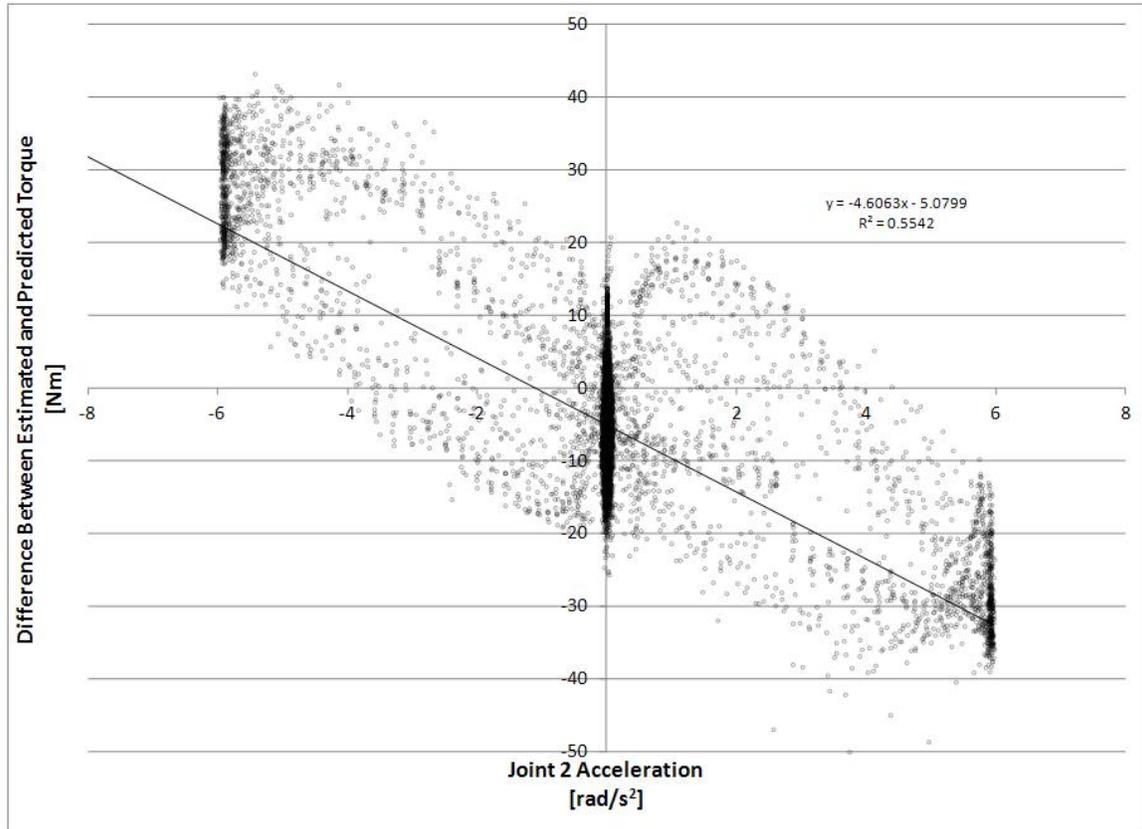


Figure 2.11: Example lumped inertia parameter graph

The data has a downward linear trend as demonstrated by the fit and expected by the model, but there is a large variation in the error at any given acceleration. The R-squared value associated with the fit is only 0.5542. It will be shown in the next chapter that the results while omitting the lumped inertia term are quite good. The inertia/acceleration term will be discussed in further detail in the Future Work section.

2.5. Parameter Estimation Results

The technique for estimating joint 2 parameters was presented in the previous section. Here the results for all joints are presented. The black box model parameters for

each of the joints is estimated in the same way as joint 2 above. The friction current versus joint velocity plots are shown in Figure 2.12.

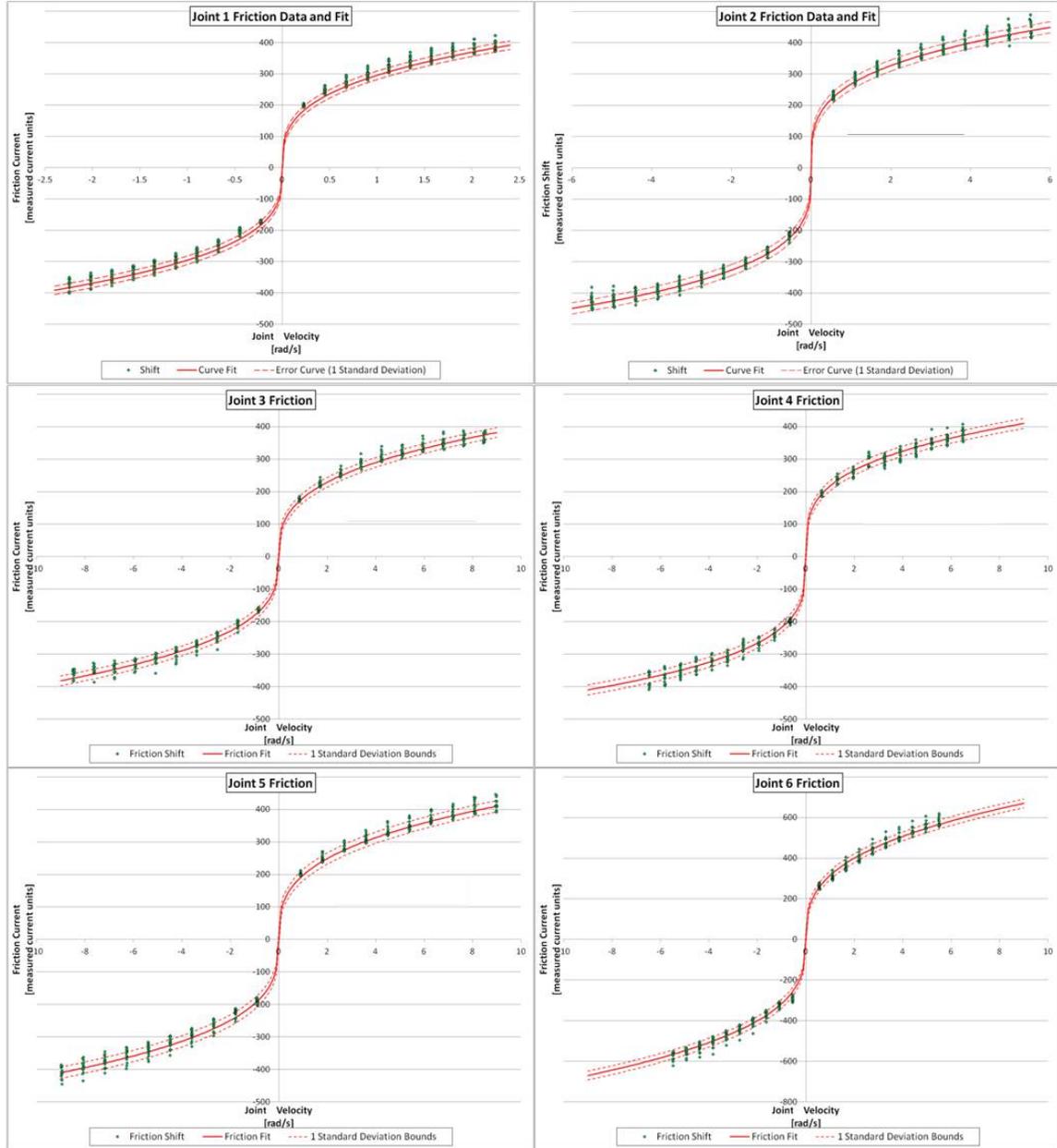


Figure 2.12: Friction model data, curves, fits, and uncertainty

Velocity dependent friction was characterized by the same equation for all joints.

$$i_{friction} = sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \quad (2.27)$$

To evaluate the fit of the curve, the uncertainty was calculated as the standard deviation of the difference between the measured and predicted friction current. The uncertainty in each of the estimates was found to be less than 22 current units for all joints. The resulting estimated torque uncertainty is dependent on the measurement because of the non-linearity in the mapping from joint current to torque. The dashed lines in Figure 2.12 indicate 1 standard deviation above and below the estimated friction current.

The black box parameters for all joints are shown in Table 2.2.

Table 2.2: Black Box Model Coefficients

	Mapping Coefficients		Friction Model		Friction Uncertainty	Hysteresis T_H
	A i_{joint}^2	B i_{joint}	D $ \dot{\theta} $	$ \dot{\theta} ^F$		
Joint 1	3.36E-05	0.112	295.724	0.322	13.6	-4
Joint 2	2.26E-05	0.117	267.345	0.290	17.6	-8
Joint 3	3.30E-05	0.094	233.976	0.327	14.9	-3
Joint 4	3.93E-05	0.065	217.814	0.288	15.4	-4
Joint 5	1.75E-04	0.022	285.269	0.334	17.3	-5
Joint 6	5.95E-05	0.034	456.312	0.369	21.6	-5

The characteristic equation mapping joint current to estimated torque is the same for all of the joints.

$$\tau_{est, \dot{\theta}=0} = f(i - i_{friction}) = sgn(\dot{\theta}_{joint}) A i_{joint}^2 + B i_{joint} + C \quad (2.29)$$

As with joint 2, parameter C is an artifact of estimation errors and is not used.

These parameters will be used in the next chapter to verify the black box model. The estimated torque will be compared to the predicted torque while moving more than one joint. The estimated torque will also be compared to the torque due to a measured contact force. These two validation techniques are a basis for the collision detection technique discussed and demonstrated later.

3. VALIDATION OF TORQUE ESTIMATE

The significance of the joint torques to human robot interaction is of interest in 3 different spaces illustrated in Figure 3.1. In joint space there is a relationship between the actuator and the joint torque. There is also a relationship between the joint torques and the manipulator contact forces, i.e. the operational space. Those forces are important to evaluating the safety of human-robot interactions.

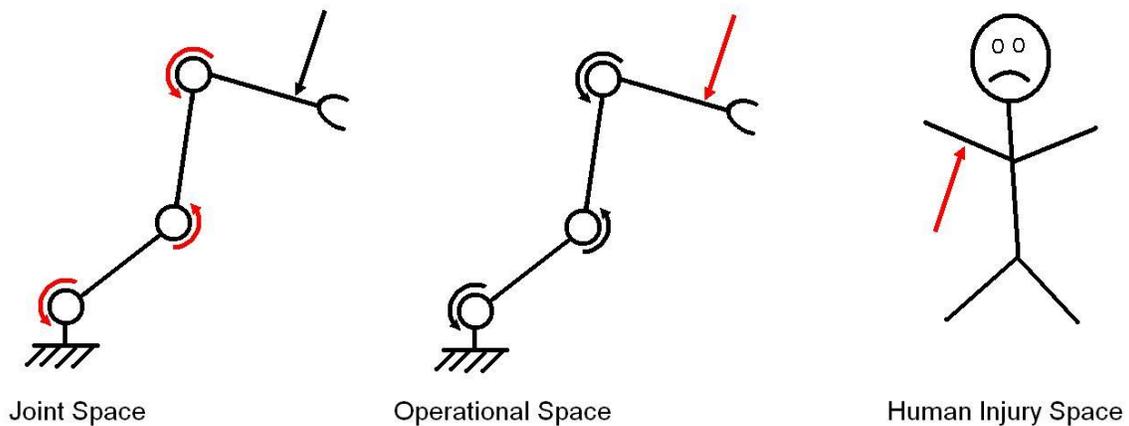


Figure 3.1: Estimated joint torque is significant to manipulator interaction and human safety

This chapter validates the black box estimated joint torque in the joint and operational spaces. Human injury is studied in later chapters. First, the manipulator is moved through several 6 DOF motions and the estimated torque is compared to the predicted torque (joint space). The second method compares the estimated torque to the torque which results from a measured contact force (operational space).

3.1. Estimated and Predicted Comparison Experiments

For the first method, the estimation error for joint 2 is calculated as the difference between predicted and estimated torque.

$$\tau_{error} = \tau_{pred} - \tau_{est} \quad (3.1)$$

The torque error is measured during various motions by moving the entire manipulator through a spatial trajectory. The torque estimation must be applicable for a variety of manipulator configurations for effective use in glovebox applications. It is particularly important that the model provides accurate torque estimates while the robot completes typical pick and place glovebox tasks and moves through the entirety of its workspace.

In the first motion, the end effector is moved linearly forward and down then returned to its starting position. (Figure 3.2)

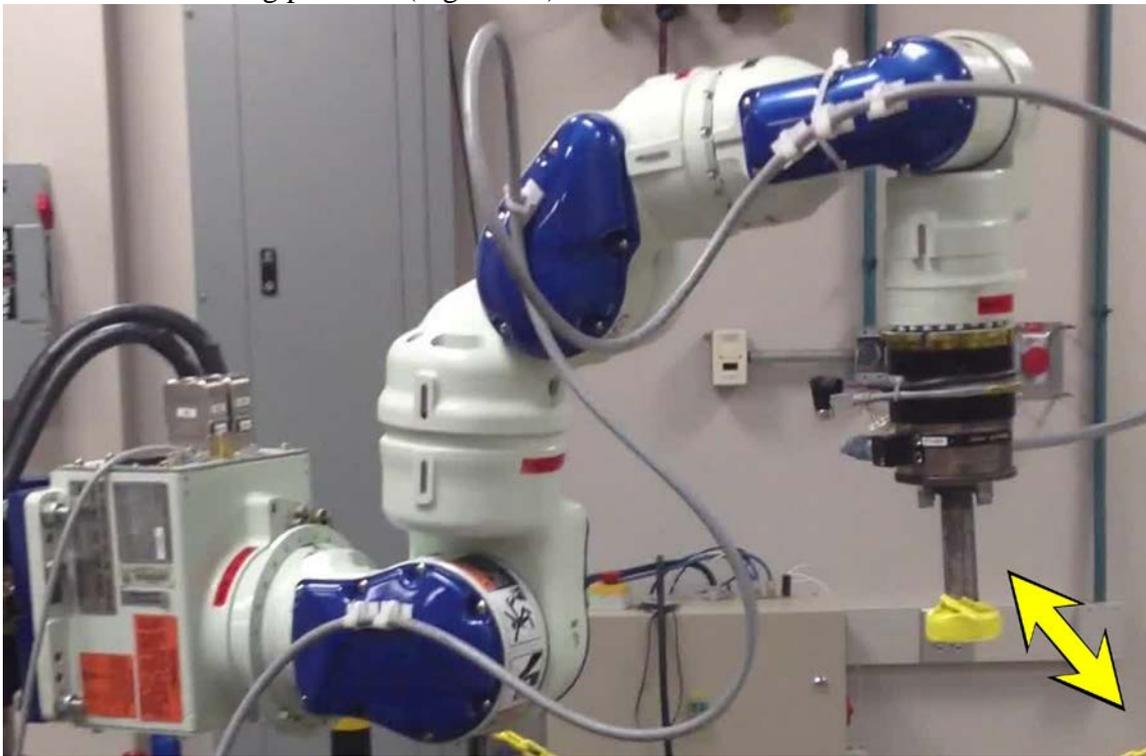


Figure 3.2: Pick and place example motion for validation

This is an important pick and place motion typical of many glovebox applications. The position of joints 2, 4, and 6 are shown in Figure 3.3.

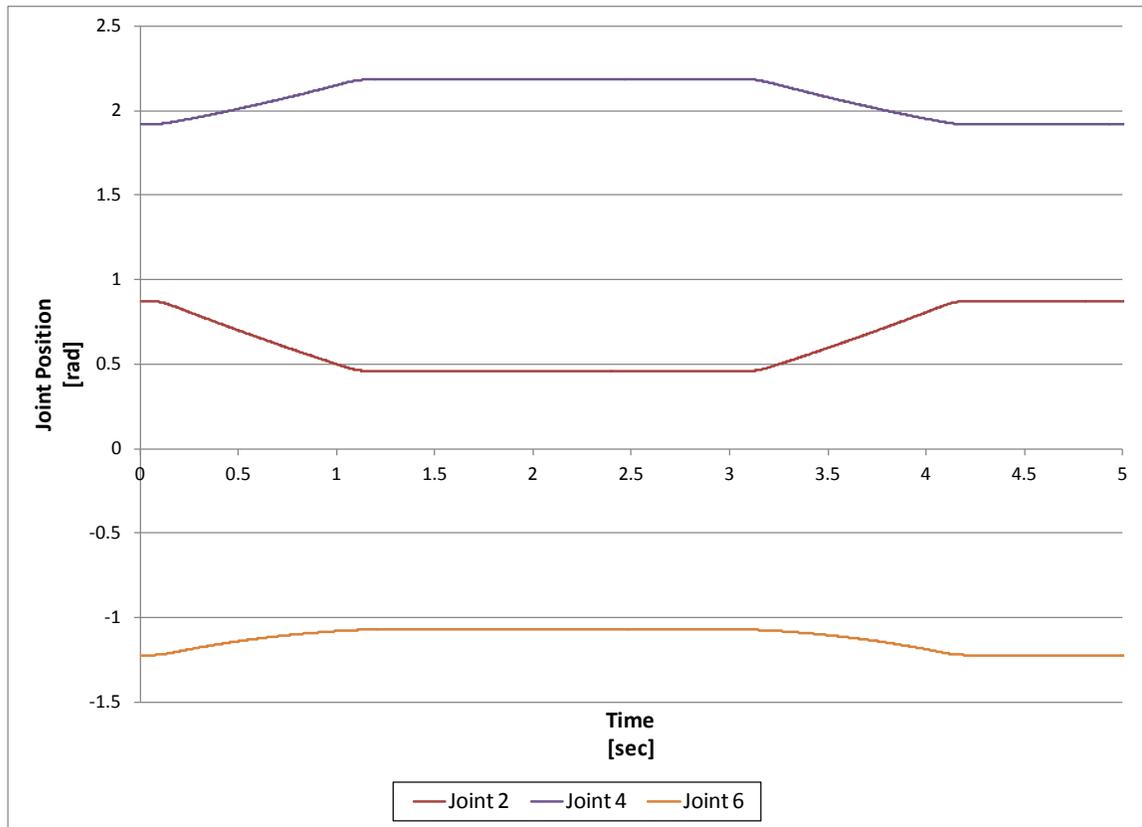


Figure 3.3: Example linear EEF validation motion positions

The torque estimation error during the motion is shown in Figure 3.4. The error remains low except when effected by joint acceleration. The increased estimation error due to acceleration can be noted at or just after the acceleration peaks (0.25, 1.25, 3.25, and 4.25 seconds). During the rest of the motion, the error remains below 5Nm.

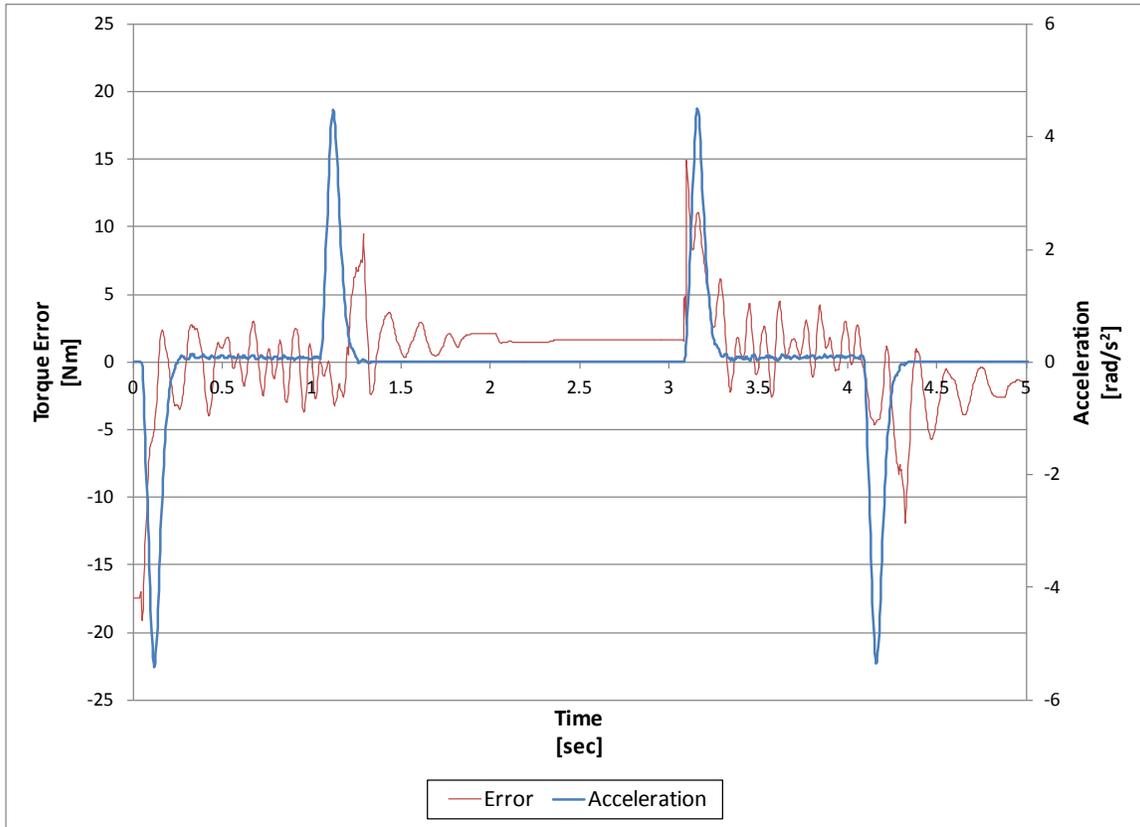


Figure 3.4: Example linear EEF validation motion error and acceleration

The next two motions move all of the joints at various joint velocities. The motions were designed to traverse the entire manipulator's work space and move all of the joints at a variety of velocities. The joint positions of the first motion are shown in Figure 3.5.

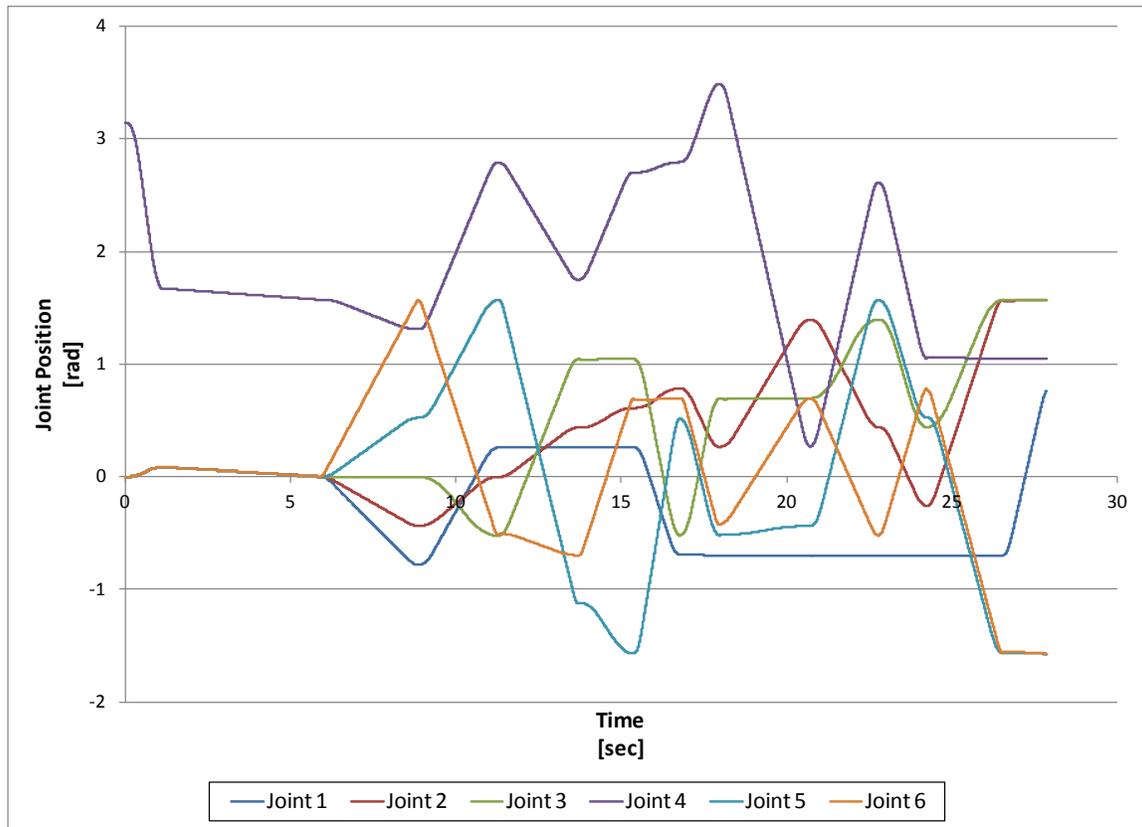


Figure 3.5: Joint positions for workspace test 1 joint move

During this motion, the estimated and predicted torques of joint 2 are nearly the same. There are a few spikes where the estimation error increases. Again, these spikes occur where the acceleration is highest. The estimation error during the rest of the motion is larger than in the first example. These data were taken on the Los Alamos manipulator while the model was tuned for the Austin manipulator. While the results on these Los Alamos data are good, they are not as accurate as for the Austin manipulator. The estimation errors of less than 20Nm are still quite good.

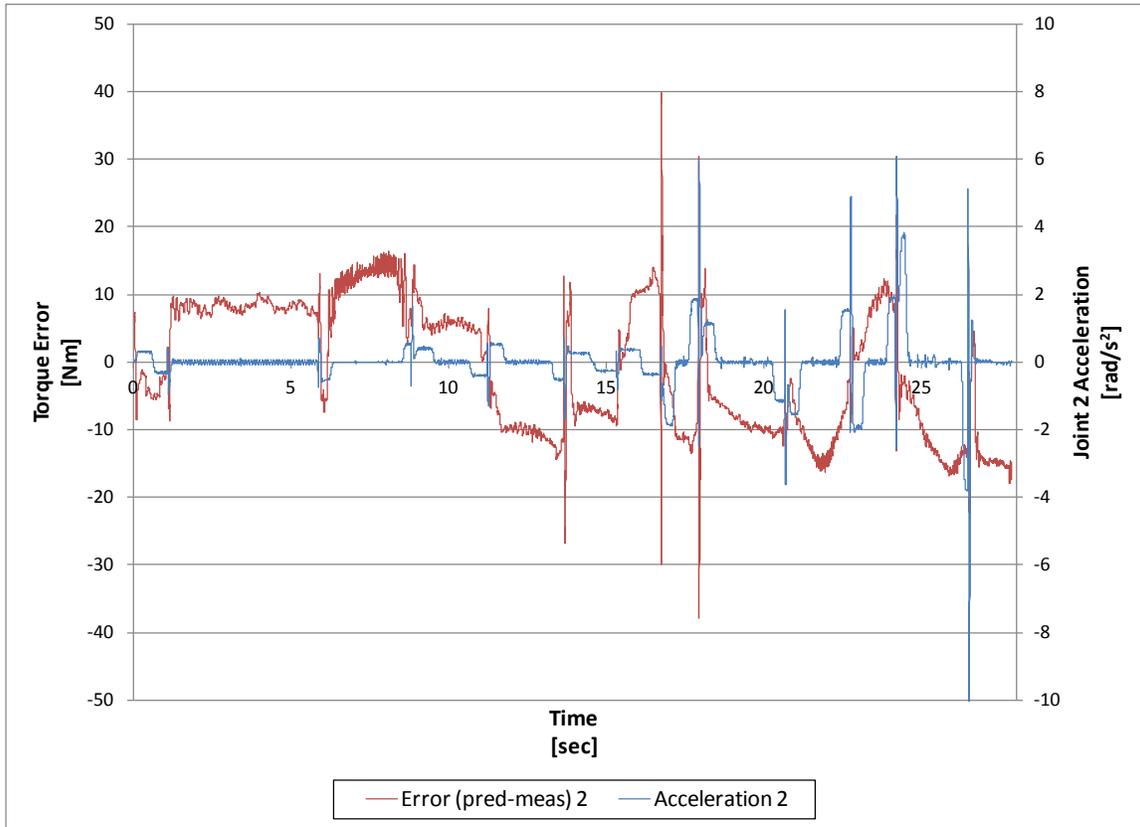


Figure 3.6: First workspace validation motion, joint 2 torque error and acceleration

The joint positions for the second workspace validation motion are shown in Figure 3.7.

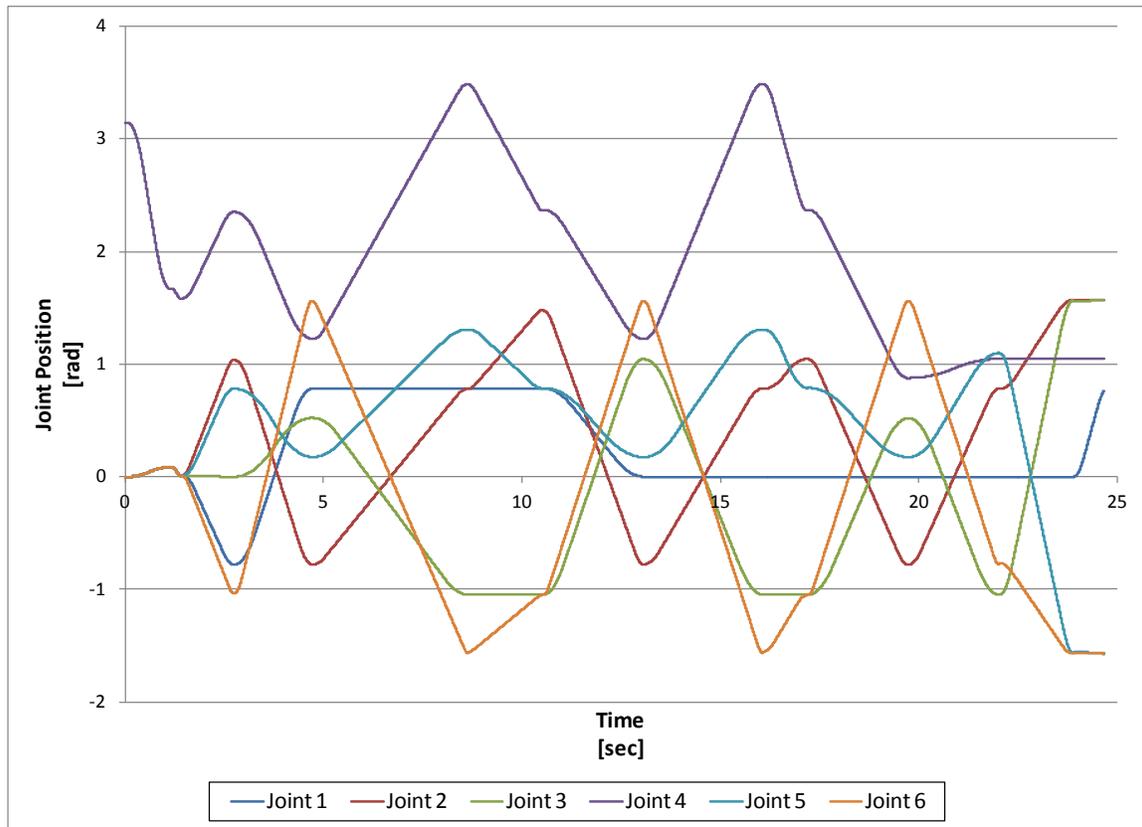


Figure 3.7: Second workspace validation test, joint positions

The estimation error magnitude (Figure 3.8) peaks around 25Nm for this motion. As in the last motion, the error is greater during high acceleration. This may be due to the absence of the acceleration-dependent inertia term, but may also be due to artificial spikes in the numerical estimation of the acceleration which propagate, via the predicted joint torque, to the estimation error.

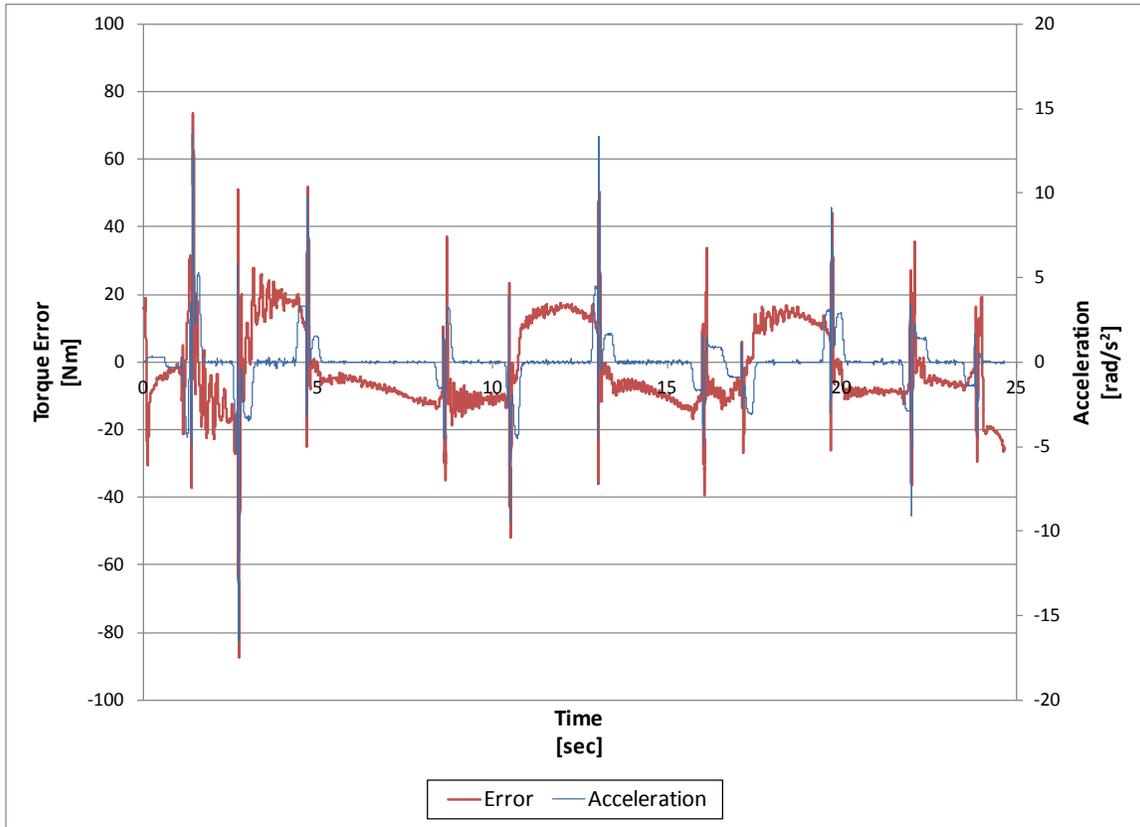


Figure 3.8: Second workspace validation motion, joint 2 torque error and acceleration

In the above tests, the estimated torque error was reasonably low for all the tests except when the acceleration spiked. The estimates are consistent even for aggressive (high velocity, all joints moving) 6 DOF motions. The estimation errors for each joint during the first all-joint motion are shown in Figure 3.9.

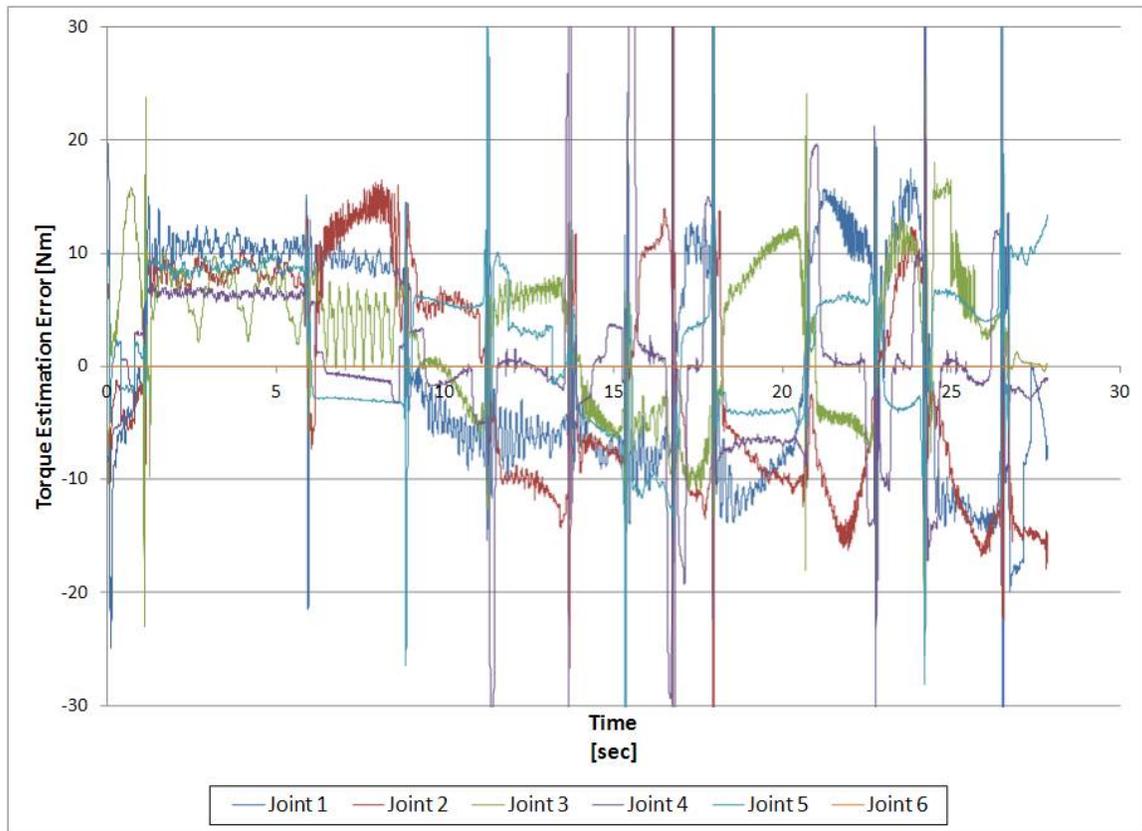


Figure 3.9: First workspace validation motion, all joints torque error

The average error and standard deviation for each of the joints for each of these tests is shown in Table 3.1.

Table 3.1: Validation Motion Statistics

	Pick and place		Workspace motion 1		Workspace motion 2	
	Average [Nm]	Std. Dev. [Nm]	Average [Nm]	Std. Dev. [Nm]	Average [Nm]	Std. Dev. [Nm]
Joint 1	4.4391	5.0078	0.6142	9.8146	0.0314	8.1604
Joint 2	0.2607	3.5832	-0.9028	9.6544	-1.9945	12.7070
Joint 3	-5.6133	4.1414	3.0950	6.1881	0.9921	9.0358
Joint 4	1.5958	1.5317	0.1803	8.9397	0.9219	6.3589
Joint 5	-5.6335	1.5256	2.0444	6.3575	-0.3515	5.9359
Joint 6	0.1112	2.8138	0.0000	0.0001	1.1532	6.1437

If the errors were all due to random system noise, the average errors would be expected to approach zero the longer the system operates. The configurations and velocities sampled are not exhaustive, which may lead to the non-zero mean errors. The average

error for each of the motions, even with the large acceleration-related spikes, is quite small, less than 10Nm for every joint in every test. The standard deviations are also less than 10Nm for all tests except joint 2 in the third test. One possible reason for increased errors on the workspace motions is the inconsistent feedback sample rate of the Los Alamos system. The two workspace tests, in contrast to nearly every other test in this work, were performed on the Los Alamos system. Variations in the feedback rate would lead to inaccuracies in the velocity and acceleration estimates.

The next section relates the estimated torque to the operational space, i.e. the accuracy of the contact torque.

3.2. Torque Due to Contact Force Validation

The second validation method evaluates the torque due to an external force, i.e. the contact torque. In these experiments, the force applied to the manipulator is measured by a six-axis force torque sensor. The predicted joint torque, $\tau_{contact,pred}$, is calculated by using equation (2.12).

$$\tau_{contact,pred} = J^T F_{meas} \quad (2.12)$$

Comparing the estimated torque due to contact and the torque due to the measured force lends operational-space significance to the estimation errors.

The joint torque due to contact forces is identified by subtracting the predicted torque (joint torque due to position, velocity, and acceleration) from the black box estimated joint torque.

$$\tau_{contact} = \tau_{est} - \tau_{pred} \quad (3.2)$$

The estimated contact torque is compared to the predicted contact torque due to a measured force. The force which would generate the estimated contact torque is compared to the measured force. The difference is the force discrepancy. Both the

contact torque and force validations are important to collision detection and human safety, covered in later chapters.

The torque error in the joints of the manipulator can be noted in typical torque units like Newton-meters. However, the torque induced by a force at the end effector changes with the configuration of the manipulator so the significance of the error in operational and human safety spaces is not immediately evident. For some insight, the error in the torque due to contact force will be related back to the equivalent contact force.

The estimated torque, $\tau_{contact,est}$, is predicted by the black box model. The force that would yield the estimated torque is the estimated force, F_{est} . Equation (2.12) for the estimated torque is divided by the equation for predicted torque, $\tau_{contact,pred}$, due to the measured force, F_{meas} .

$$\frac{\tau_{contact,est}}{\tau_{contact,pred}} = \frac{J^T F_{est}}{J^T F_{meas}} \quad (3.3)$$

For the purpose of evaluating the error in the operational space, it is assumed the direction of the estimated force is the same as the measured force.

$$\frac{\tau_{contact,est}}{\tau_{contact,pred}} = \frac{J^T \hat{F}_{dir} |F_{est}|}{J^T \hat{F}_{dir} |F_{meas}|} \quad (3.4)$$

So the vector portions of top and bottom are the same. The unit torque vector can be found by factoring out the magnitude for the torque.

$$\frac{|\tau_{est}| \hat{\tau}_{unitforce}}{|\tau_{pred}| \hat{\tau}_{unitforce}} = \frac{J^T \hat{F}_{dir} |F_{est}|}{J^T \hat{F}_{dir} |F_{meas}|} \quad (3.5)$$

Simplifying and rearranging yields the magnitude of the estimated force in terms of the measured force, measured torque, and estimated torque.

$$|F_{est}| = |F_{meas}| \frac{|\tau_{est}|}{|\tau_{pred}|} \quad (3.6)$$

The error force is the difference between the estimated force and measured force.

$$F_{err} = |F_{est}| - |F_{meas}| \quad (3.7)$$

This equation can be rewritten in terms of the measured and estimated torques.

$$F_{err} = \left(\frac{|\tau_{est}|}{|\tau_{pred}|} - 1 \right) |F_{meas}| \quad (3.8)$$

The above can be used to evaluate the significance of the torque error in terms of the contact force. This evaluation relies on the direction of the measured force. When the force is very small, the force error becomes unusable.

The contact force validation test was performed by pressing the end effector in a linear motion against three different objects: a block of open-cell foam, a piece of Plexiglas, and a 5 inch by 5 inch piece of fiberboard on the foam. (Figure 3.10)

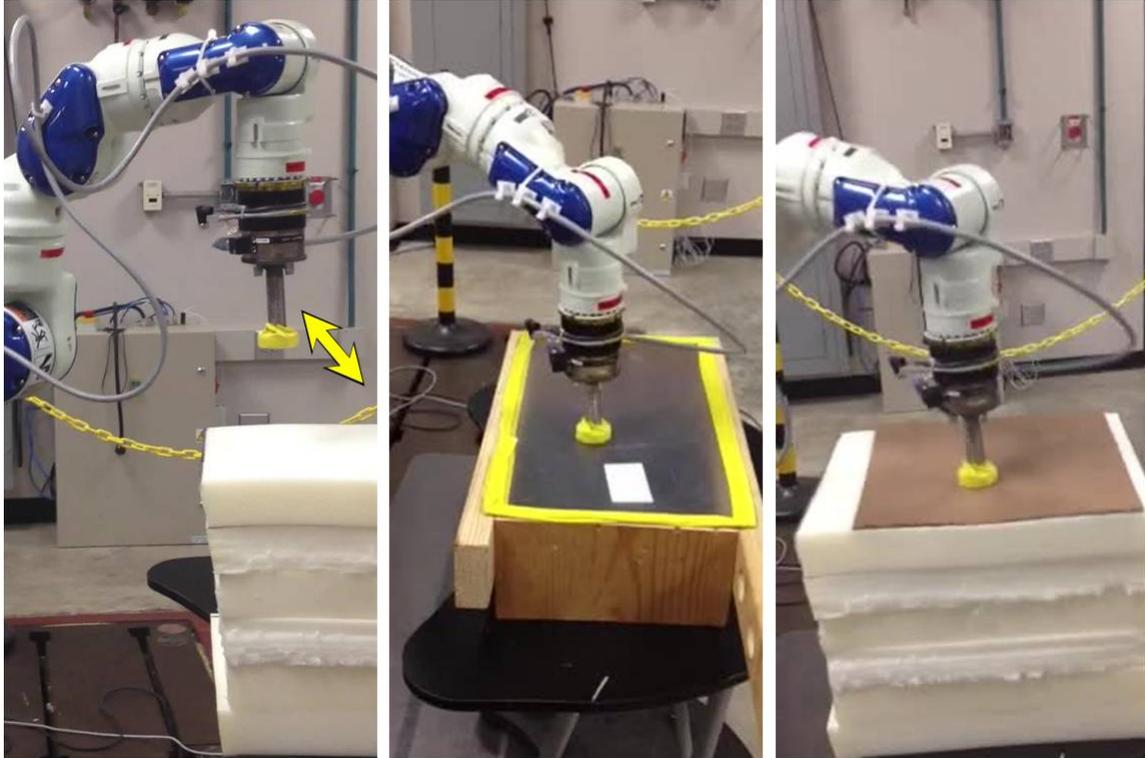


Figure 3.10: Contact test setup with foam (left), Plexiglas (center), and fiberboard on foam (right)

The different objects offer different effective stiffnesses. The velocity of the end effector was also varied during these tests. The force during contact was measured by a six-axis force/torque sensor mounted at the end effector.

The force magnitude, the estimated and measured contact torque, and the force discrepancy for 25mm/s EEf velocity on foam are shown in Figure 3.11. (Contact is not made until roughly 1.7 seconds after the motion starts.) During the initial contact, while the force is increasing, the force error is less than 5N (1.12lb).

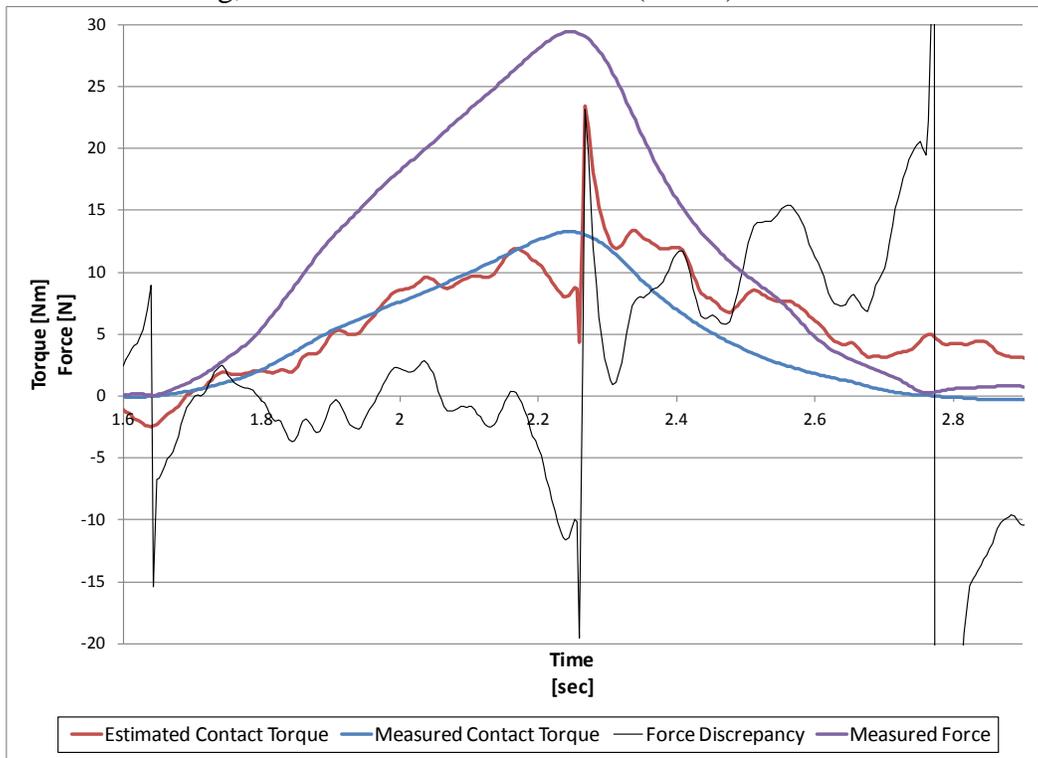


Figure 3.11: EEf pressing on foam at 25mm/s

The error increases during acceleration. (Figure 3.12) The spike during contact is associated with the hysteresis as the joint velocity passes through zero and reverses directions. The spikes on either side of the contact, at about 1.63s and 2.78s, are because of the sensitivity of equation (3.8) to small measured contact torques.

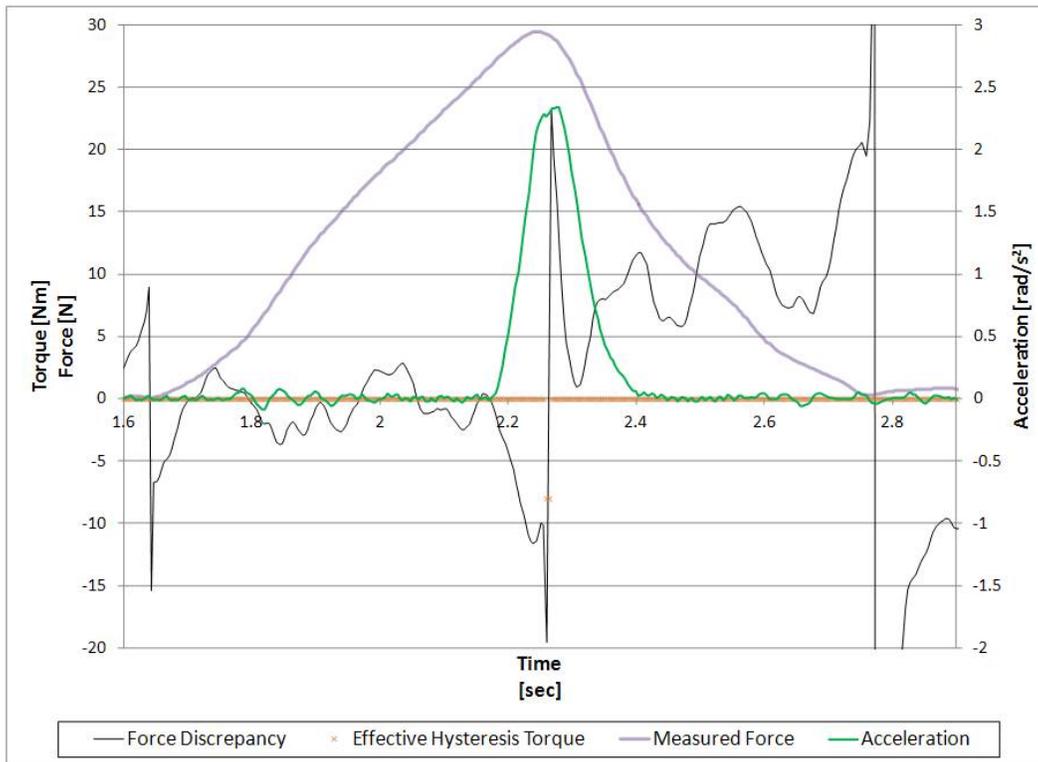


Figure 3.12: Joint 2 force error, hysteresis, and acceleration for 25mm/s EEF velocity on foam

Figure 3.13 shows contact torques and forces when the EEF velocity is 50mm/s against the foam. Again, the force error is less than 5N before the joint accelerates. For the 75mm/s test (Figure 3.14), the approach force error was slightly larger, but still less than 10N (2.24lb).

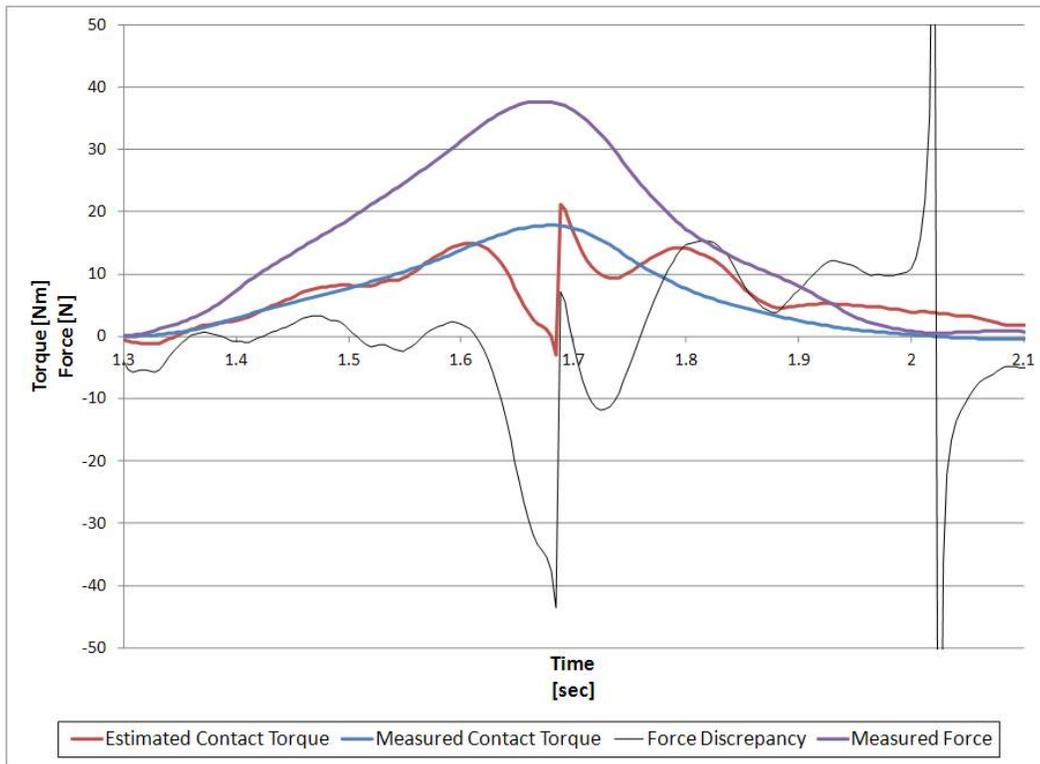


Figure 3.13: EEF pressing on foam at 50mm/s

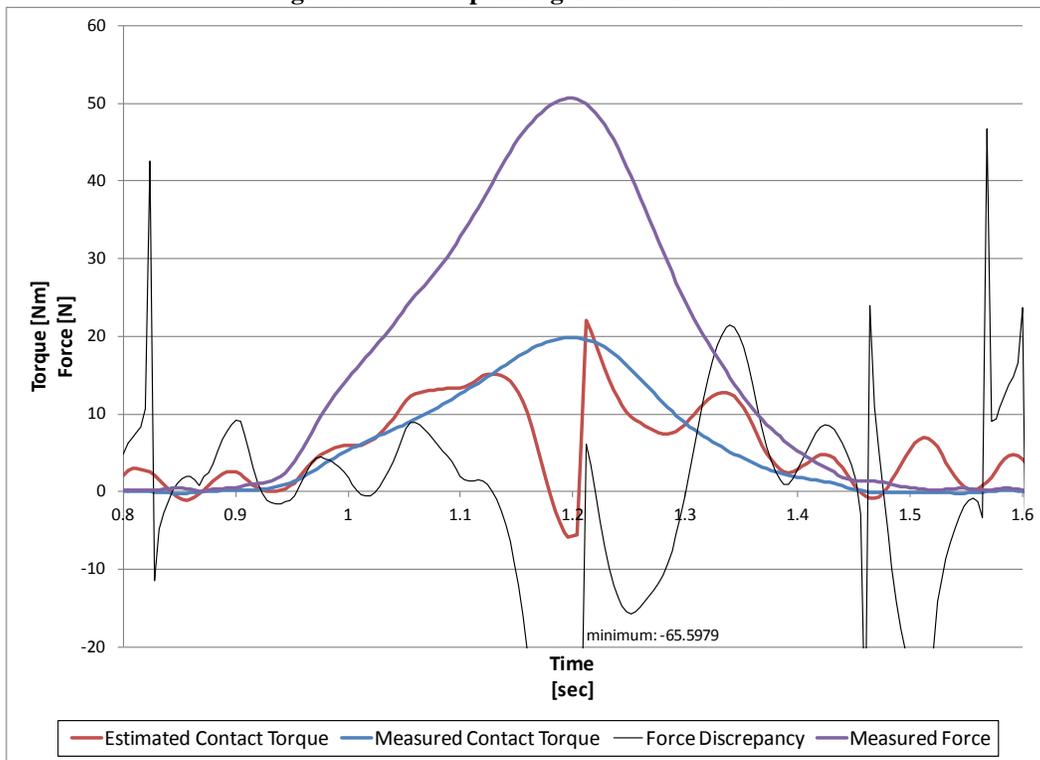


Figure 3.14: EEF pressing on foam at 75mm/s

When pressing at 25mm/s against Plexiglas, the results are similar. Force discrepancy is less than 10N until influenced by the acceleration and hysteresis. The maximum force is higher, as expected from a stiffer object.

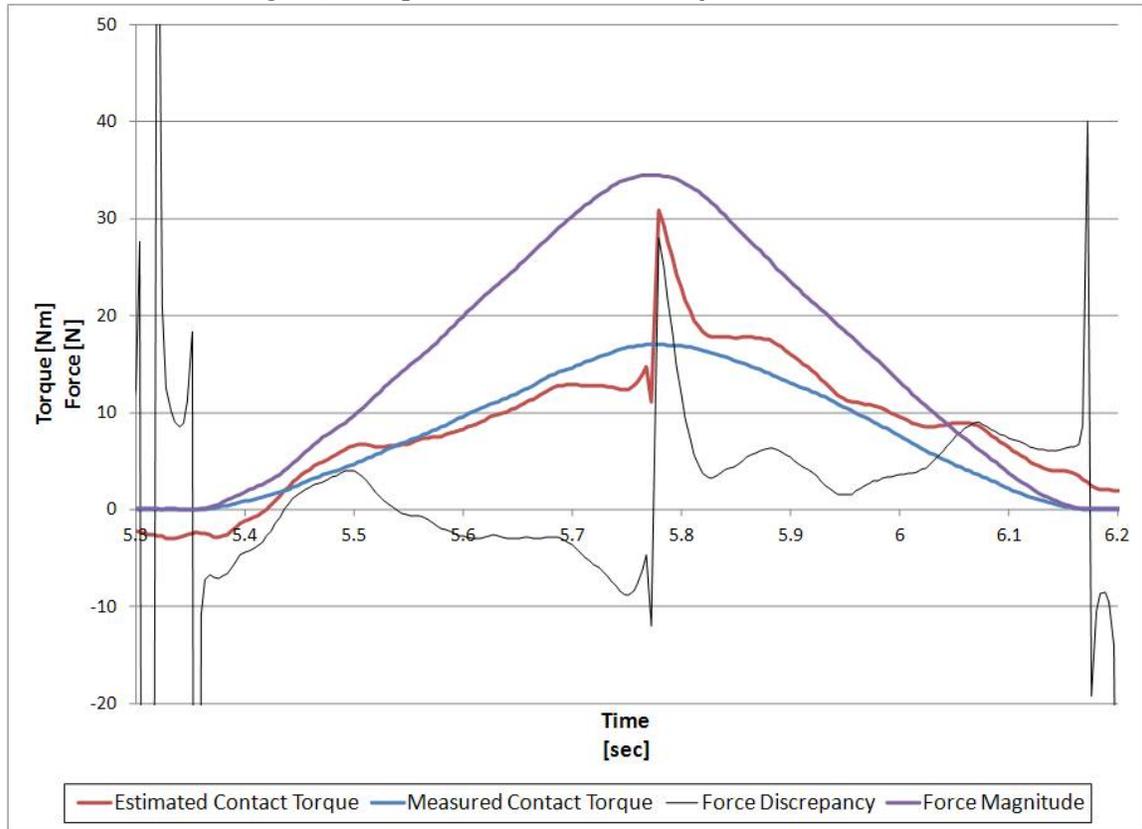


Figure 3.15: EEF pressing on Plexiglas at 25mm/s

Next the EEF was pressed against Plexiglas at 125mm/s. The measured force and contact torques are shown in Figure 3.16. The EEF velocity was higher and Plexiglas is stiffer than the foam so the maximum force is more than twice that of the 75mm/s test on foam (110N, 24.7lb). However, the force error remains less than 10N except during acceleration.

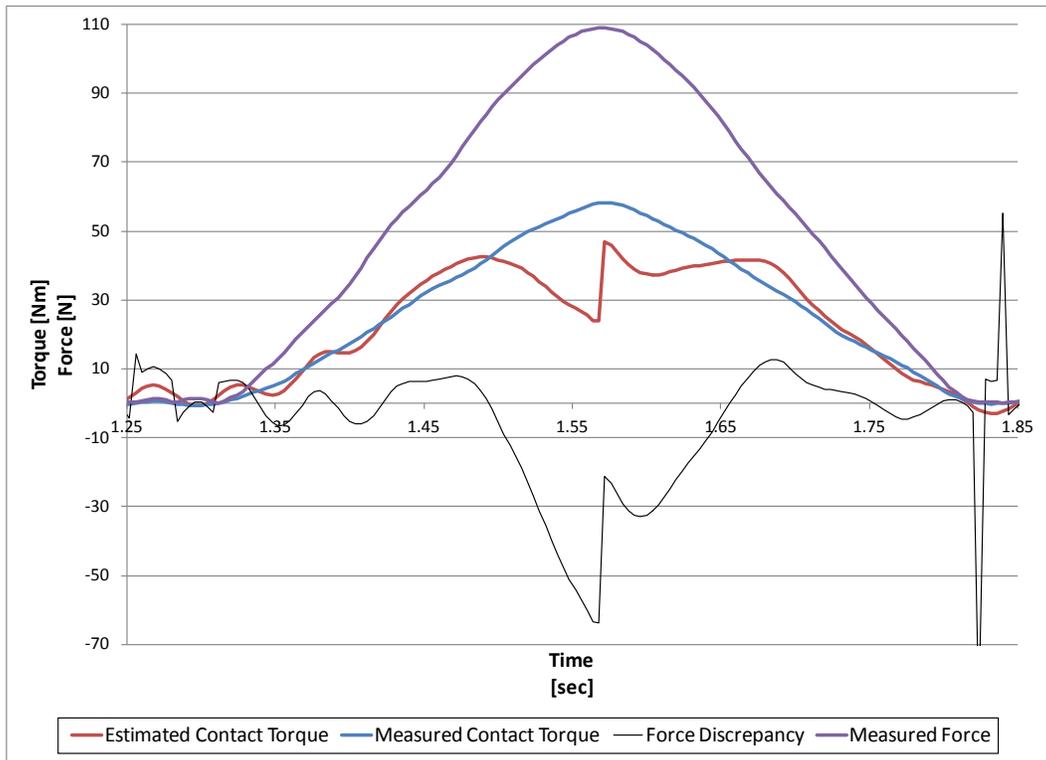


Figure 3.16: EEF pressing on Plexiglas at 125mm/s

A piece of fiberboard 5 inches by 5 inches was placed on the foam to change the effective stiffness. At 25mm/s EEF velocity, the maximum force was 55N (12.4lb). Maximum error was less than 5N except during acceleration.

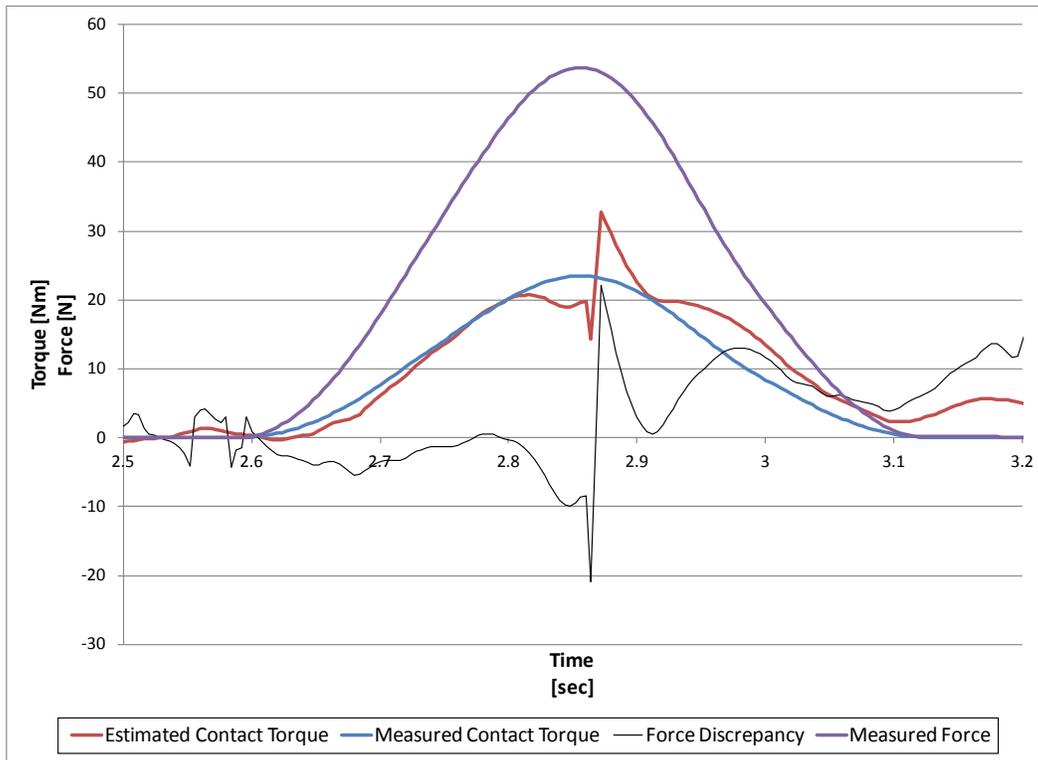


Figure 3.17: EEF pressing on 5x5 fiberboard on foam at 25mm/s

The estimated contact torque followed the measured contact torque well for all tests except during acceleration. The increased estimation error during acceleration was noted in the free motion validations of Section 3.1 Estimated and Predicted Comparison Experiments. The estimated torques have potential applications in a variety of torque feedback algorithms. In this work, estimated torques will be used to demonstrate collision detection on the SIA5D and to evaluate the effectiveness of the collision detection with humans.

4. COLLISION DETECTION

In this chapter, the estimated joint torque will be used to detect collisions. A broad review of collision detection was provided in the introduction. The review here will focus only on point-of-actuation techniques. The method used in this work is presented. Collision detection using black box estimated joint torques is demonstrated at the end of this chapter.

4.1. Point of Actuation Collision Detection Review

Promising work has been done in point-of-actuation collision detection. Several different methods have been demonstrated using actuator position, velocity, torque, and motor current feedback. Many methods compare predicted and measured values and detect a collision as an unacceptable difference in these quantities. Several significant and relevant works are reviewed here.

Je et al. [2009] use an actuator current disturbance observer method to detect collisions. The method measures the input current to each joint. A collision is assumed to occur if the input current changes abruptly, thus it does not require an accurate dynamic model or a current-to-torque mapping. Je's method does require that the manipulator's motion not cause a rapid change in current (due to high accelerations, abrupt changes in direction, etc.) which may lead to false positives. Additionally, contacts resulting in slowly building forces may go undetected as the current does not change quickly. Collisions with soft, compliant objects, such as a human abdomen, would have slowly changing interaction force (and joint torque) profiles and prove harmful if the force or displacement grows too large. In Je, the observer also used the motor voltage – data not available in this work. Additionally the use of torque

thresholds, as presented later, instead of a current observer, offers a means to study collision detection effectiveness.

De Luca et al. [2006] use the total manipulator energy and generalized momentum to determine when a collision has occurred. The expected energy and momentum are calculated based on desired trajectories, i.e. joint position and velocity, of the manipulator. A disturbance observer detects when a collision causes the momentum or energy to differ from the expected. This method is very similar to the joint torque disturbance observers discussed below but does not require calculation of joint accelerations. Numerically estimating the velocity and acceleration introduces noise to the torque estimate, thereby decreasing sensitivity. De Luca's method cannot be used here because the position-controlled manipulator used in this work closes the position loop before a disturbance in the position feedback is noticeable.

Takakura et al. [1989] and Ralph and Pai [1995] use manipulator models to predict joint torques for a freely-moving manipulator. The predicted torques are compared to the measured torques. If the manipulator experiences unexpected contact, the joint torques change. The difference in predicted and measured torque is called the disturbance. Collisions are identified when the magnitude of the disturbance is greater than a threshold value (Figure 4.1, discussed in greater detail in Section 4.2). This method requires an accurate model of the manipulator in order to predict the joint torques. This method most closely represents the method used in this work. However, instead of measuring the joint torque, as Takakura, et al. and Ralph and Pai do, the joint torque is estimated from the black box model presented and demonstrated in previous chapters. Human-robot collision detection effectiveness will be addressed in more detail in a later chapter.

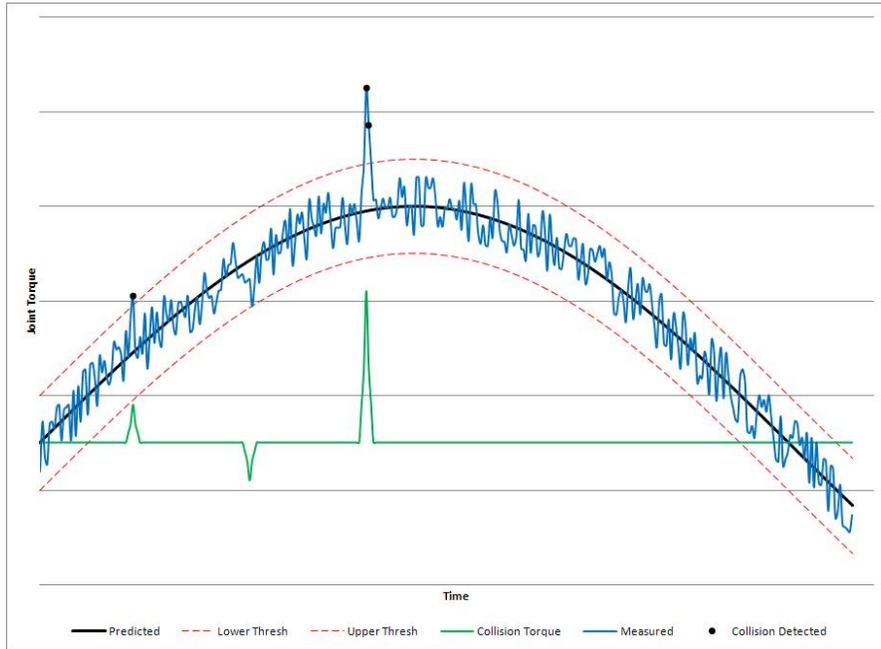


Figure 4.1: Illustrative collision detection with collision torque and sensor noise

4.2. Demonstrated Method

In the previous two chapters, a method for estimating joint torque was presented and validated. In this chapter, the estimated torque is used to detect collisions. In the last chapter, the difference between estimated and predicted torque was treated as estimation error when it was known no contact existed. The same difference was identified as contact torque in the presence of a known external force. In this chapter, either state is possible. The goal is to correctly identify when the manipulator has made contact with the environment.

A threshold is chosen for the accepted uncertainty in estimation error. When the error is less than the threshold, the error is attributed to model uncertainties and noise. When the error exceeds the threshold, it is treated as contact torque and a collision is detected.

$$if(|\tau_{error}| > \tau_{threshold}), then collision \quad (4.1)$$

Figure 4.1 is representative of what the estimated and predicted torques may look like during a motion with collision detection. The green line indicates the true contact torque (unknown without additional sensing) and is shown to illustrate that real collisions below a certain threshold may not be detected (the negative contact torque in the figure). When the estimated torque crosses the red dashed thresholds, a collision is detected.

When a collision is detected, a signal is sent to the controller to abort the current motion. (Figure 4.2) This is used in lieu of the emergency stop to prevent erratic motions and avoid pinning humans in a dangerous position while the robot is restarted and re-enabled. [Steinfeld 2009] A reaction motion can be sent to the controller or the robot can wait for an operator to command an appropriate response.

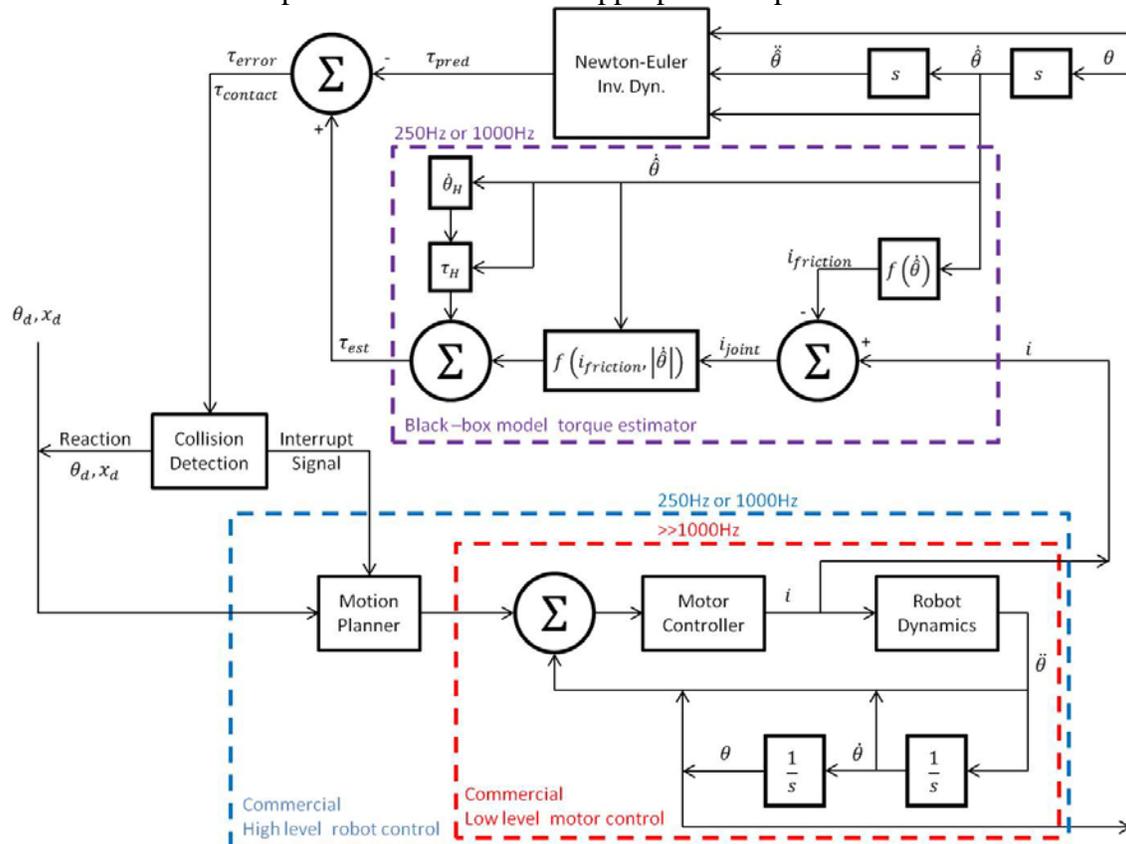


Figure 4.2: Commercial system with black box torque estimation and collision detection

Of note in Figure 4.1 is that not every contact is detected. The second contact goes undetected even though it has the same torque magnitude as the first. The model sensitivity and the chosen thresholds influence the collision detection performance in joint space.

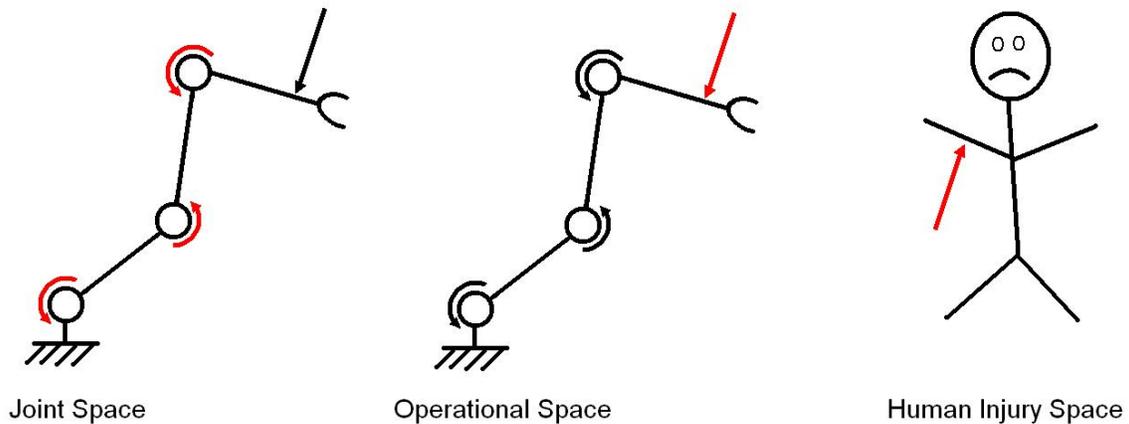


Figure 4.3: Illustration relating Joint, Operational, and Human Injury spaces

There are other factors, such as manipulator configuration, which influence the operation space. Collision detection factors in joint and operational space will be discussed in this chapter. (Figure 4.3) Human injury will be discussed in a later chapter.

4.3. Considerations in Joint Space

In order to detect a collision, the estimation error must exceed the accepted thresholds. If the estimation error was zero, a collision would be detected as soon as the contact torque exceeded a value equal to the threshold. However, as shown in the previous chapter and illustrated in Figure 4.1, the error is rarely exactly zero. When it is not zero, the distance to each threshold changes; one becomes nearer and the other further. Consider the illustrative data in Table 4.1.

Table 4.1: Example collision torques required to detect collision

Error Without Contact	Threshold	Required Positive Torque	Required Negative Torque
0	10	>10	<-10
5	10	>5	<-15
10	10	>0	<-20
-7	10	>17	<-3
-10	10	>20	<0

When the error is positive, a contact which generates a positive torque will not need to be as large as one which generates a negative torque. Assuming the estimation error can have any value between but not exceeding the thresholds, the worst-case contact torque would be equal to the distance between thresholds, as illustrated in Figure 4.4.

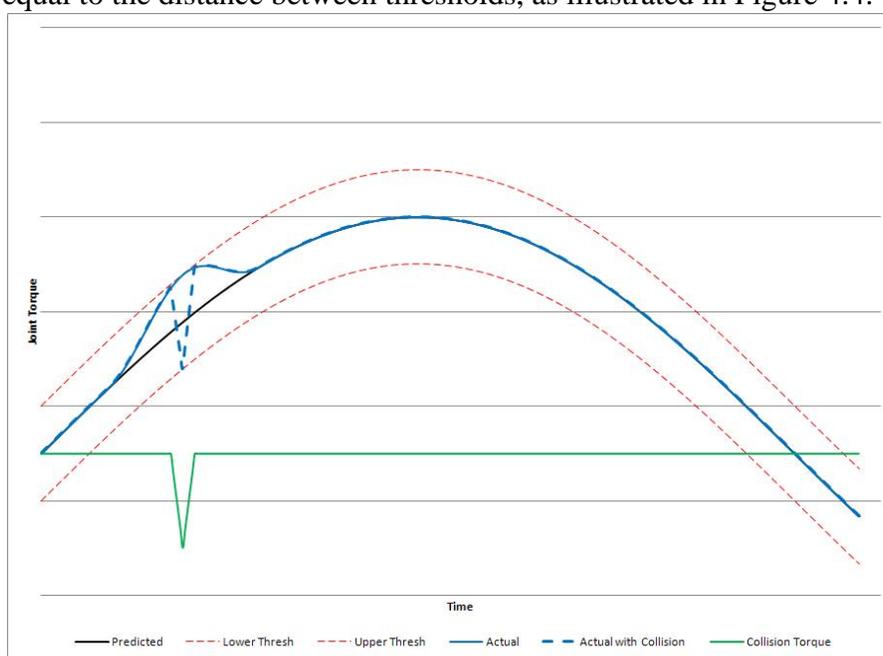


Figure 4.4: Illustration of worst case collision torque for detection

In the example illustrated in Figure 4.4, if the contact torque had been positive instead of negative, it would only need to be greater than zero. The minimum detected contact torque is then $+0$ while, in the opposite direction, it was $-2\tau_{threshold}$. The deviation in the illustration could be inverted, making the minimums -0 and $2\tau_{threshold}$.

Instead of addressing the minimum detected torque, this work will reference the maximum torque which may go undetected⁴.

Feedback frequency also affects the characteristics of the collision response. Between samples, the change in torque, $\Delta\tau$, is determined by the rate of torque change, $\frac{\Delta\tau}{\Delta t}$, and the sample time, Δt .

$$\Delta\tau = \frac{\Delta\tau}{\Delta t} \Delta t \quad (4.2)$$

Combined with the effect of the thresholds, the maximum torque which may go undetected is

$$|\tau_{max,undetected}| = 2\tau_{threshold} + \left| \frac{\Delta\tau}{\Delta t} \right| \Delta t. \quad (4.3)$$

The rate of torque change is related to the end effector speed, the manipulator configuration, and the stiffness of the contact object.

4.4. Considerations in Operational Space

The relationship between the contact force and the joint torque was presented in previous chapters. The torque due to contact is equal to the transpose of the Jacobian times the contact force.

$$\tau_{contact} = J^T F_{contact} \quad (2.12)$$

The equation cannot be solved directly for the contact force. However, given the maximum *torque* which may go undetected (Equation (4.3)), direction of the force, and contact Jacobian, it can be used, via the force estimation techniques in Chapter 3, to estimate the corresponding *force*. The subscripts of equation (3.6) are adapted to solve for the detection force given a force in a particular direction.

$$|F_{max,undetected}| = |F_{sample}| \frac{|\tau_{max,undetected}|}{|\tau_{sample}|} \quad (4.4)$$

⁴ The maximum torque which may go undetected assumes the estimation error, without contact, may have any value between the thresholds but not exceeding them. $-\tau_{threshold} \leq \tau_{error} \leq \tau_{threshold}$.

The sample force, F_{sample} , is any force in a direction of interest. The torque due to that sample force, τ_{sample} , is calculated using equation (2.12).

4.4.1. NULL SPACE AND CONDITION NUMBER

The equation for contact force (eq. (2.12)) is a linear system of the form

$$J^T F_{contact} = \tau_{contact}; Ax = \mathbf{b}. \quad (4.5)$$

The determinant of the matrix J^T can be calculated from the product of its eigenvalues. When the determinant of J^T is zero, the system is in a singular configuration. Given a singular configuration, there exists a null space which contains the set of non-zero $F_{contact}$ for which the solution, $\tau_{contact}$, is zero⁵. For the robot contact problem, the null space is the set of forces which yields zero torque in the joints. If $F_{contact}$ is in the null space of J^T , then so is $kF_{contact}$, i.e. if a force is in the null space, so are all other magnitudes of that force in that direction. These null space forces are absolutely undetectable as they generate zero torque in the joints.

Regardless of the size of the null space, there will be forces which *can* be detected. These are the forces in the row space of J^T . In the best case, a small input $F_{contact}$ yields a large $\tau_{contact}$. In these cases, $F_{contact}$ will be easily detected. But it is possible an input $F_{contact}$ will yield a small $\tau_{contact}$ ⁶ such that $F_{contact}$ must be very large to yield a detectable output. It is not guaranteed forces in the row space will be easily detected, only that they generate a non-zero torque in the joints.

So regarding detectibility, there are 3 types of forces.

- Undetectable forces – the system (J^T) is in a singular configuration. The force, no matter the magnitude, will not generate any torque in the joints.

⁵ The null space also includes the zero vector, but when the null space only considers the zero vector it is generally not referred to as a null space.

⁶ $\tau_{contact}$ doesn't need to be large, only non-zero, to be in the row space instead of the null space.

Reaching one of these singularities is unlikely for a position controlled system⁷. In numerically computed systems such as this one, this is unlikely ever to occur because floating point precision will generally lead to non-zero values even when they ‘should’ be zero.

- Difficult to detect forces – the force is in the row space, and thus detectable, but a large input force is required to generate a torque greater than the detection threshold.
- Easily detected forces – the force is in the row space and a detectable torque is generated with a force of a reasonable magnitude.

The magnitudes of force and torque which represent “large” and “reasonable” will be dependent on the system and the objectives. For work with sensitive objects, a “reasonable” force will be less than for work with robust objects.

The condition number is used to estimate when the configuration is near a singularity. The condition number is the ratio of the maximum eigenvalue to the minimum eigenvalue. However, when the matrix is not square, eigenvalues can’t be calculated so the matrix’s singular values are used instead. A matrix’s singular values are found by Singular Value Decomposition (SVD).

The SVD of an $m \times n$ matrix, A yields three matrices.

$$A = U\Sigma V^T \quad (4.6)$$

The matrices U and V^T are $m \times m$ and $n \times n$ matrices. The columns of U span the column space (u_1 to u_r) of A and null space (u_{r+1} to u_m) of A^T .

$$U = \begin{bmatrix} \vdots & & \vdots & \vdots & & \vdots \\ u_1 & \cdots & u_r & u_{r+1} & \cdots & u_m \\ \vdots & & \vdots & \vdots & & \vdots \end{bmatrix} \quad (4.7)$$

The rows of V^T span the row space (v_1^T to v_r^T) and null space (v_{r+1}^T to v_n^T) of A .

⁷ It is generally desirable to avoid singularities in J for kinematic reasons. The determinant of J^T is the same as the determinant of J , so if one is zero (singular) then so is the other.

$$V^T = \begin{bmatrix} \cdots & v_1^T & \cdots \\ & \vdots & \\ \cdots & v_r^T & \cdots \\ \cdots & v_{r+1}^T & \cdots \\ & \vdots & \\ \cdots & v_n^T & \cdots \end{bmatrix} \quad (4.8)$$

The singular values are along the diagonal of the $m \times n$ matrix Σ . In the standard notation, the matrices are decomposed such that the values of sigma are in descending order where the greatest singular value is the first element of the matrix, σ_1 , and σ_r is the smallest non-zero value.

$$\Sigma = \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & 0 & & \\ & 0 & & & 0 & \\ & & & & & \ddots \\ & & & & & & 0 \end{bmatrix} \quad (4.9)$$

If the matrix A is not singular, $r = n$ and there will be no zero singular values. When computing numerically, as in most control algorithms and for all purposes in this work, the singular values will usually all be non-zero. The condition number is used to estimate when the matrix is near or at a singularity (obscured by rounding/floating point errors).

The condition number is the ratio of the maximum singular value to the minimum. Since the singular values are ordered along the diagonal, the condition number, C , is

$$C = \frac{\sigma_1}{\sigma_r}. \quad (4.10)$$

Due to numerical computation, when the matrix is at a singular value, the diagonal is likely to contain very, very small, but non-zero numbers in the lower corner, driving the condition number toward infinity.

The condition number identifies when the system is at or near a singularity but it cannot identify which forces will be undetectable. To see if a force will be undetectable, the SVD of J^T can be substituted into equation (4.5).

$$\tau_{contact} = U \Sigma V^T F_{contact} \quad (4.11)$$

Multiplying the input first by the V^T matrix yields a *transformed input*, y

$$y = V^T F_{contact} \quad (4.12)$$

If the transformed input is only in the rows $r + 1$ to n , it is in the null space of the system, i.e. it can be described by the vectors which span the null space.

$$y_{null} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y_{r+1} \\ \vdots \\ y_n \end{bmatrix} \quad (4.13)$$

When this input is substituted into equation (4.11),

$$\tau_{contact} = U \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & 0 & & \\ & 0 & & & \ddots & \\ & & & & & 0 \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y_{r+1} \\ \vdots \\ y_n \end{bmatrix} \quad (4.14)$$

the zeros of the singular value matrix will result in zeros in the output.

4.4.2. SYSTEM NEAR SINGULARITY, FORCE IN ‘NEAR-NULL’ SPACE

If the system is only numerically approaching the singularity, the least singular values, $(\sigma_{r+1} \dots \sigma_n)$, will not be zero, but will be small.

$$\tau_{contact} = U \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & \sigma_{r+1} & & \\ & 0 & & & \ddots & \\ & & & & & \sigma_n \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y_{r+1} \\ \vdots \\ y_n \end{bmatrix} \quad (4.15)$$

The forces are no longer undetectable because the product of Σ and V^T will be non-zero.

But if the values of sigma are small, it might be expected the

The product of Σ and y is the *detectable transformed input*, z .

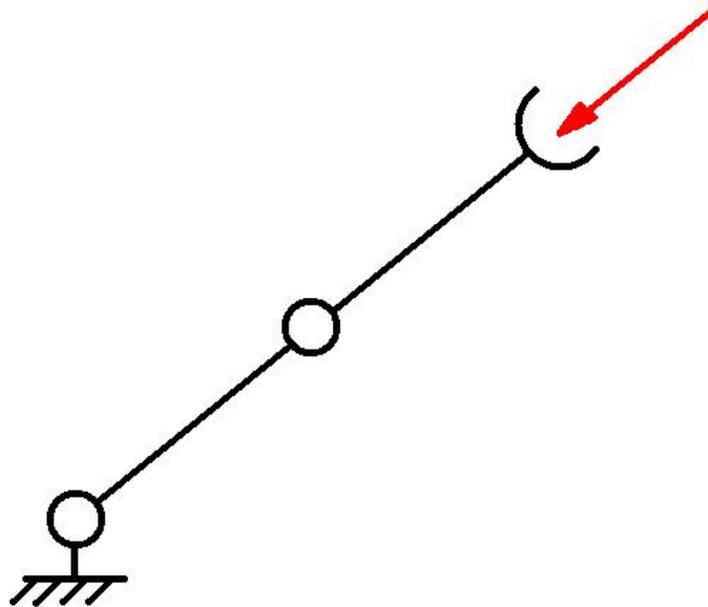


Figure 4.5: 2 DOF planar robot at a singularity

This is important to consider when planning paths and motions while using collision detection⁸. Singular configurations will limit or eliminate the ability to detect certain forces. It is also worth noting that, in the above – very academic – scenario, the robot is also incapable of contributing force from its own actuators at the point of collision.

Once the collision has been detected, the manipulator must stop or take actions to alleviate the collision force. Even if control actions happen almost immediately, until such actions are taken, the contact force magnitude continues to increase until the manipulator is completely stopped. This is illustrated in Figure 4.6. In some cases, the motion planner response rate is different than the feedback and detection rates. This allows the force to increase even more before the response begins. The motion planner response is the limiting factor. Even though detection may occur sooner, only the response feedback rate will be examined.

⁸ It is already common practice, for other reasons, to avoid singularities in robot motion plans.

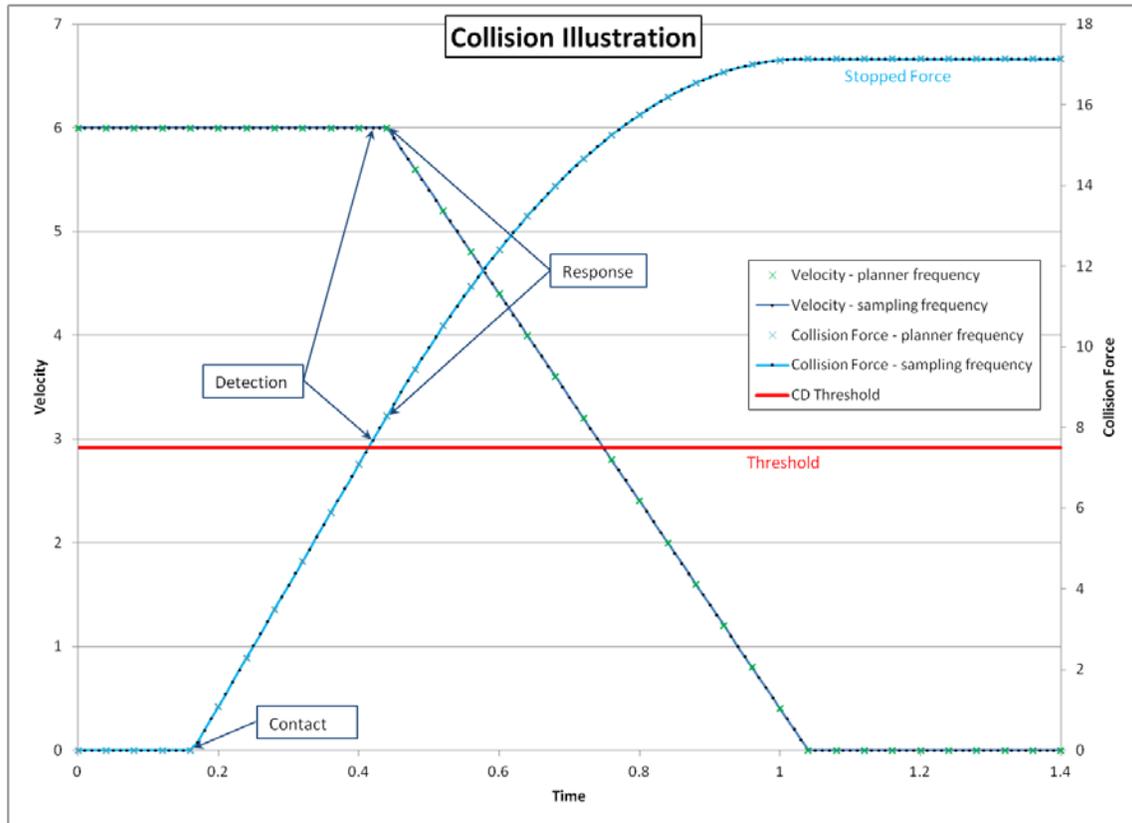


Figure 4.6: Illustration of effects of bandwidth on collision detection response

A non-zero stopping distance is required after detection and response. The collision force continues to build as the manipulator stops. The operating velocity and the maximum acceleration affect the time to stop and the maximum force. The robot inertia and actuator power will influence the maximum acceleration. However, for a position controlled manipulator, the limits are often set in software.

The rate at which the contact force increases, i.e. the slope of the line before response, is dependent on the spring constant of the material of collision and the manipulator velocity. The faster the approach and the stiffer the material, the faster the force will rise. The slope of the force line relates to the significance of the response frequency and time to stop. The lower the slope, the less influence the acceleration, feedback, and response frequency have on maximum collision force. So to reduce the

maximum collision force, the velocity, stiffness, and time to stop should be minimized while the acceleration and response time should be maximized.

4.5. Demonstration

In this section the collision detection method is demonstrated using the SIA5D serial manipulator. The end effector is moved linearly to make contact with materials of different spring coefficients just as in the force validation motions. A collision is detected when the estimation error exceeds the threshold for at least one of the joints. The contact force is measured during the collision by a six-axis force/torque sensor mounted at the end effector. Collisions were detected against Plexiglas, a board on foam, a banana, and a glovebox window. Plexiglas and a board on foam were chosen for their spring constants, capable of developing significant force over a desirable hand displacement. The banana has properties similar to human flesh. The glovebox glass is significant to glovebox automation.

4.5.1. PLEXIGLAS

For the first set of collisions, a piece of Plexiglas 1/8" thick was placed on a wooden frame. The end effector was moved in a Cartesian motion at different velocities. When a joint's estimated contact torques exceeded the associated threshold, a collision was detected. The motion was aborted, bringing the EEF to a stop along the same direction it was moving. Then the EEF reversed directions to alleviate the contact force and torques. The setup is shown in Figure 4.7.

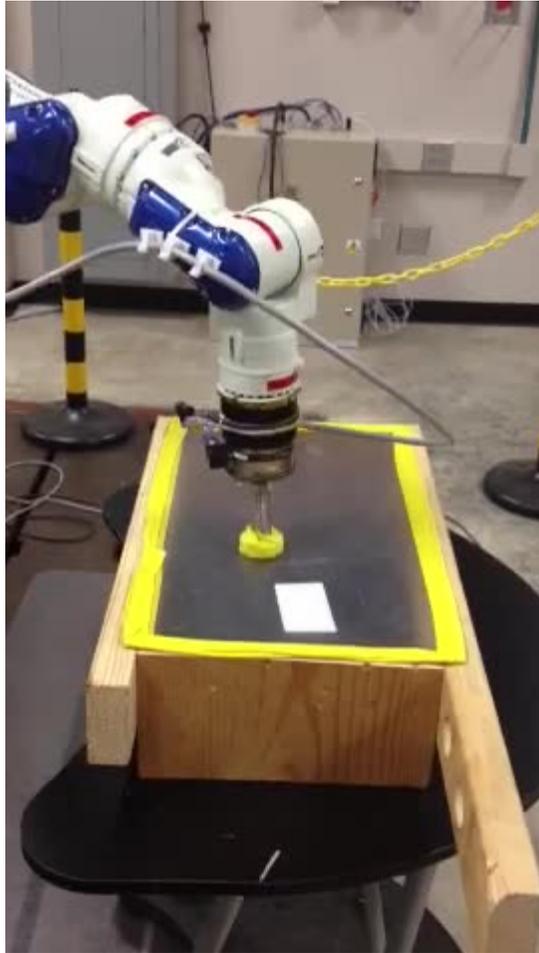


Figure 4.7: Setup for collision detection on Plexiglas target

Collisions were tested at a variety of velocities on the Plexiglas. The thresholds were chosen to reduce false positives while maintaining sensitivity. The velocities and thresholds for the tests are shown in Table 4.2.

Table 4.2: Velocities and joint thresholds for Plexiglas collision tests

EEF Velocity [mm/s]	Threshold [Nm]						
	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
25	20	10	20				
50							
75							
100							
125							
150							
175							
200							

The estimated contact torque for the first motion, moving the EEF at 25mm/s, is shown in Figure 4.8.

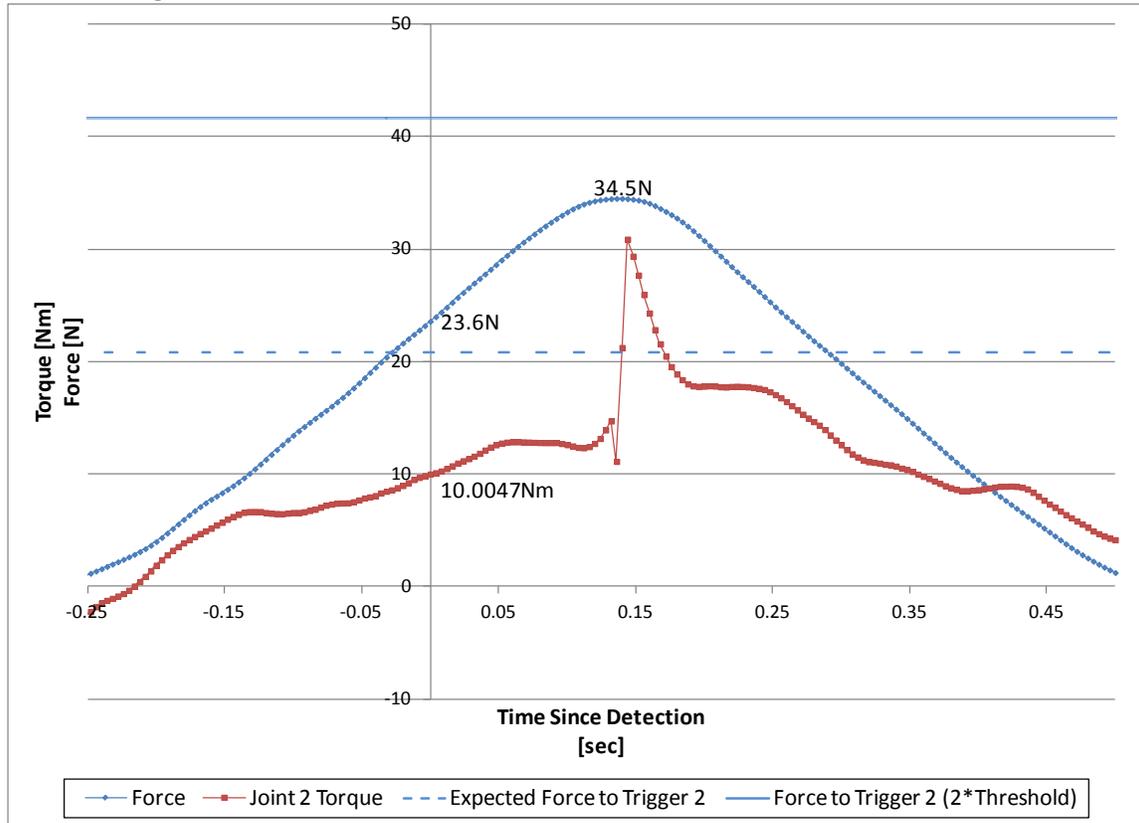


Figure 4.8: Estimated joint 2 torque and measured EEF force at 25mm/s with Plexiglas

The time in the graph has been normalized so that the collision is detected at time $t = 0$. The torque and force at detection are labeled. The torque at detection is barely more than

the threshold torque. The effects of the digital system mean a torque discrepancy, i.e. difference between the threshold and the actual value, of only 0.0047Nm. The force expected to generate a torque equal to the threshold based on the manipulator configuration at the time of detection is plotted as a dashed blue line, roughly 21Nm. The maximum expected detection force, that associated with twice the threshold, is plotted as a thick and thin pair of parallel lines, around 42Nm in this case. The measured force at contact of 23.6N is well below the maximum and very near the expected value.

Collision detection was successful with similar graphs for all the tests except for the tests at 75mm/s, 175mm/s, and 200mm/s. Those tests resulted in a false-positive before contact was made with the Plexiglas.

Table 4.3: Plexiglas tests with forces and torque

EEF Velocity [mm/s]	Force at Detection [N]	Joint 2 Torque at Detection [Nm]	Maximum Force [N]
25	23.6	10.0047	34.4989
50	19.8	10.0006	52.2063
75	0.4	-10.5544	1.28664
100	29.6	10.8505	130.568
125	22.3	11.4633	108.977
150	24.8	10.2377	122.42
175	1.8	-28.5837	4.61985
200	2.9	-58.8579	5.15481

The error during acceleration is higher so collision detection was not enabled until 0.5s after the motion started, to allow the acceleration effects to die out. The false positives occurred where the acceleration effects had not completely disappeared before enabling collision detection. Data for the 175mm/s test are shown in Figure 4.9 below. In co-robotic applications, the velocities will be restricted lower.

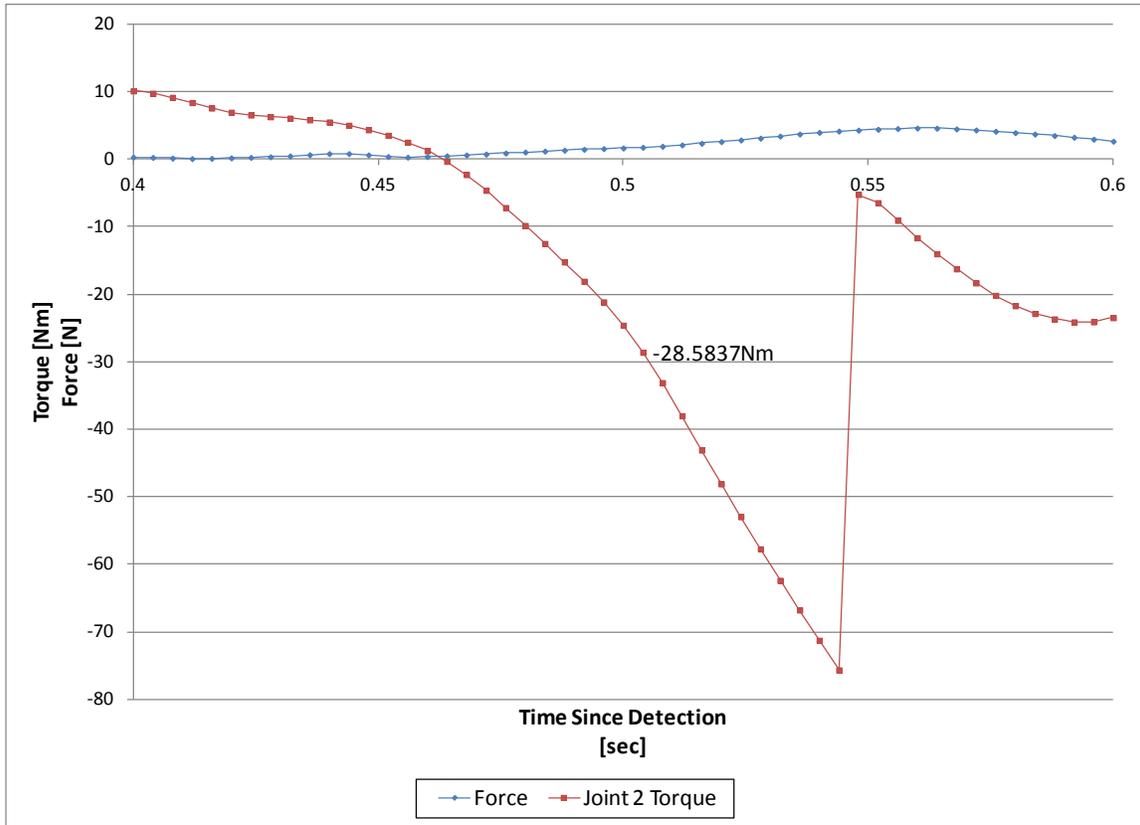


Figure 4.9: Estimated joint 2 torque and measured EEF force at 175mm/s with Plexiglas

The velocities 175mm/s and 200mm/s are too high for the chosen thresholds. Figure 4.10 graphs the data of a 200mm/s Plexiglas collision when all joints have a 20Nm threshold. It is worth noting that EEF speeds this high well exceed any participating EEF speeds expected in a glovebox setting, especially for co-robotic tasks.

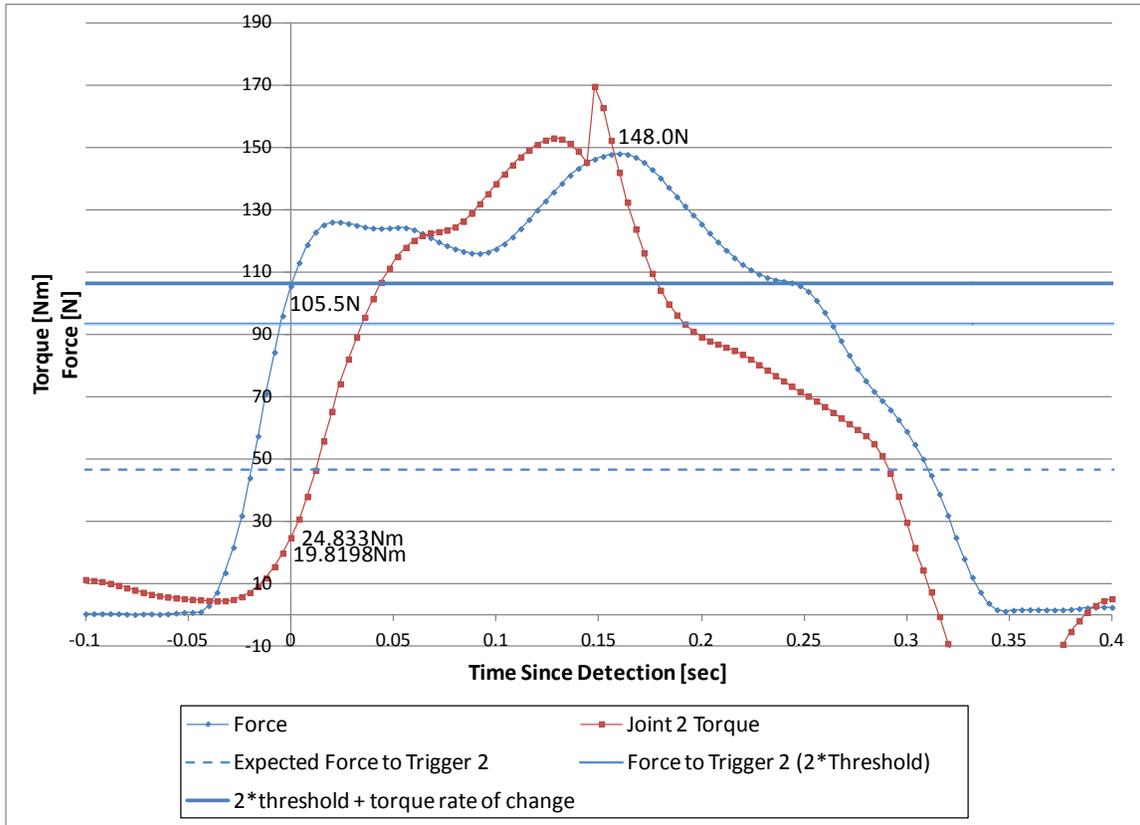


Figure 4.10: Plexiglas collision detection at 200mm/s with 20Nm thresholds

The joint torque at detection is 24.833Nm, almost 5Nm greater than the detection threshold. The torque at the sample before detection is 19.8198Nm. The force at detection exceeds the force predicted by the threshold torque. But when the torque rate of change uncertainty, as discussed in Section 4.4, is accounted for, the trigger force is accurately predicted.

4.5.2. 5"x8" FIBERBOARD ON FOAM

Tests were repeated at the same velocities for the piece of 5x8 fiberboard on foam. The setup is shown in Figure 4.11. The same EEF velocities and joint thresholds were used as for the Plexiglas tests.

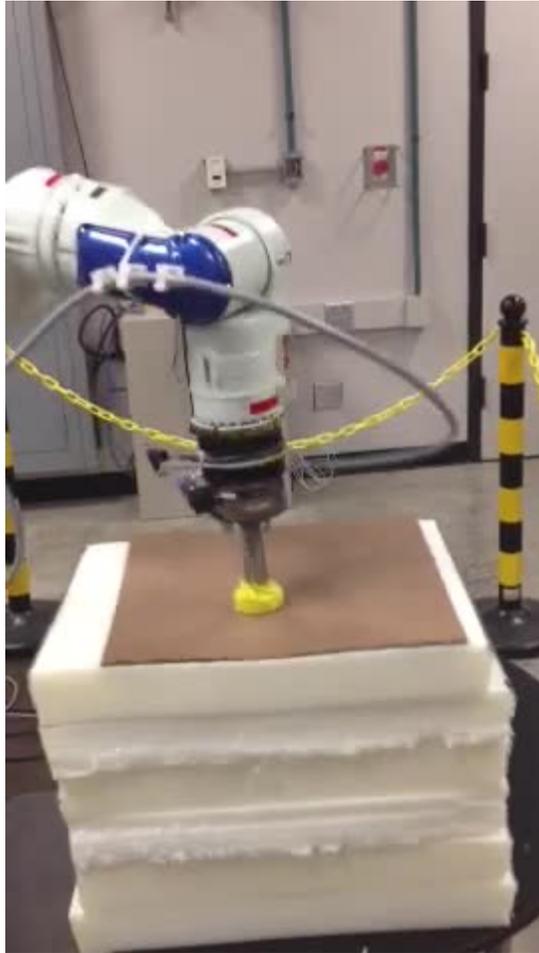


Figure 4.11: EEF collision detection tests with fiberboard on foam

The force and torque profiles looked very similar to the Plexiglas tests. The test at 25mm/s is shown in Figure 4.12.

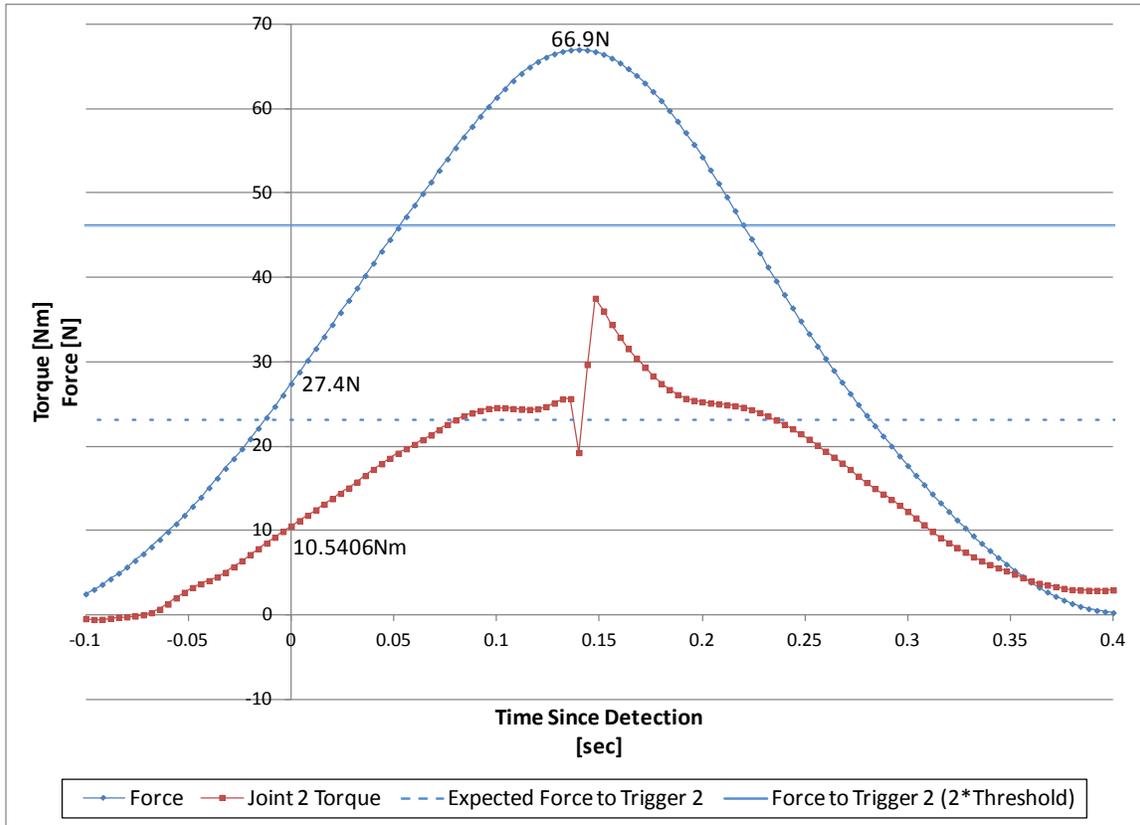


Figure 4.12: Collision detection at EEF 25mm/s on 5x8 fiberboard on foam

The difference between the torque at detection and the threshold torque was higher, in this example and for all velocities. The force at detection was within a few Newtons of the expected value and well below the maximum. Again, most tests with the 5x8 piece of fiberboard on foam yielded similar results. Again, three false positives were recorded. This time at 100mm/s, 175mm/s, and 200mm/s.

Table 4.4: Results of CD on 5x8 fiberboard on foam

EEF Velocity [mm/s]	Torque at Detection [Nm]	Force at Detection [N]	Threshold Exp. Detection Force [N]	Maximum Force [N]	Comments
25	10.5406	27.42	23.07	67.00	
50	10.8256	31.16	22.84	97.24	
75	11.3673	34.95	22.71	136.74	
100	-10.2357	0.56	N/A ^[1]		False Positive
125	11.2073	36.47	22.57	140.33	
150	10.7165	32.12	22.81	142.28	
175	10.1609	0.23	N/A ^[1]		False Positive
200	10.2594	0.69	N/A ^[1]		False Positive

[1] The expected detection force utilizes the measured force to determine force detection. Because detection occurred before an actual collision, the measured force cannot be calculated.

For the tests on the fiberboard, force at detection exceeds the threshold-predicted force. With a mean torque estimation error of zero, detection forces below and above the threshold predicted value are expected. This is likely because 1) the error characteristics change when the end effector makes contact and 2) the torque (ergo the force) at detection is necessarily greater than the threshold.

The false positives triggered early but in the 100mm/s test, significant contact was still made.

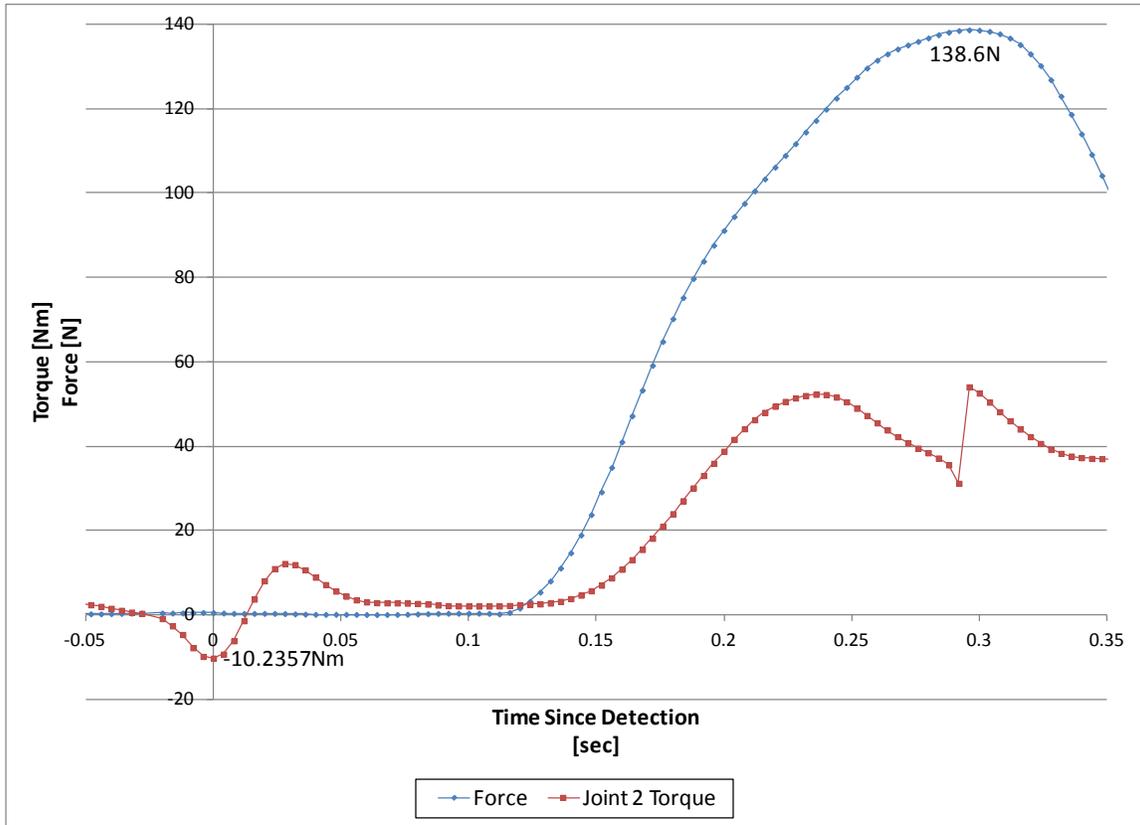


Figure 4.13: Collision detection at EEF 100mm/s on 5x8 fiberboard on foam

The contact force still reached 138.6N despite triggering collision detection before making contact. In this case, the collision was triggered due to excessive acceleration. Just before the ripple in the joint torque, at -0.024s, a data point is missing. This caused an artificial ripple in the estimated acceleration.

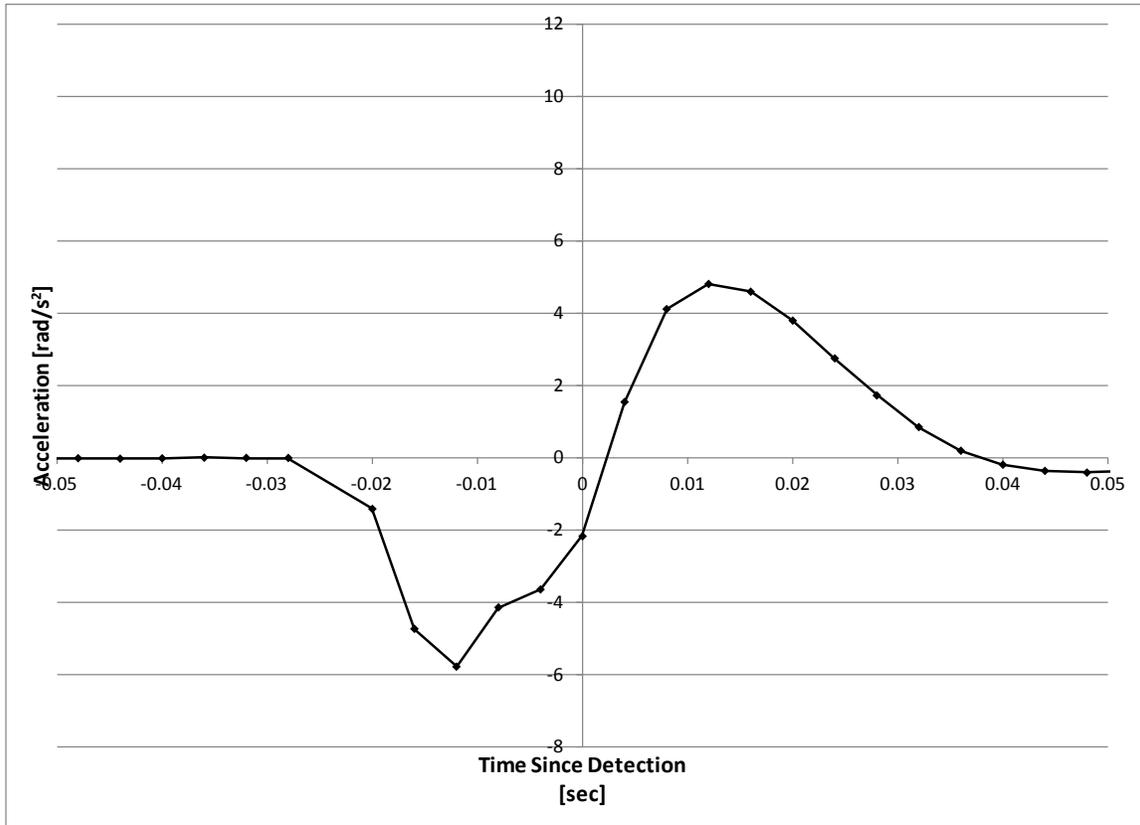


Figure 4.14: Acceleration at EEF 100mm/s on 5x8 fiberboard on foam

The other two false positives occurred because the velocities (175 and 200mm/s) were too high for the chosen thresholds, consistent with the Plexiglas tests.

4.5.3. GLOVEBOX GLASS

The glass is a critical part of glovebox containment. Glass broken in a robot collision may permit contaminants and radiation to harm humans in the area. The collision detection system was tested by moving the EEF against the glass of a glovebox. The manipulator was setup so that the end effector approached the glovebox glass at an angle perpendicular to the glass, as in Figure 4.15.

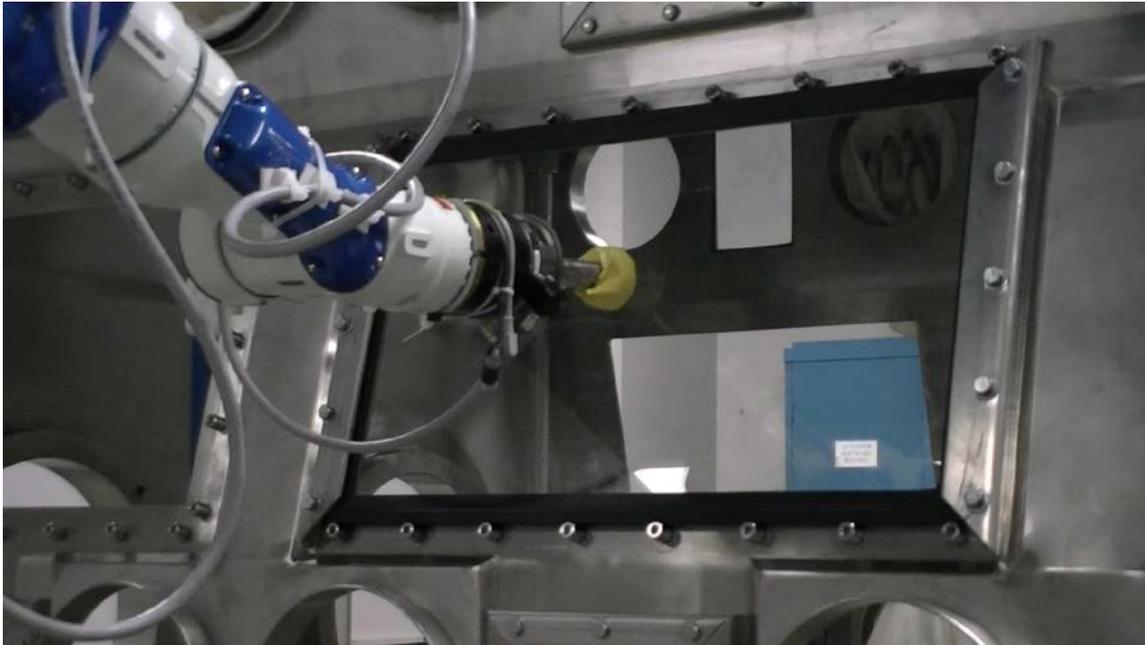


Figure 4.15: Setup of collision detection tests with glovebox glass

The glovebox glass was tested with the same torque thresholds as the Plexiglas and 5x8 fiberboard on foam. The velocity was reduced to EEF speeds more closely representing those that might be expected in a glovebox. The test velocities and thresholds are shown in Table 4.5.

Table 4.5: Glovebox glass collision detection results

EEF Velocity [mm/s]	Threshold [Nm]		Torque at Detection	Threshold [Nm]				
	Joint 1	Joint 2		Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
12.5	20	10	10.077	20				
12.5			10.0865					
12.5			10.2956					
12.5			10.0548					
12.5			10.008					
12.5			10.3158					
12.5			10.5145					
25			10.0402					
25			10.2564					
25			10.4315					
25			10.035					
25			10.3394					
37.5			10.6593					
37.5			10.1706					
37.5			10.6765					

Unfortunately, the force sensor failed during the glovebox glass tests so force measurements could not be taken. Again, the detection occurred in joint 2 for every test. The joint 2 torques at detection are shown in Table 4.5. The cells are colored green for the lowest detection torque and red for the highest. In every case the collision was successfully detected and the glovebox glass didn't break.

4.5.4. BANANA

Lastly, collisions were detected against a banana. It was infeasible to test with human subjects but a banana has a soft flesh susceptible to bruising. The setup for testing collisions with the banana is shown in Figure 4.16.

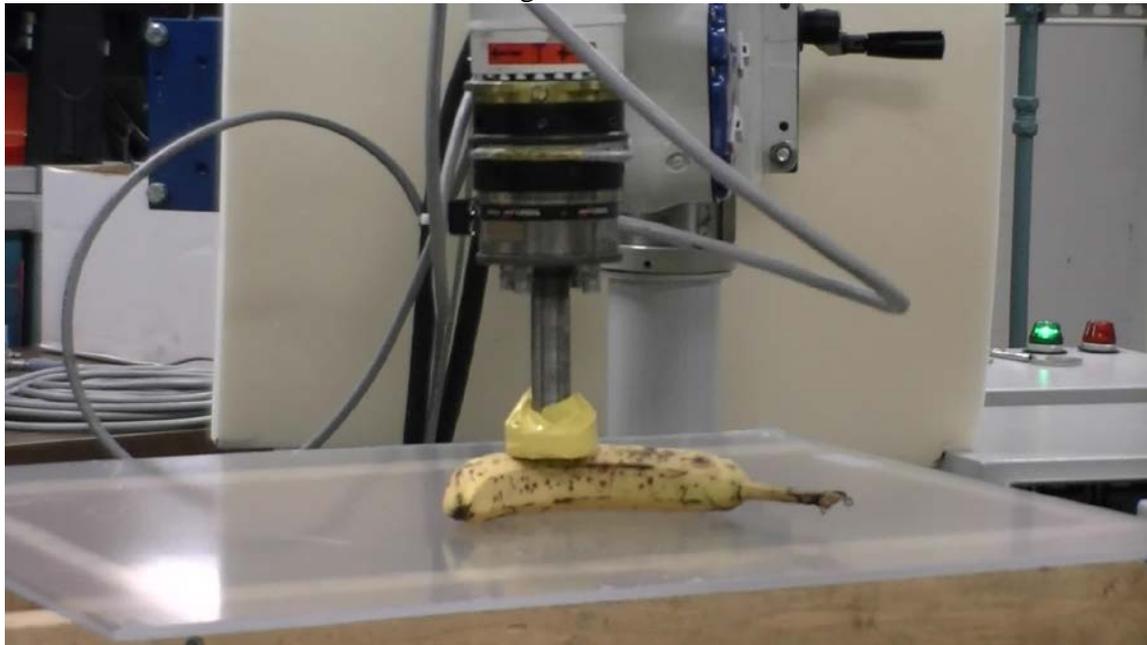


Figure 4.16: Setup for collision detection with banana

The collisions were performed at lower velocities. The detection threshold for joint 2 was lowered to 7Nm to increase the sensitivity. The EEF velocities and joint thresholds are shown in Table 4.6.

Table 4.6: Banana collision detection results

EEF Velocity [mm/s]	Threshold [Nm]		Detection Torque [Nm]	Threshold [Nm]					Exp. Detection Force [N]	Detection Force [N]	Maximum Force [N]
	Joint 1	Joint 2		Joint 3	Joint 4	Joint 5	Joint 6	Joint 7			
5	20	7	7.1482	20					16.2	14.9	18.4
10			7.2275						15.7	18.4	28.0
15			7.3942						15.6	15.3	39.9
17.5			7.3371						15.7	15.8	63.6

Results for the banana collisions were similar to the other tests. The test at 5mm/s is shown in Figure 4.17 as an example.

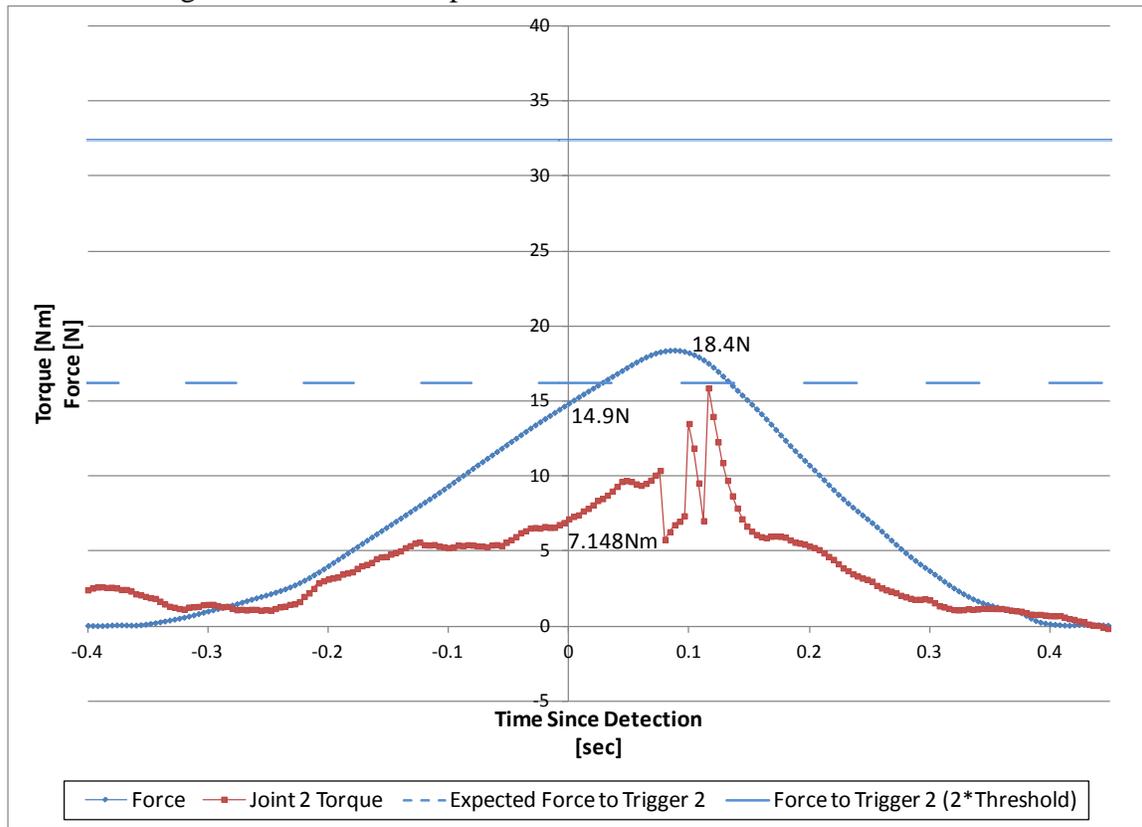


Figure 4.17: Collision detection at EEF 5mm/s with banana

The maximum force for the highest velocity (Figure 4.18) was 63.6N (14.3lb). For the banana tests, two of the four tests triggered at a force below the threshold-expected detection force. The tests resulted in a noticeable, though slight, deformation of the banana. Despite the advanced ripeness of the banana (Figure 4.16), the skin stayed intact and the banana proved edible.

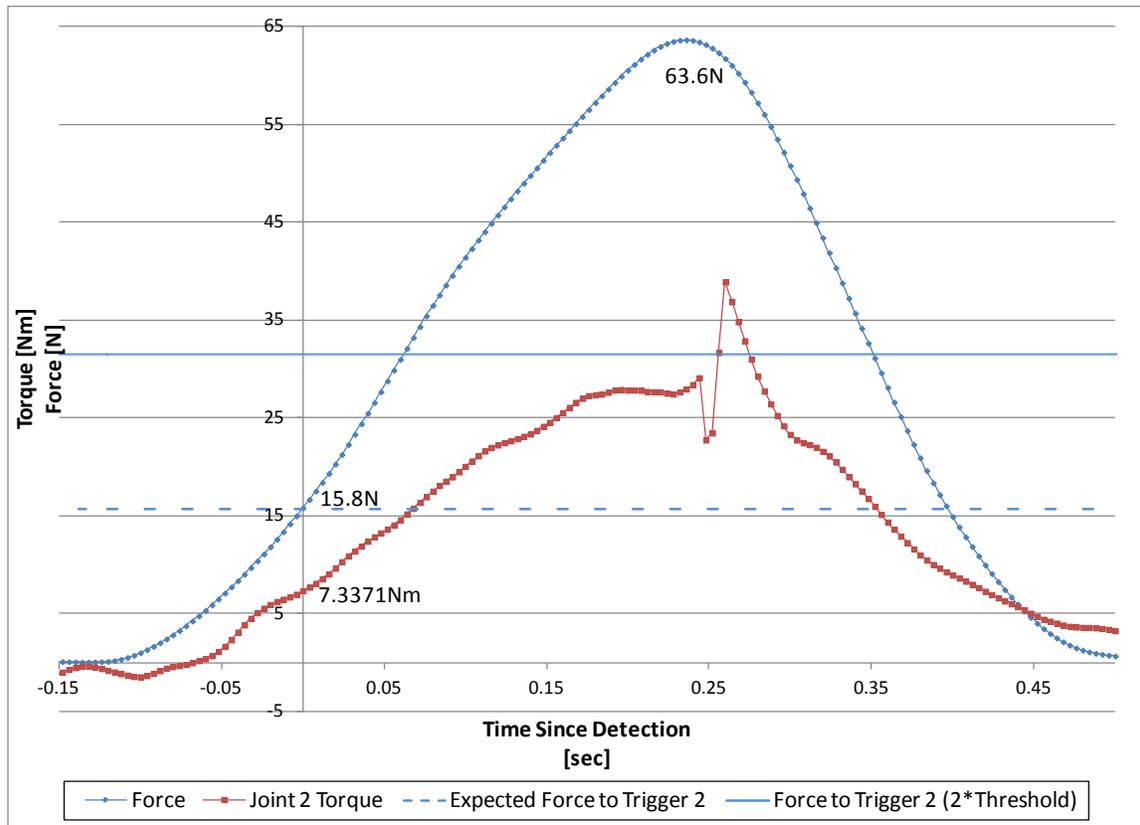


Figure 4.18: Collision detection at EEF 17.5mm/s with banana

4.6. Summary

The black box estimated joint torque is used for collision detection with a variety of objects, ranging from a ripe banana to the brittle glovebox glass. Collisions are detected and the response is sufficient to prevent breaking glovebox glass or damaging the banana. The force at detection is estimated using the direction of the measured force. The direction of contact is not available during normal online collision detection but the maximum force at detection is important for simulating collisions in the next chapter and to evaluating the effectiveness of the collision detection system for human safety. The next chapter will use the presented data, as well as other results from collision detection tests to develop and validate a collision detection simulator.

5. COLLISION SIMULATION

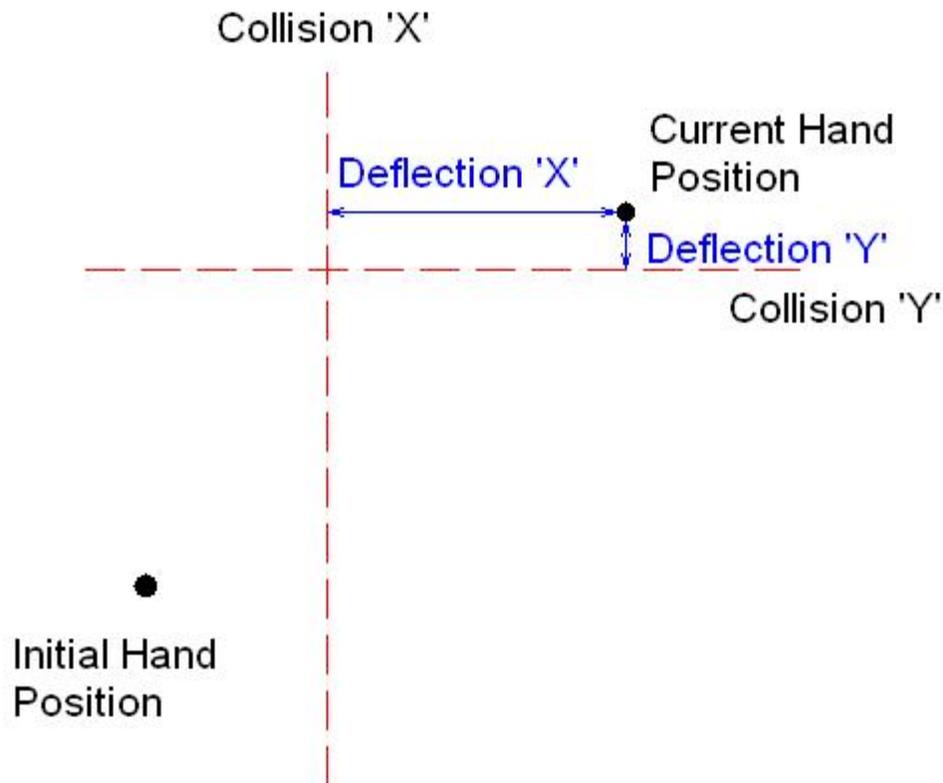
Effective collision detection was demonstrated with several objects in the last chapter. One of the major objectives of this dissertation is to study the effectiveness of collision detection for improving the safety of human-robot interaction. It is not feasible to conduct experiments with human subjects. In order to gain some understanding of the effectiveness, a collision simulator is developed from the experimentally validated models developed and tested in Chapters 3 and 4.

In the next chapter, the effectiveness of collision detection for human safety will be studied, but for now it is relevant to briefly introduce the BGIA report on human robot safety. [BGIA 2011] The report provides linear spring constants for contact with different parts of the body. The spring constants will be used to simulate the force that is generated between human and robot during a collision. The report also provides maximum acceptable forces and compression distances for different parts of the body. The data provided by the BGIA will be used to simulate collisions with the human body. Then the results of the simulation will be compared to the maximum allowable forces and pressures.

First the simulator is presented. Then the simulated data are compared to experimental data for the 50mm/s Plexiglas test. Finally a few observations will be made about the performance of collision detection given different system parameters, i.e. how is the maximum force during collision affected by changing the threshold, velocity, feedback frequency, etc.

5.1. The simulator

The collision simulator is coded in C++ using Microsoft Visual Studio. The simulator uses a modified PID controller to move the end effector at a constant linear velocity. The object of collision is simulated as a distance from the initial hand position. Forces on the end effector are generated based on the distance beyond this collision limit.



Based on the model for human body parts presented in the next chapter, forces are linearly proportional to the compression distance.

$$F_x = k_x d_x \quad (5.1)$$

The force in the x direction, F_x , is the product of the effective spring constant in that direction, k_x , and the deflection in that direction, d_x .

The joint torques due to the end effector force are calculated by OSCAR dynamics libraries. This is actual contact torque in the joints. Noise in the estimated contact torque is simulated by a normally distributed number. The mean and standard deviation of the noise are inputs to the simulator.

A collision is detected when the simulated contact torque exceeds the chosen detection thresholds. The manipulator responds by slowing as quickly as possible while maintaining the desired EEF trajectory. The force continues to increase during this phase.

The simulator parameters for the object and the robot can be changed to signify different or improved robots. On the object side of the collision, the stiffness can be changed to simulate different objects, such as human flesh. The robot configuration, velocity, and acceleration can be adjusted to simulate typical motions. The joint detection thresholds can also be adjusted to simulate improvements to the black box model or a different manipulator for which the model estimates are not as accurate. The maximum force and object compression distance are the relevant outputs. The outputs and inputs are listed in Table 5.1.

Table 5.1: Simulator input and output parameters

Input	
θ_{init}	Initial joint positions
\dot{x}_d	Desired EEF velocity
$\ddot{\theta}_{max}$	Maximum joint accelerations
x_{coll}	Distance to collision relative to initial hand position
K	Object stiffness
Δt	Controller/feedback sampling rate
$\bar{\tau}_{err}$	Estimated torque error mean
σ_{τ}	Estimated torque error standard deviation
τ_{thresh}	Detection threshold torques
n	Number of simulations
Output	
x_{comp}	Compression distance during collision
F	Collision force

The simulator is a windows console application. The system asks the user to keep or change the default input values. The system then performs the simulations a number of times. Output data from each simulated collision are saved to a text file. The text file contains the information necessary for statistical analyses of the simulations.

One possible instance of the simulator is illustrated in Figure 5.1. The manipulator is positioned with the end effector pointing downward, as if it were reaching for or lowering an object. In the figure, the red box and spring indicate the object being collided with. The red arrow indicates the force generated on the manipulator during the collision while the blue arrow indicates the velocity of the end effector.

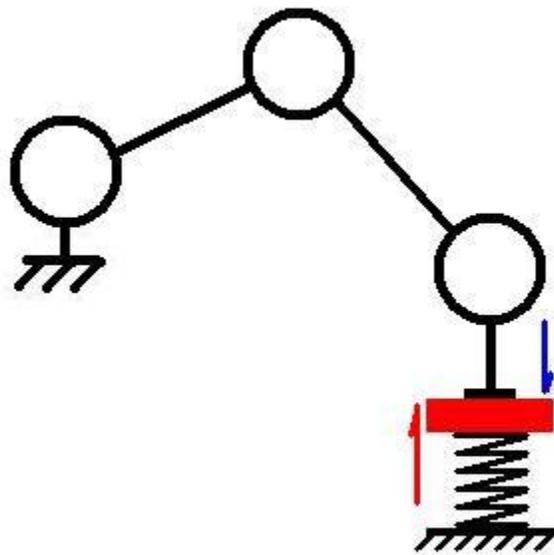


Figure 5.1: Robot collision simulator diagram

The manipulator dynamics and kinematics are simulated at a faster rate than the controller. A block diagram of the simulator is shown in Figure 5.2. The large dashed rectangle encloses the components simulated at the faster rate. Simulating these components at a higher rate captures real-time operation of the manipulator while the controller responds at a limited rate. Information is only available to the controller, the smaller dashed box, at the lower controller/feedback rate, Δt .

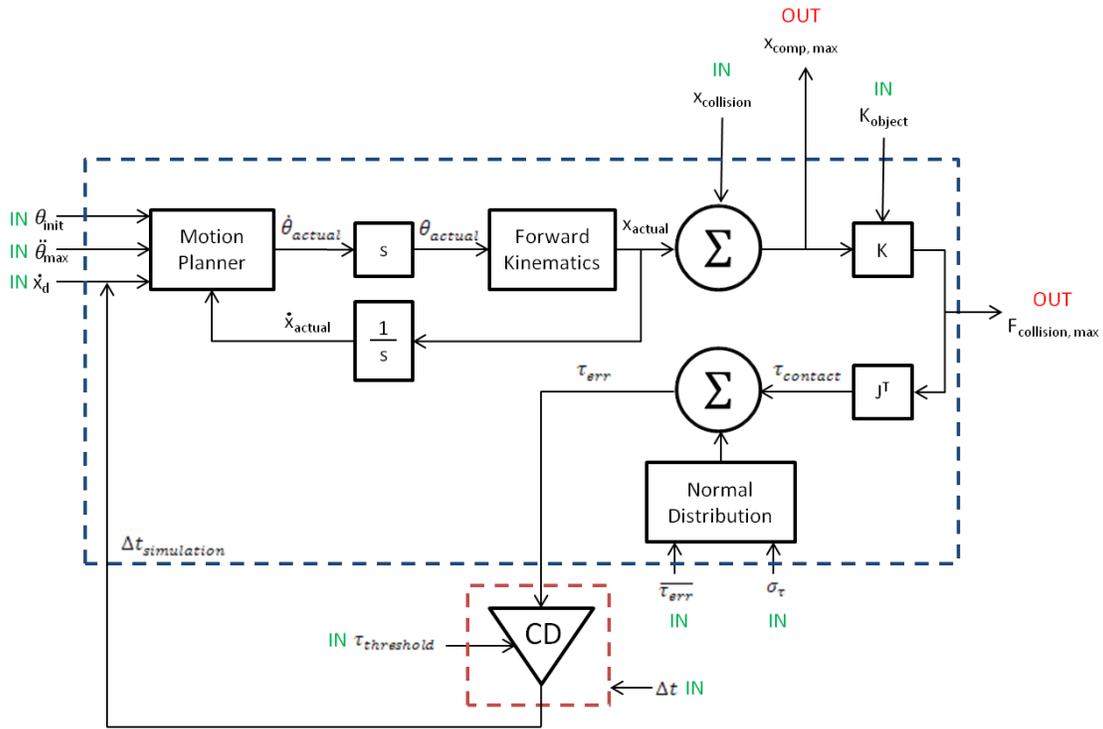


Figure 5.2: Simulator block diagram

The uncertainty and noise in the torque signal are modeled as a normal distribution. The torque uncertainty mean and standard deviation are inputs. The simulation is run many times to estimate statistical maximum collision force and compression.

An example set of simulation data are shown in Figure 5.3. The simulated and feedback torque are both shown relative to the time of collision detection. The black line indicates the simulated torque, calculated at the rate indicated by $\Delta t_{simulation}$. The controller torque is only evaluated at the feedback rate of the controller, Δt . The red line indicates the deflection of the object in mm. The blue line is the force between the object and the manipulator.

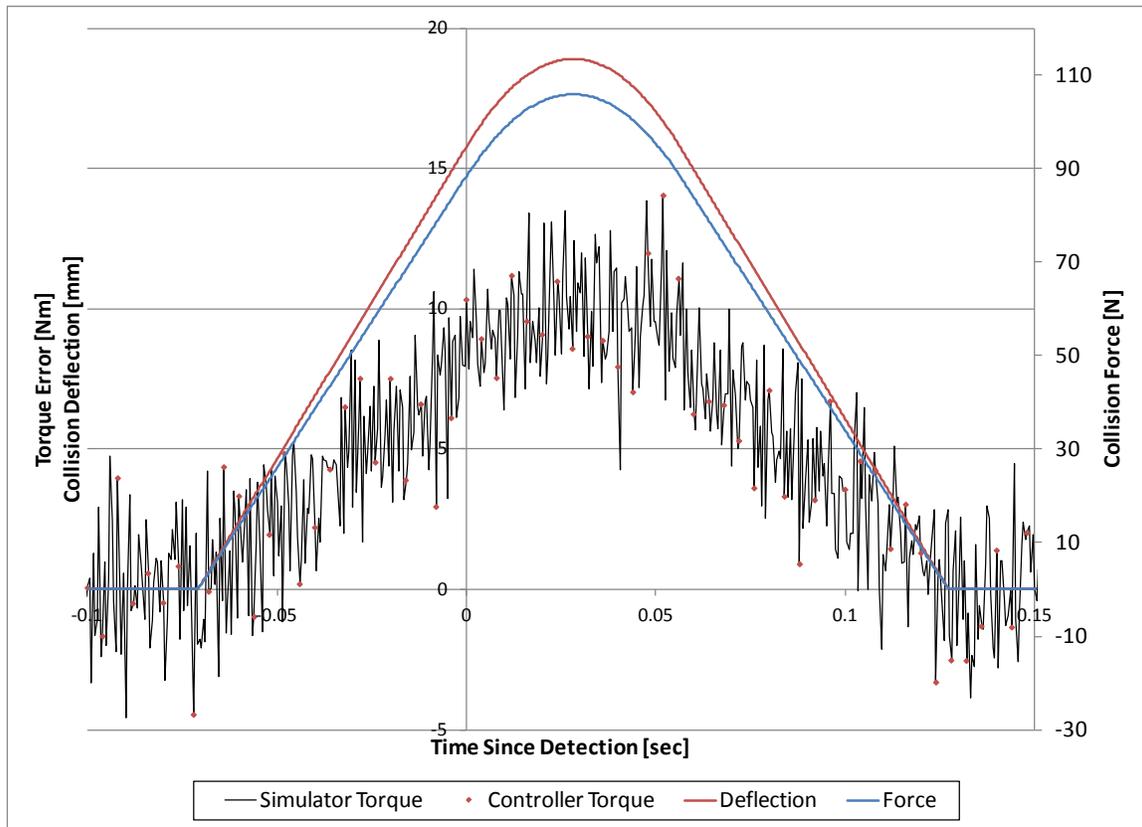


Figure 5.3: Example collision simulation data

In the example, the simulated torque exceeds the detection threshold at a time when the controller does not receive feedback so the collision is not detected. In order to capture the statistical effects of the uncertainty, the simulations are run many times. For each of these simulations, the torque and force at detection and the maximum force and deflection are recorded. The data are then aggregated to get the statistical results.

5.2. Validation

Experimental results are compared to these simulated results for the Plexiglas at 50mm/s. The stiffness of the Plexiglas was estimated from the measured force and hand displacement during the experimental collisions. The force was plotted against the displacement and a line was fitted to the data. An example is shown in Figure 5.4.

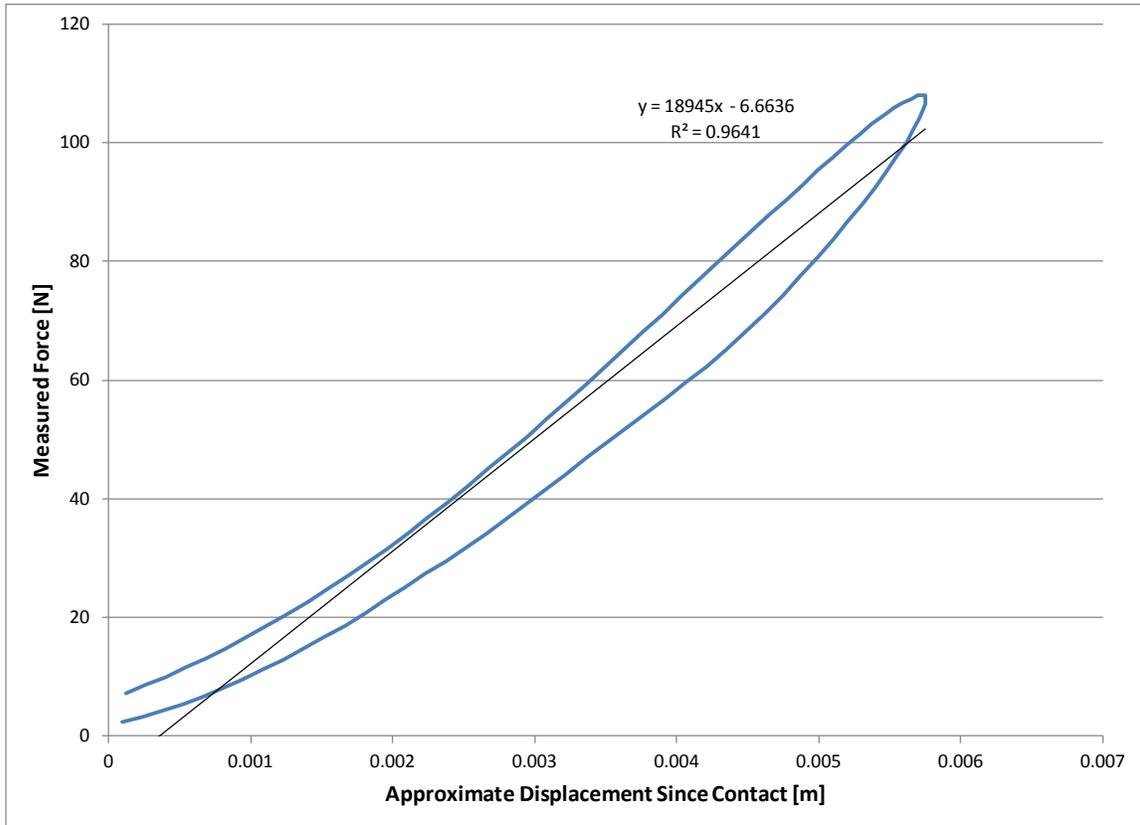


Figure 5.4: Example Plexiglas stiffness estimation

In the data illustrated, the stiffness is 18900N/m. The approximate stiffness for the three experiments combined was 19100N/m. The maximum force for each of the three experiments was roughly 108N.

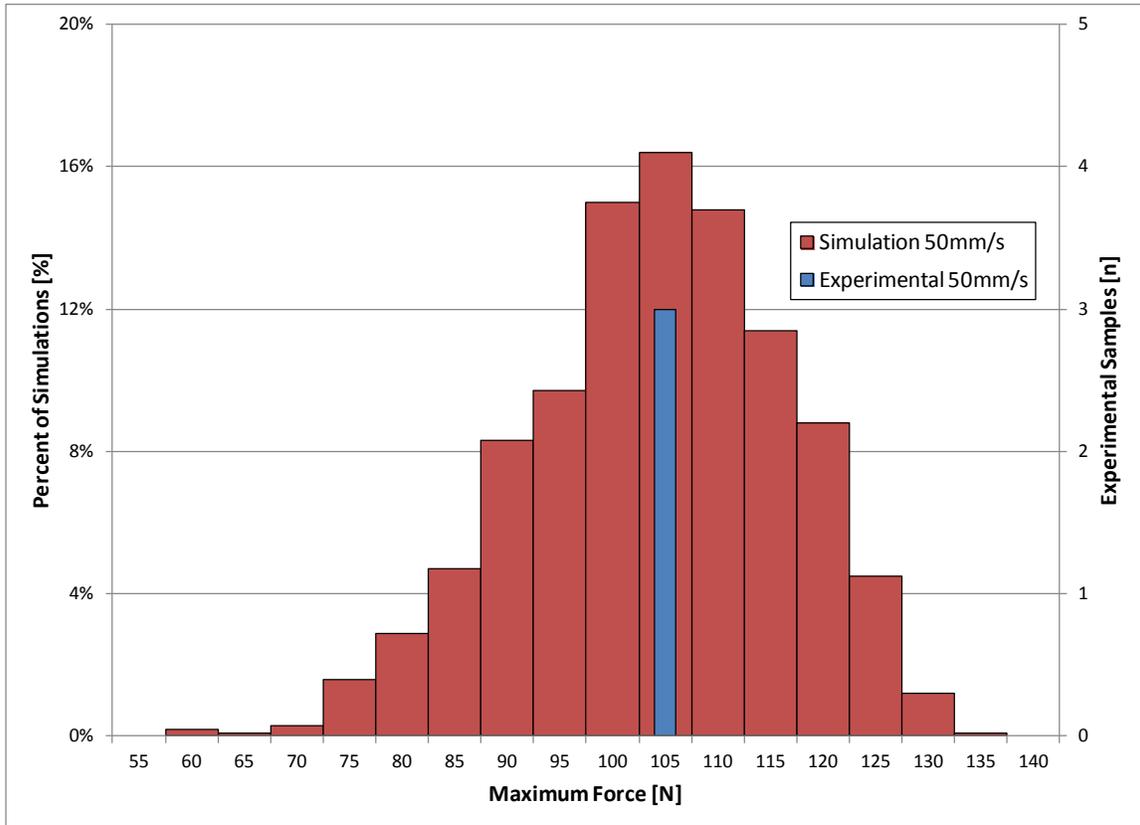


Figure 5.5: Simulated ($K=19100\text{N/m}$) and experimental results for 50mm/s collisions with Plexiglas

All three of the experimental results are in the same bin, the most likely bin according to simulations. The simulated results are representative of the experimental results.

5.3. Parameter comparison

As was seen in the validation experiment, the simulator doesn't incorporate variability in any parameters except the estimated contact torque. The system has variable noise, but the other parameters are fixed for a given situation. The effects of other input parameters are compared by running 1000 simulations with each parameter changed. The average and standard deviation of maximum force for each parameter is plotted. The basic set of parameters is shown in Table 5.2.

Table 5.2: Parameters for control-group simulations

Parameter	Control-group Parameter Value
Maximum Joint Acceleration	[4.5, 4.5, 4.5, 4.5, 4.5, 4.5, 4.5]rad/s ²
Detection Thresholds	[20, 10, 20, 20, 20, 20, 20]Nm
Noise Average	0Nm
Noise Standard Deviation ^[1]	[3.636, 1.818, 3.636, 3.636, 3.636, 3.636, 3.636]Nm
Object Stiffness	5600N/m
EEF Velocity	50mm/s
Feedback Rate	0.004s

[1] The noise standard deviation was set to 1/5.5 of the detection thresholds by default

Increasing the velocity, as in Figure 5.6 and Figure 5.7, increases the maximum force exponentially. As the velocity increases, the force increases, as noted in Figure 5.6 by the shift to the right. The distribution also becomes shorter and spreads, indicating an increase in standard deviation.

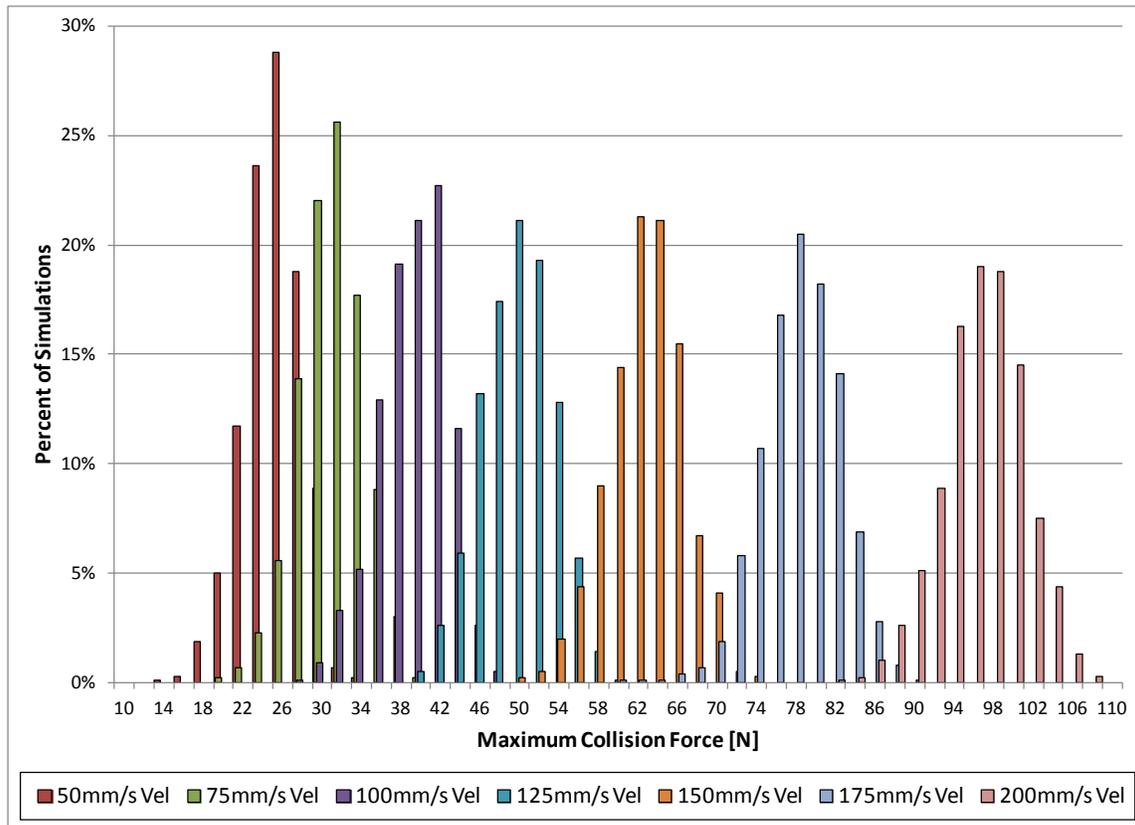


Figure 5.6: Distribution of maximum force for 1000 samples at different EEF velocities

A second-order polynomial can be fitted to the maximum force versus the end effector velocity. Increasing the velocity, v , with a fixed acceleration, a , will increase the stopping distance, x , exponentially.

$$x = \frac{v_0^2}{2a} \quad (5.2)$$

The standard deviation increases slightly. As the velocity increases, the rate of change of torque increases, so the variation in the maximum force also increases.

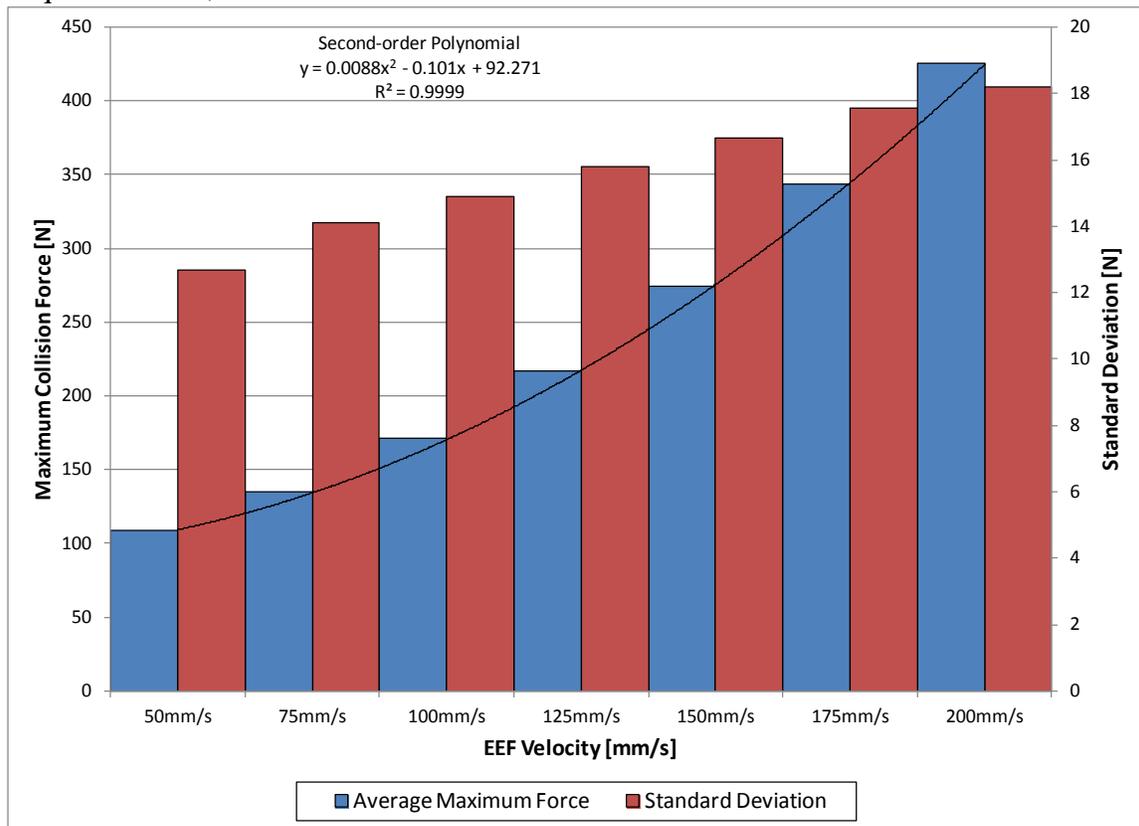


Figure 5.7: Collision force dependence on eef velocity

The dependence on object stiffness was also checked. (Figure 5.8) Increasing the object stiffness increases the maximum force linearly. The standard deviation also increases, but only minimally.

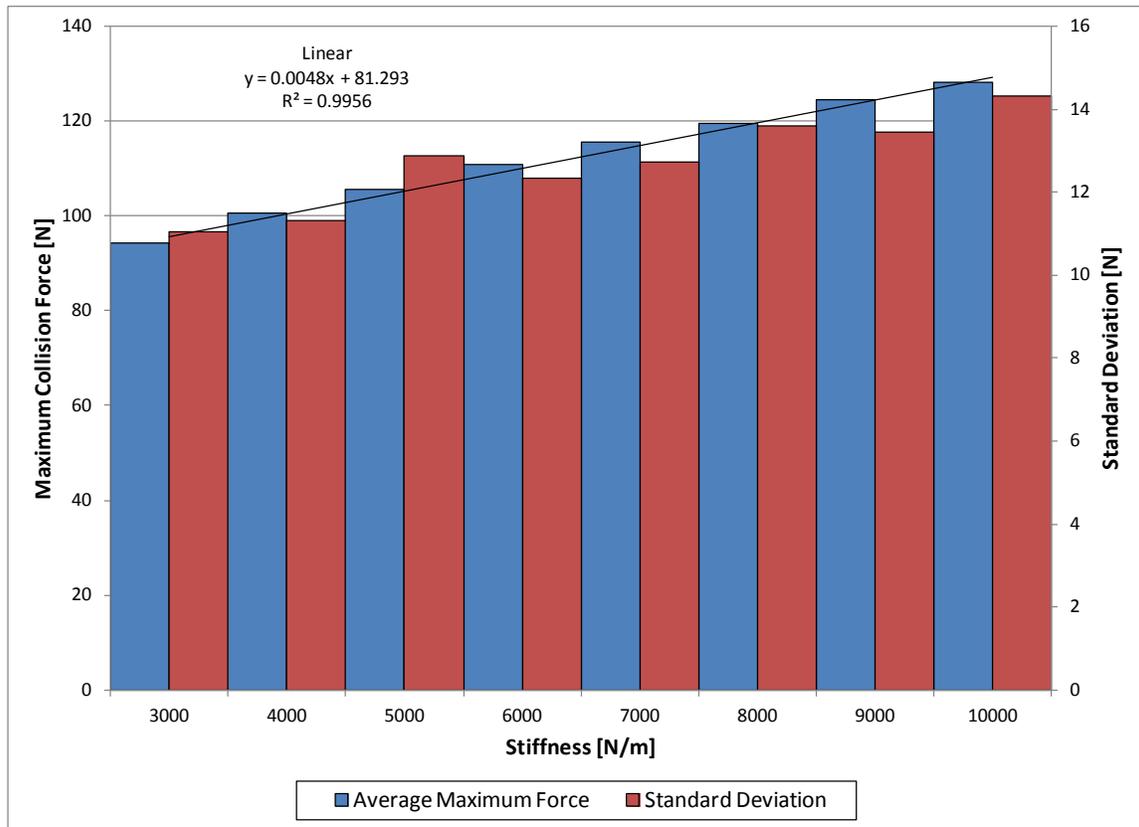


Figure 5.8: Collision force dependence on stiffness

Increasing the detection threshold (Figure 5.9) increases the average maximum force linearly. To minimize false positives in a real system, the threshold might be chosen based on the standard deviation of the torque estimation error. For these simulations, the threshold is chosen to be 5.5 times the standard deviation of the estimation error. However for simulations, it is more logical to select the simulated threshold and then calculate the standard deviation of the estimation error. Because the estimation error standard deviation is changed, the standard deviation of the maximum collision force also changes.

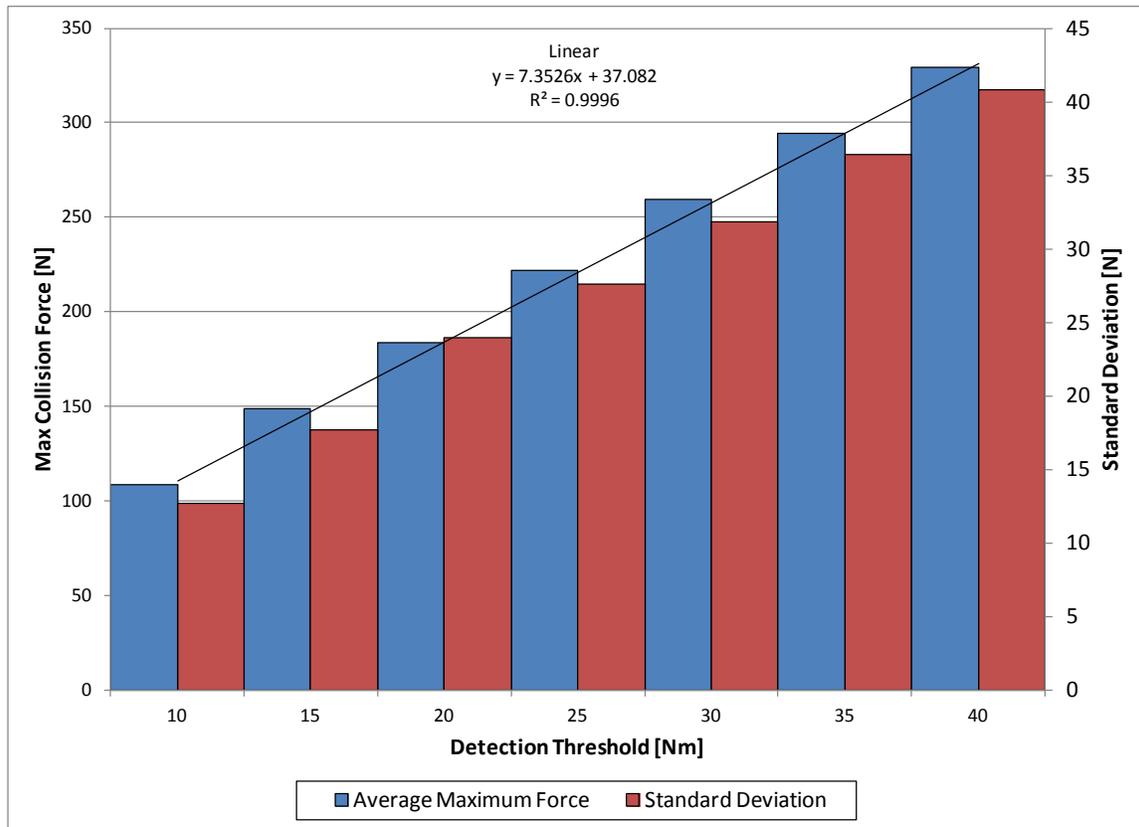


Figure 5.9: Collision force dependence on detection threshold

Next the feedback rate was changed. It was changed as low as 0.001 seconds and as high as 0.016 seconds. The maximum force has a power dependence, $y = Ax^B$, on the feedback rate. The standard deviation also increases slightly with the increasing time between feedback. This is to be expected as the increased time between samples allows the force to change more before detection.

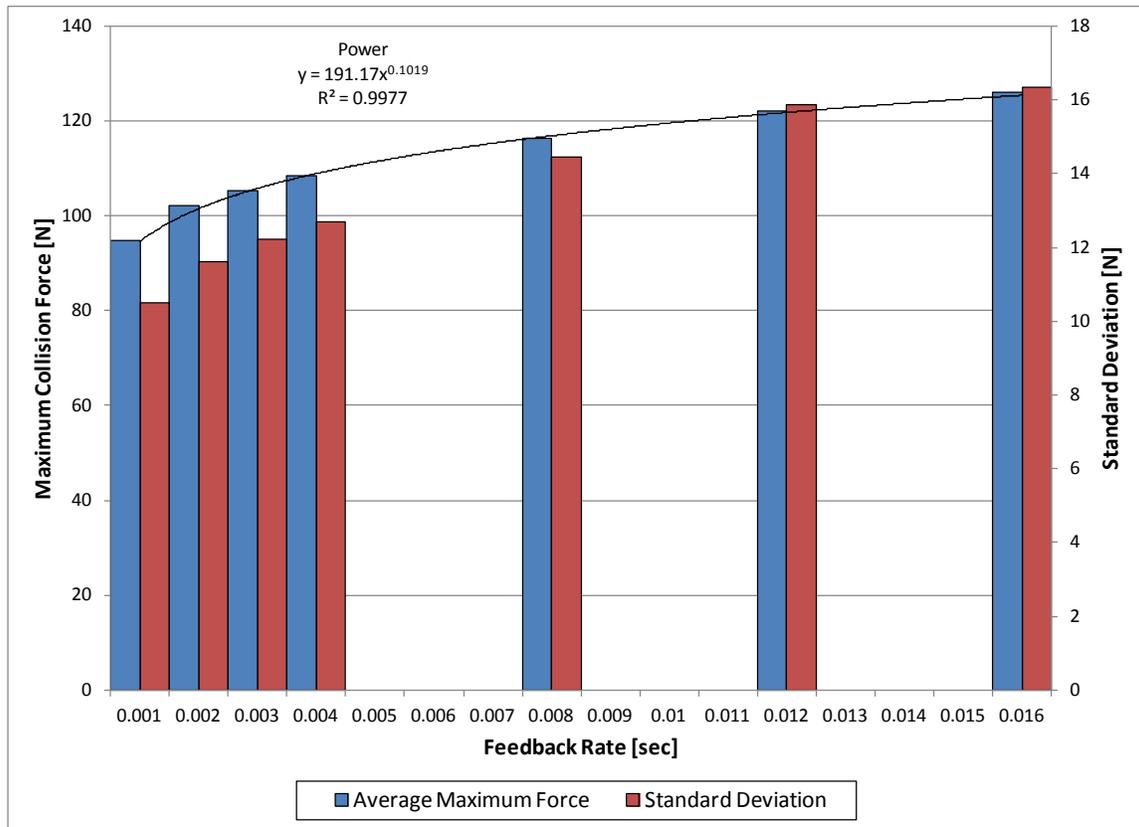


Figure 5.10: Collision force dependence on controller feedback rate

Increasing the joint acceleration decreases the maximum collision force. This relationship is also best described by a power equation, $y = Ax^B$. The acceleration doesn't affect the standard deviation. Only the distance to stop after detection is affected by the maximum acceleration.

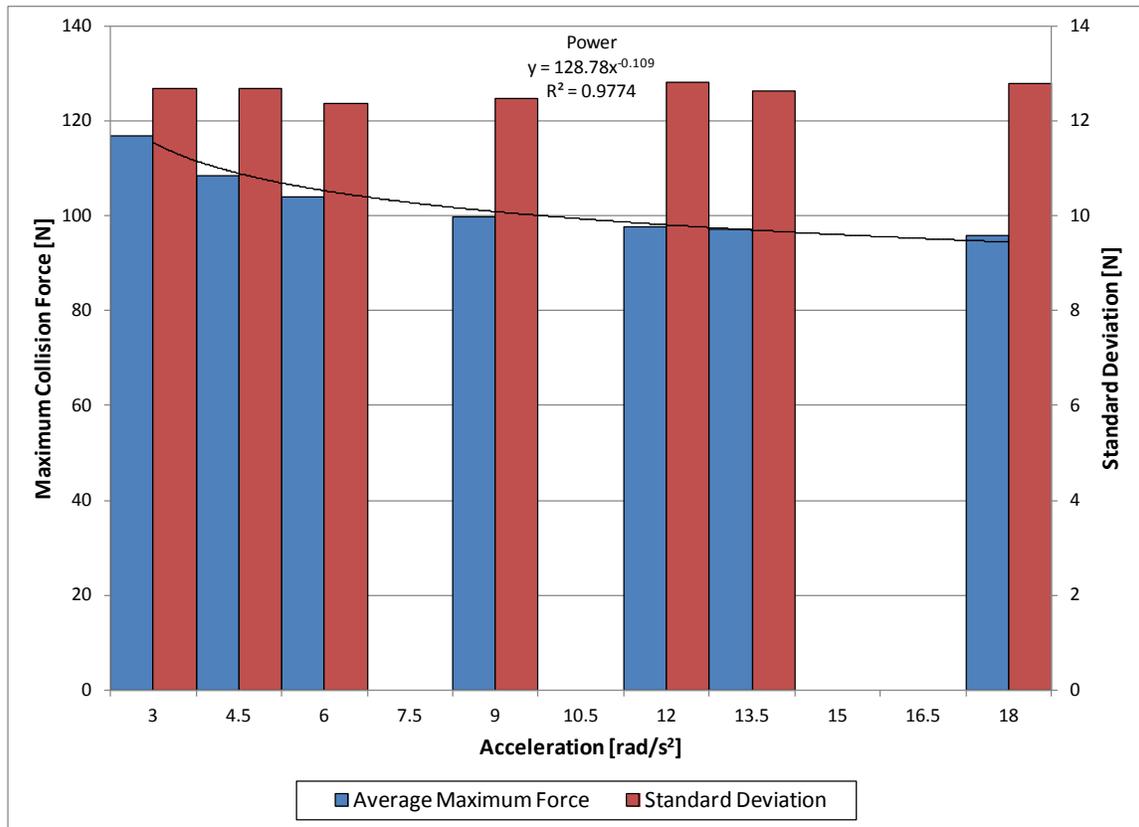


Figure 5.11: Collision force dependence on maximum joint acceleration

The relationships between parameters and maximum force are summarized in Table 5.3.

Table 5.3: Relationship between parameters and maximum force

Parameter	Effect of Increasing Parameter	Relationship
Object Stiffness	Increase Force	Linear
EEF Velocity	Increase Force	Quadratic
Detection Thresholds	Increase Force	Linear
Feedback Rate	Increase Force	Power
Maximum Joint Acceleration	Decrease Force	Power

The approximate relationship between the maximum force and each parameter is shown on each parameter chart. The ‘instantaneous’ change in force due to a relatively small change in parameter is the derivative of the relationship at the point of interest. The derivatives and small changes are shown in Table 5.4.

Table 5.4: Instantaneous change in force for small change in given parameter

Parameter	Approximate Change in Force for Change in Parameter (d/dx)	d/dx at Control Settings	Change of Parameter	Change in Maximum Force [N]
Object Stiffness [N/m]	0.0048	0.0048	-1000	-4.8
EEF Velocity [mm/s]	$0.0176v_0 - 0.101$	0.779	-10	-7.79
Detection Thresholds [N]	7.3526	7.3526	-5	-36.763
Feedback Rate [s]	$19.48\Delta t_0^{-0.8981}$	2774.5	-0.001	-2.7745
Maximum Joint Acceleration [m/s^2]	$-14.037a_0^{-1.109}$	-2.648	1	-2.648

For example, at the control settings used for simulations, the instantaneous change in force due to a change in stiffness is $0.0048 \frac{N}{N/m}$. If the stiffness was decreased by $1000N/m$, the maximum force would be expected to decrease by about $4.8N$.⁹

The simulator and these results can be used as a tool when designing or implementing collision detection on a robotic system. When evaluating proposed changes to the system, the cost of one parameter can be balanced against the benefits of one or more other parameters. For example, if the velocity must be increased, the decrease in feedback rate or joint acceleration needed to compensate for the velocity effect can be estimated. The table and the simulation charts on the previous pages can also be used with a cost estimate for changing each parameter to determine the most efficient means for improving the system.

5.4. Chapter Summary

Collisions are simulated by generating a force at the end effector based on a linear spring model. Collisions are detected when the estimated joint torque exceeds the threshold. The simulation continues until after the end effector slows to a stop and reverses direction. The estimation uncertainty is studied by running the simulations many times with randomly distributed estimated torque noise.

⁹ The relationship between the force and the stiffness is linear, so the change in stiffness can be arbitrarily large and still give an accurate prediction for the change in force. For the EEF velocity, feedback rate, and maximum acceleration, the relationships are not linear so the change in each parameter must be small.

Collisions with the Plexiglas target are simulated using the experimentally estimated spring constant. Simulations accurately predict the experimental results.

The effect of changing system parameters on the maximum collision force is studied. The maximum collision force is dependent on the square of the velocity, i.e. increasing velocity has a large incremental effect on the maximum collision force. The improvement to force by decreasing the feedback time step grows exponentially.

This chapter validated the simulator and evaluated system parameters which affect the maximum collision force. The next chapter will use the simulator with the characteristics of the human body. Collisions with the hands and arms will be simulated in different manipulator configurations and with different velocities.

6. EFFECTIVENESS OF COLLISION DETECTION

Collision detection was demonstrated using the SIA5D serial manipulator in Chapter 4. It was shown – as one example – that collisions with glovebox glass could be effectively detected and responded to. Effectiveness of the collision detection system with human targets must be explored in other ways. A brief introduction discusses some key points from the literature. Collisions with human hands and arms are simulated using the technique presented in the previous chapter. The effectiveness of the collision detection system is discussed based on the simulation results.

6.1. Introduction

Haddadin et al. [2008] consider two categories of human-robot collisions: blunt contacts and penetrating/slicing/puncturing contacts. This distinction has particular importance when handling hazardous materials in a containment environment such as a glovebox. Intuitively, a puncture collision has more injury and damage potential. Whereas the blunt collision causes discomfort, the puncture may break protective barriers (e.g. gloves, skin, etc) creating a direct path for contaminants to enter the bloodstream. This is in addition to the inherent injury associated with puncture of the human skin/flesh.

That is not to say that blunt collisions are not dangerous. Blunt collisions can cause whiplash or break bones. Haddadin further classifies blunt collisions as clamped and unclamped collisions. Clamped collisions are those in which the object with which the robot collides cannot absorb the impact as kinetic energy. Unclamped, then, are situations in which the object is allowed to move as a result of collision. Unclamped collisions may include a human standing in open space and being hit in the chest or head.

Clamped collisions could include a human standing against a wall or a collision with an arm in a gloveport, hand and elbow resting on the glovebox.

6.1.1. SHARP/PENETRATING COLLISIONS

Haddadin et al. [“Soft Tissue...”, 2010] did a study of collisions involving sharp tools. Two types of tests were performed: stabbing and cutting. Several different sharp implements, a scalpel, kitchen knife, scissors, steak knife, and a screwdriver, were used to stab and cut a pig leg. The extent of the damage was measured in penetration depth and length of incision. Stabbing and cutting were all tested with joint torque collision detection and three reaction strategies; don’t react, stop immediately, and set joints to gravity compensation only.

Sharp objects in the glovebox present great hazards, whether handled by humans or robots. The protective gloves and the human skin are important barriers and are easily punctured by sharp objects. Therefore in practice, the use of sharp objects in a glovebox is minimized. Also, sharp objects that are used in a glovebox are safely stored or covered when not actively being used. In the near term, the most likely application of robotics in gloveboxes is material handling. Therefore, collisions with sharp objects are unlikely and will not be studied as part of this work.

6.1.2. BLUNT COLLISIONS

6.1.2.1. Unclamped collisions

Haddadin, et al. [“The Role of ...Part 1” 2008] found that for collisions with the unclamped head, the response could not be fast enough to affect the injury to the human. The relevant parameters were robot mass and velocity. The Head Injury Criteria was also calculated for EEF-head collisions for robots of other masses. For all masses, it was

found that robots moving up to 2.5 m/s could not impact the unclamped head hard enough to cause an unacceptable probability of injury.

6.1.2.2. Clamped collisions

Clamped, blunt collisions have been investigated by Haddadin et al. [“The Role of ... Part 2” 2008] using several industrial manipulators and the Deutsches Zentrum fuer Luft- und Raumfahrt (DLR) Lightweight Robot (LWR-III) research robot. Haddadin studies collisions to the head and chest of a crash test dummy. The conclusions compare the effectiveness of collision detection algorithms on robots of different mass. In the case of the DLR LWR-III, the collision detection was unnecessary because the robot reached low-level actuator limits, halting the robot, before causing injury.

In the glovebox, the manipulator will be moving at low velocities so the impact associated with unclamped collisions is not significant. Instead, this work is concerned with clamped blunt collisions. Most research on blunt collisions, and particularly blunt clamped collisions, [Haddadin et al., “The Role of ... Part 2”, 2008] has focused on the head or the chest. For glovebox applications, the head and chest are outside of the robot’s workspace and protected by the glovebox steel and glass. Collisions with hands and arms are studied here.

6.1.3. HUMAN COLLISION RESPONSE

The German agency Deutsche Gesetzliche Unfallversicherung (BGIA) issued a report that studies humans and robots working in collaborative spaces. The report presents some basic concepts of human-robot collisions and gives data for human response.

Body part	Clamping Force	Impact Force	Stress	Compression Constant
	[N]	[N]	[N/cm ²]	[N/mm]
Cranium/Forehead	130	175	30	150
Face	65	90	20	75
Neck (side)	145	190	50	50
Neck (Front/throat)	35	35	10	10
Back/Shoulders	210	250	70	35
Chest	140	210	45	25
Stomach	110	160	35	10
Pelvis	180	250	75	25
Bottom	210	250	80	15
Upper arm/Elbow	150	190	50	30
Lower arm/Wrist	160	220	50	40
Hand/Finger	135	180	60	75
Thigh/Knee	220	250	80	50
Lower Leg	140	170	45	60
Foot/Toes/Ankle	125	160	45	75

Table 6.1: Limits for human-robot collisions [translated from German by the author. source: BGIA 2011]

“The observance of the limits of both injury criteria ensures that the injury severity of the local strain on a particular body part is tolerable.” [BGIA 2011, translated by author] If the BGIA limits are exceeded, there is a risk of an intolerable injury. When in a clamping collision, the clamping force is the maximum allowable force on the identified body part. The clamping force is the maximum allowable for a long duration clamping collision. The impact force is the maximum allowable force for an impact collision, i.e. unclamped collisions. The maximum allowable stress is the force divided by the contact area. The stress and clamping force are inherently related. The overriding limit changes with the contact area. Figure 6.1 shows the Cranium/Forehead clamping force (green line) and stress (purple line) limits versus the contact area. For a collision to be acceptable, it has to be less than the minimum of the two limits. Combinations of

force and contact area below the red line are in the acceptable range. Those above the red line risk intolerable injury.

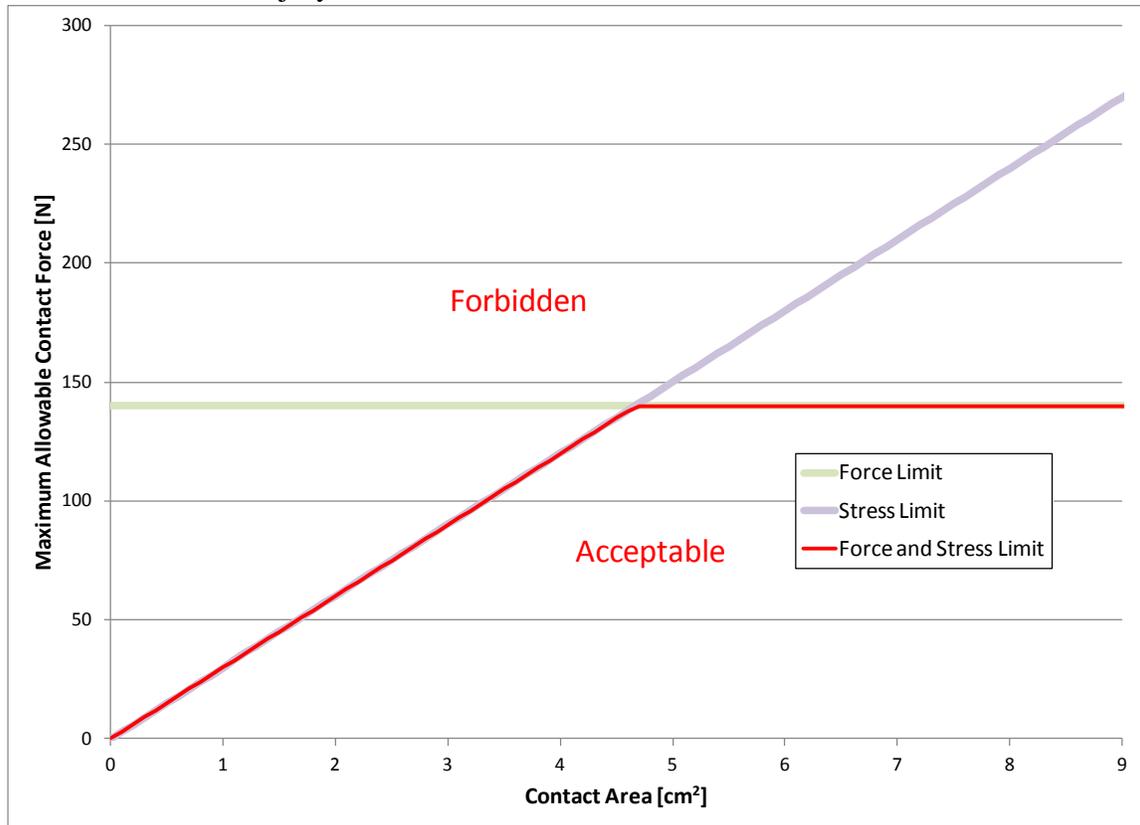


Figure 6.1: Cranium/Forehead force and stress limits

For small contact areas, the stress will dominate the limits. For example, with a contact area of 1 cm², a force of 40 N on the forehead exceeds the maximum allowable stress but not the force. For large contact areas, the clamping force limit dominates. If the contact area is 5 cm², a force of 140 N would exceed the maximum clamping force but the stress would only be 28 N/cm².

The compression constant is the relationship between the compression distance and the contact force. The report models the response force, F , as a linear spring.

$$F = k_{CC}x \quad (6.1)$$

The compression constant of the particular body part, k_{cc} , is multiplied by the compression distance, x . If the robot compresses the cranium 1mm, the force between the robot and the head will be 150 N.

These data are used to simulate collisions and study the effectiveness of collision detection with the SIA5D system. We demonstrate a means for simulating collisions given specific conditions (configuration, human body part, etc.) in order to determine if the proposed model and collision detection algorithm can meet these limits if an accident occurs in select operation scenarios.

6.2. Human-Robot Collision Simulations

Collisions with the human body are simulated using the simulator from the previous chapter. Human body data from Table 6.1 are used for spring constants and acceptable limits. This work is focused on manipulators in a glovebox, so focus will be given to the hand/finger, the lower arm/wrist, and the upper arm/elbow. For the human body, the pressure factor must be considered as well. This is calculated by dividing the maximum contact force by a reasonable or expected contact area. It should be emphasized that this work focuses on blunt collisions so the contact areas are not expected to be very small. For perspective, the face of an American 1 cent coin (penny) has a surface area of 2.85cm^2 . Motions are tested at different speeds in different directions for a few configurations.

6.2.1. STANDARD CONFIGURATION, DOWNWARD CLAMP

First the collisions will be simulated using the control parameters from the previous chapter. The configuration of the manipulator is the same as in Figure 6.2, which is very nearly the same as the experimental configuration.

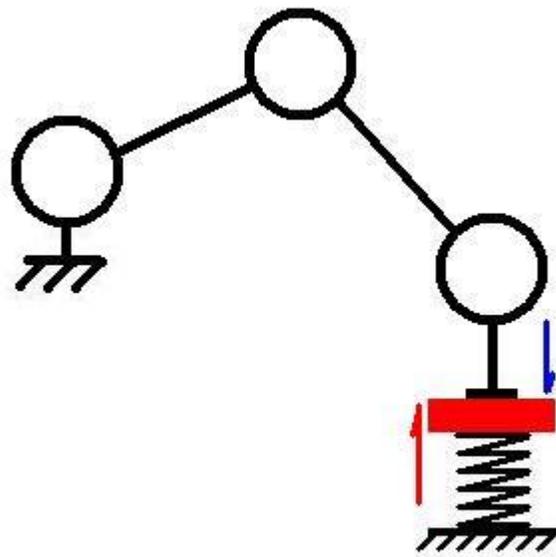


Figure 6.2: Standard configuration downward clamping motion

Results of simulated collisions with the human hand are shown in Figure 6.3.

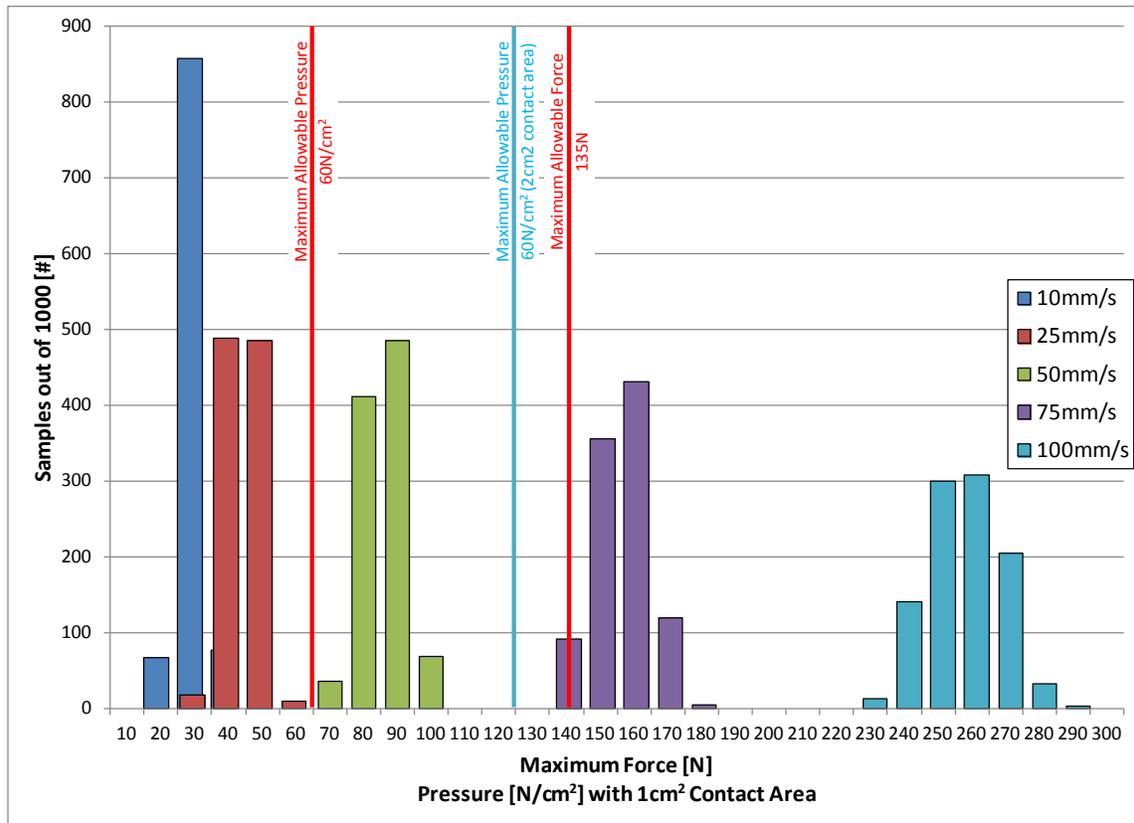


Figure 6.3: Simulated collisions with human hand at different EEf velocities

The red lines indicate the maximum compression force and pressure according to the BGIA data. The pressure changes with the surface area of the contact. The pressure in red is calculated assuming a 1cm^2 contact area. The cyan line indicates the pressure assuming a 2cm^2 contact area. For comparison, the EEf used in the experimental section is 17.3cm^2 . With such a large contact area, the limiting factor for all of the simulations is the force, not the stress. At speeds of 75mm/s and 100mm/s , the simulated collisions exceeded the maximum clamping force. The contact area makes a significant difference in which velocities are acceptable in this configuration. When doubled, the 50mm/s motions are within acceptable bounds. If the contact area were halved, to only 0.5cm^2 , even the 25mm/s collisions would exceed the maximum pressure.

The same collisions are simulated against the lower arm/wrist area in Figure 6.4.

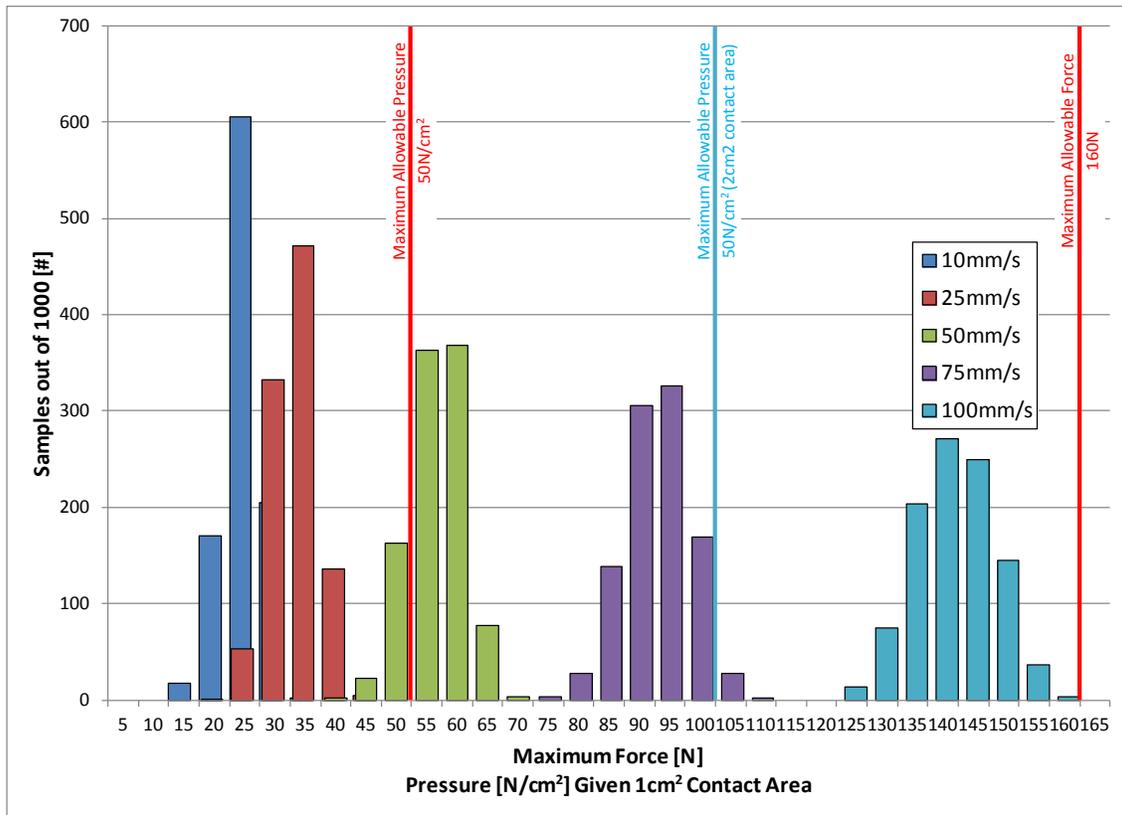


Figure 6.4: Simulated collisions with human lower arm/wrist at different EEF velocities

For all tested speeds, the force is less than the Maximum Allowable Force. Given a contact area of 1cm^2 , the tests at 10mm/s and 25mm/s indicate the pressure would not exceed the allowable maximum. If the contact area is doubled to just 2cm^2 , all tests at 50mm/s and most at 75mm/s generate less than the maximum allowable pressure. Only 29 of the tests at 75mm/s exceeded the pressure. This implies that, given the 2cm^2 contact area, collisions at 75mm/s for the given configuration would result in injury to the lower arm/wrist approximately 3% of the time.

Collisions against the upper arm are shown in Figure 6.5.

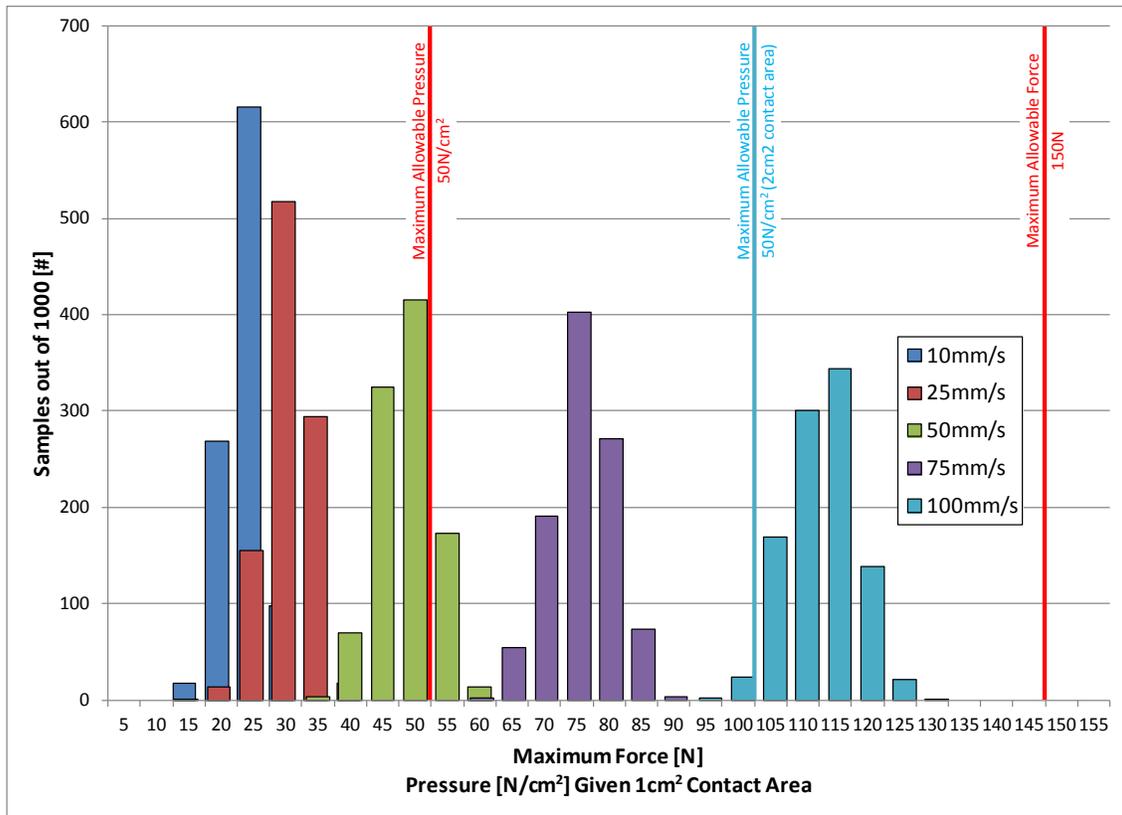


Figure 6.5: Simulated collisions with human upper arm/elbow at different EEF velocities

As with the lower arm, simulations at both 10mm/s and 25mm/s indicate results well below thresholds of force and pressure (for both 1cm² and 2cm² contact areas). Again, doubling the contact area indicates an increase in the allowable end effector velocity. For the upper arm, all collisions at 75mm/s indicate effective detection and response.

With low speeds, simulated motions in the downward direction in this configuration do not exceed the BGIA maximum forces for the hand and arm. If the EEF moves in a different direction, the generation of torques in the joints will be different. The joint that triggers a collision may be different and may require a larger contact force. The downward motion triggers collisions in joint 2, which has a large moment arm. End effector motions parallel to joint 2 with the same initial configuration are examined next.

6.2.2. STANDARD CONFIGURATION, SIDEWAYS CLAMP

The manipulator is tested in the same configuration. Now the EEF is moved in a direction perpendicular to the page (Figure 6.2). When the EEF hits the simulated object the force is generated parallel to joint 2. Instead of triggering joint 2, as in the previous examples, the torque in joints 1 and/or 3 exceed the threshold values (Figure 6.6).

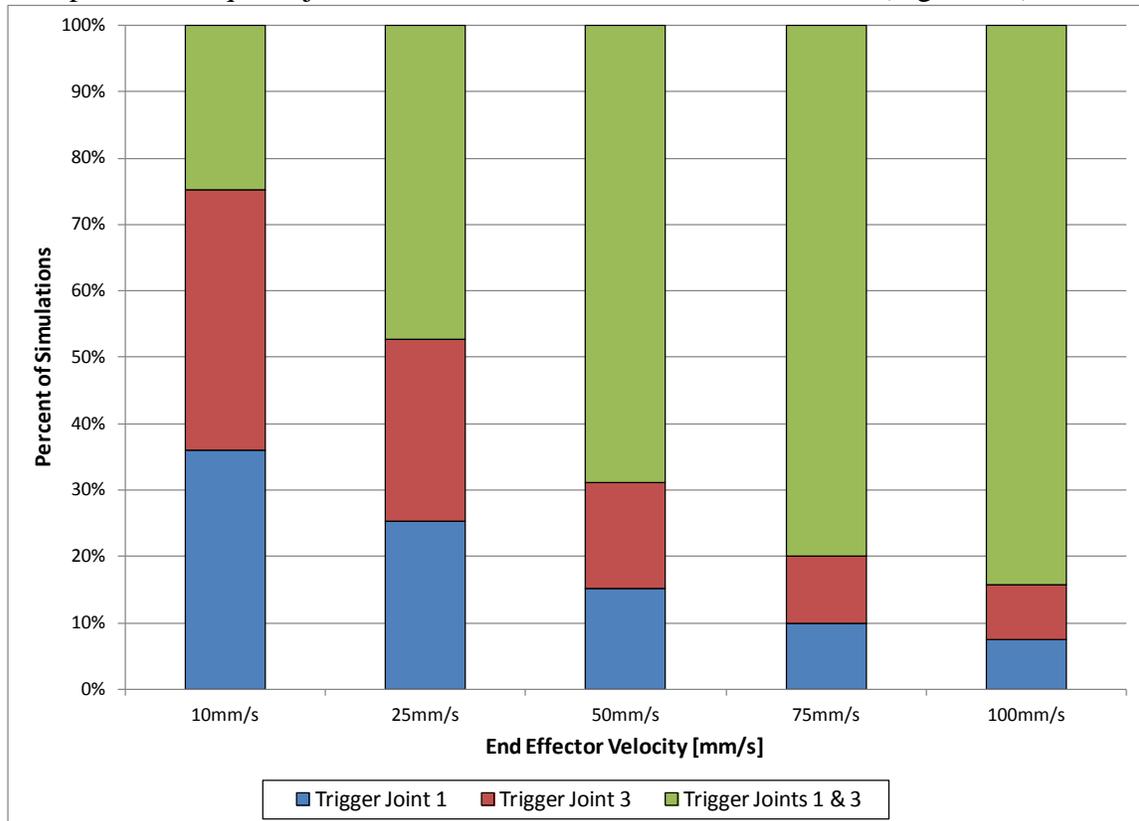


Figure 6.6: Trigger joints for lateral hand motions

Collisions are detected by joints 1 and 3 equally (blue and red bars). The collision is detected by both joints simultaneously less than 1/3 of the time at 10mm/s. As the velocity increases, the torque exceeds the threshold in both joints more frequently. With an increasing velocity, the torque change between feedback measurements increases so it becomes more likely that the collision will be detected by both joints.

The results at different velocities are similar to those for the downward clamping motion. At 10mm/s, all 10,000 simulations are below the maximum allowable force. The maximum pressure with a 2cm² contact area is also below the maximum allowable pressure.

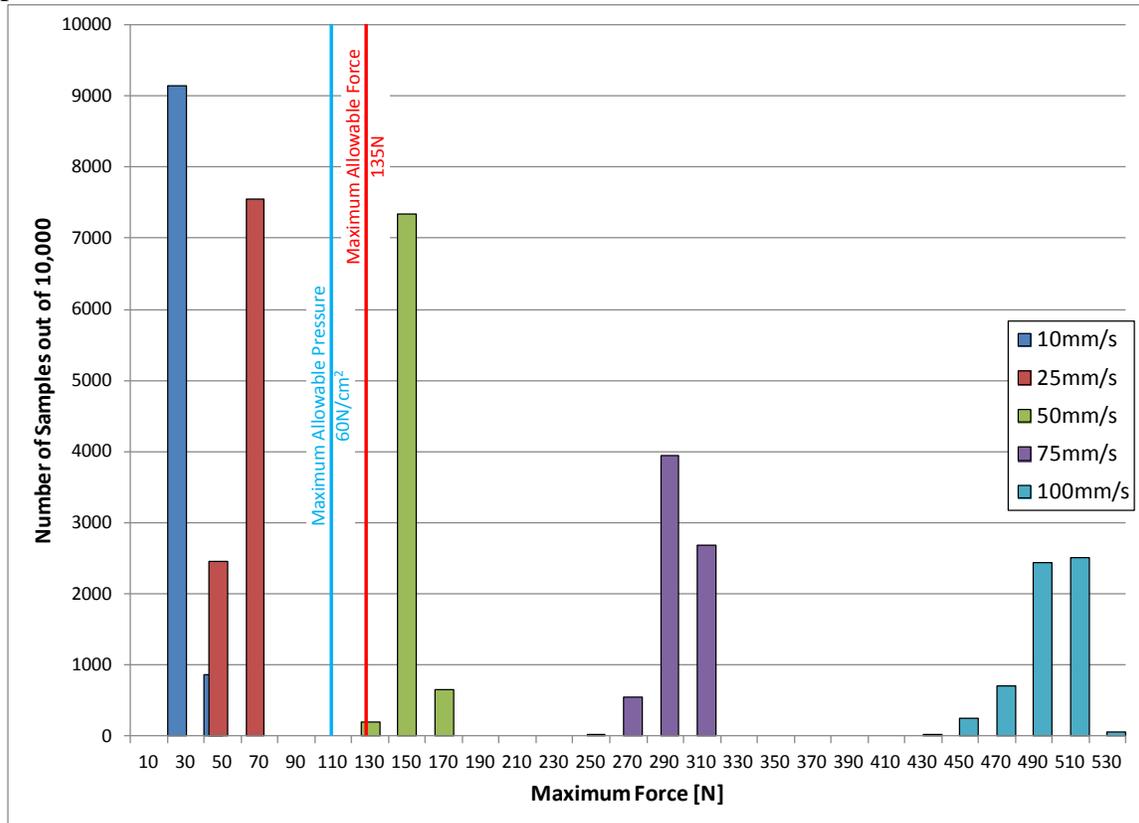


Figure 6.7: Results of simulated collisions with hand during lateral EEF motions

Similar to the downward motion, at velocities of 25mm/s the maximum force is below the acceptable level. With a contact area of 2cm², the pressure is also below the maximum allowable level.

6.2.3. RANDOM CONFIGURATIONS AT 10MM/S EEF VELOCITY

Given one configuration and two directions of motion for collisions, the demonstrated system would be expected to safely detect and respond to collisions at velocities of 10mm/s and 25mm/s. But this is only one configuration and two directions

of motion in the entire workspace. To get some idea of the level of safety in the entire workspace, the simulator is run 61,000 times with random initial joint positions and a random EEf velocity. Each initial joint position is evenly distributed within the hardware joint range (Table 6.2).

Table 6.2: Range of random joint positions

	Minimum [degrees]	Maximum [degrees]
Joint 1	-180	180
Joint 2	-110	110
Joint 3	-170	170
Joint 4	0	205
Joint 5	-180	180
Joint 6	-110	110
Joint 7	0 ^[1]	

^[1]Joint 7 does not affect results

The end effector velocity has a normally distributed magnitude with mean 10mm/s and a standard deviation of 2mm/s (Figure 6.8).

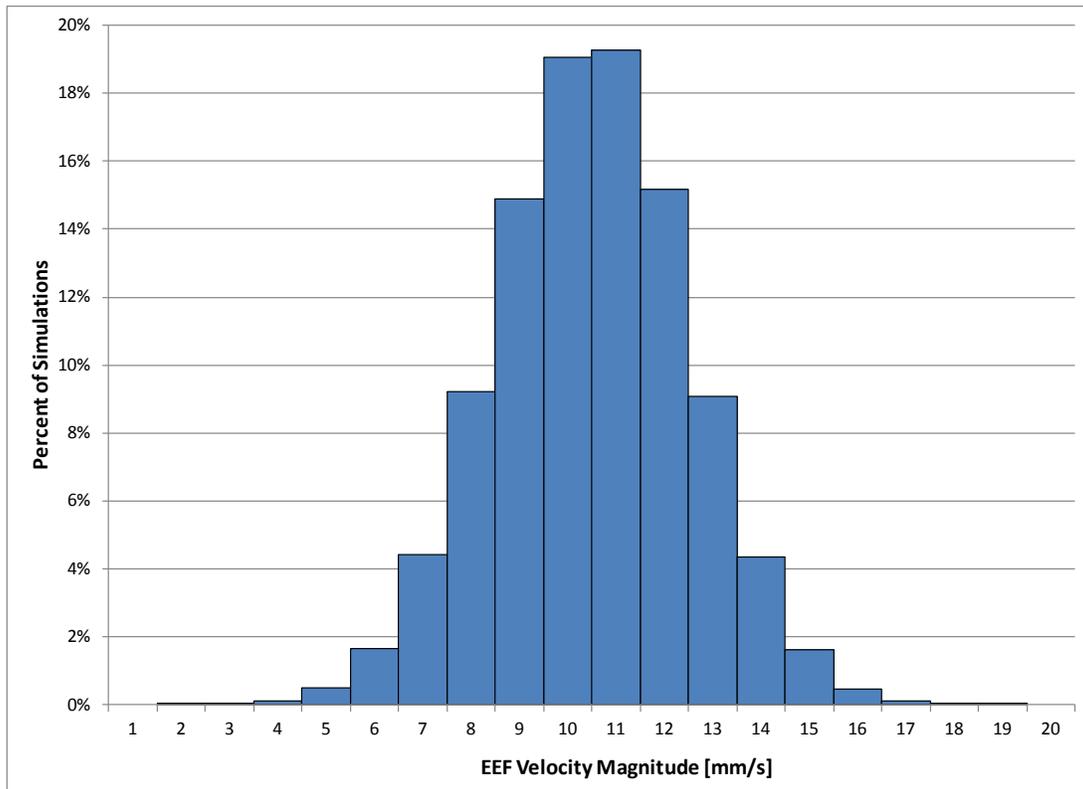


Figure 6.8: EEf velocity distribution

The angle of the velocity is varied by two evenly distributed angles. The first angle is measured from the Z-axis about the X-axis and varies between 0° and 180° . The second angle is measured about the Z-axis and varies between -180° and 180° . Combining these angles gives an even distribution in all directions from a given point. Collisions are simulated with the human hand.

The maximum force distribution for these simulations is shown in Figure 6.9.

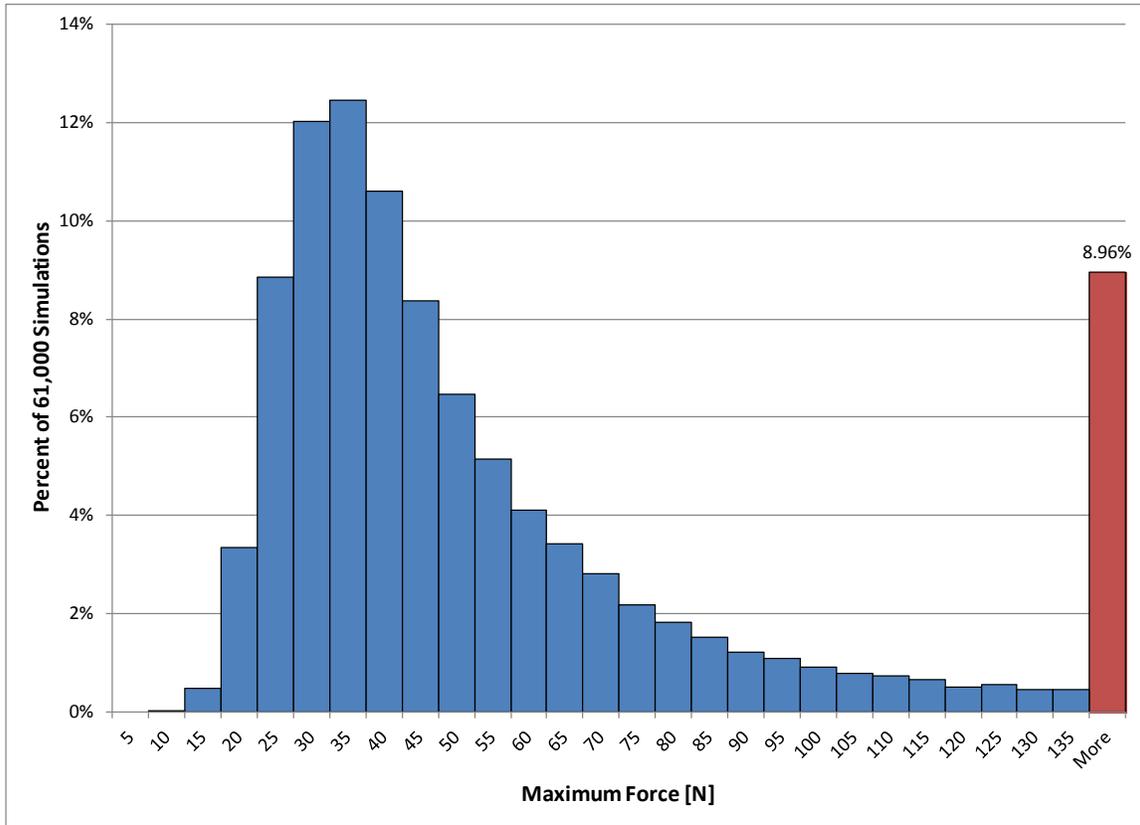


Figure 6.9: Maximum force distribution with random positions and velocities with hand collisions

Less than 9% of the positions and velocities resulted in collisions exceeding the maximum allowable force. Most of the positions and velocities resulted in collision forces around 30-40N.

To get an idea of where the excessive forces occur, the workspace was sliced along planes parallel to the X-Z axis. The end effector position at detection was plotted in each of the six slices. The first graph shows all the end effector detection locations for a Y coordinate less than 0.5m. The origin is centered at the intersection of the axes of the first and second joints, i.e. at the shoulder.

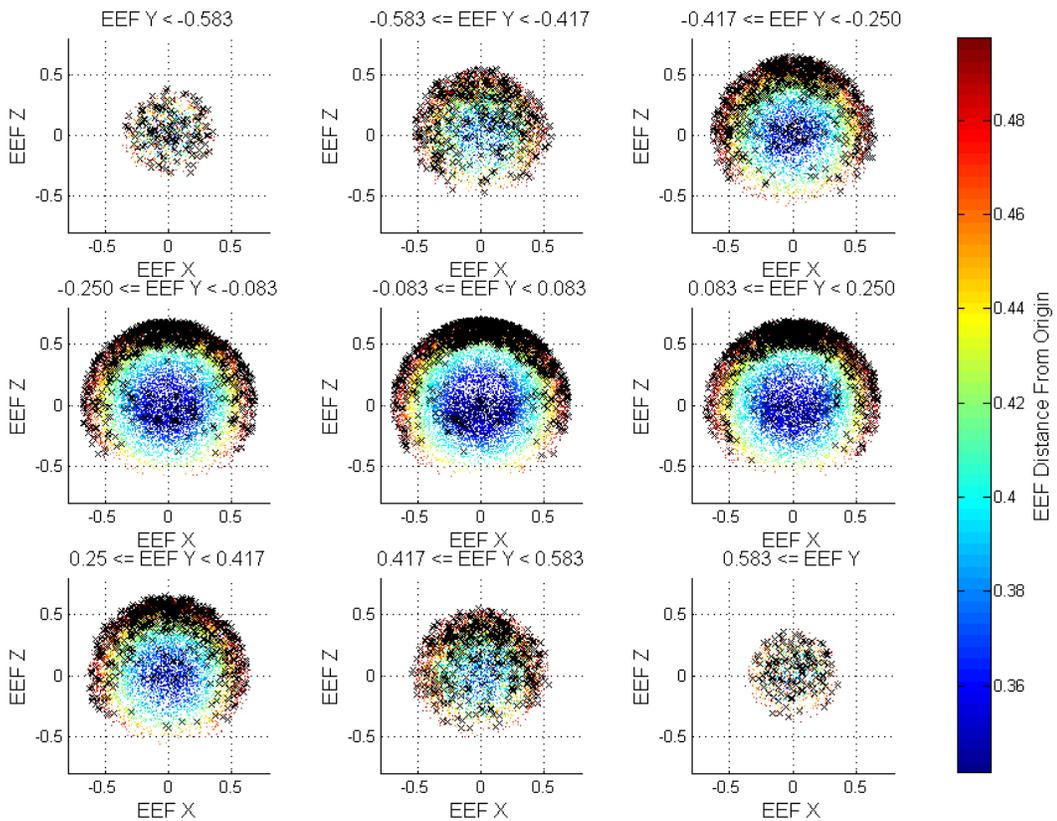


Figure 6.10: EEF locations at detection (black X indicates force exceeded acceptable level, color of dot is distance from origin for acceptable-force collisions)

For collisions where the force exceeded the acceptable level (135N), a black x was used. Where the force remained below the acceptable level a colored dot was used. The color of the dot indicates the distance from the origin to the end effector. As might be expected, most of the points where the force exceeds the safe level are at the edges of the workspace, where the manipulator is completely extended (similar to Figure 6.11).

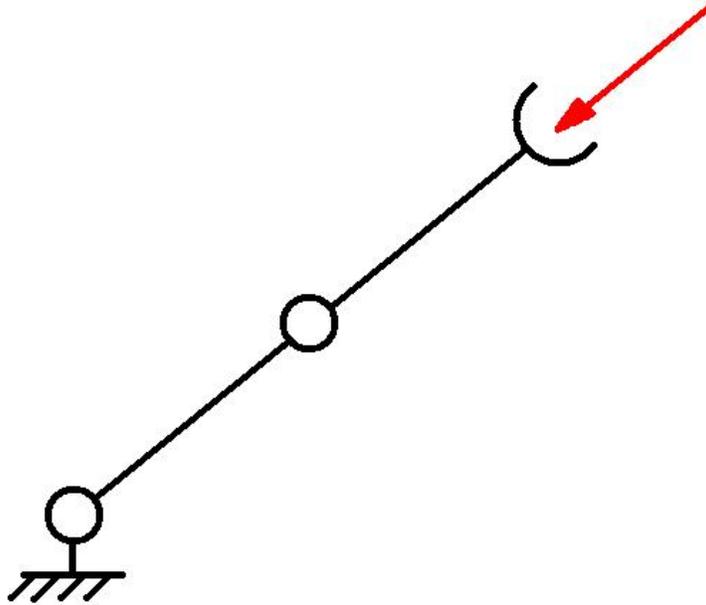


Figure 6.11: 2 DOF planar robot at a singularity

Configurations where the manipulator would collide with itself have not been excluded from this data set but would be excluded by more advanced motion planning systems. Such systems are also likely to avoid the kinematic singularities that lead to high or undetectable forces..

6.2.4. SENSITIVITY PREDICTION

As discussed in 4.4.2, there are some configurations that are more insensitive to collisions. When the condition number approaches infinity, the Jacobian transpose is approaching a singularity. A high condition number does not guarantee a high collision force. (Figure 6.12) The singularity is direction dependent so the manipulator may still remain sensitive in other directions.

Maximum Force [N]	Log Maximum Force	Condition Number																																							
		1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2	2.1	2.2	2.3	2.4	2.5	2.6	2.7	2.8	2.9	3	3.1	3.2	3.3	3.4	3.5	3.6	3.7	3.8	3.9	4	More								
<=19.95	1.30	0	0	6	13	38	29	22	26	17	11	13	12	8	12	10	7	6	5	4	3	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			
<=21.88	1.34	0	2	19	43	61	48	39	21	18	12	18	12	8	7	4	3	4	3	3	2	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1				
<=23.99	1.38	0	19	58	107	115	69	61	38	32	35	19	12	8	7	9	3	2	1	3	1	1	2	0	1	1	1	1	1	1	1	1	1	1	1	1	1				
<=26.30	1.42	4	64	104	130	156	128	79	51	46	41	24	18	17	8	7	5	3	3	3	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1				
<=28.84	1.46	8	135	183	243	199	153	100	64	60	44	26	19	18	12	17	6	9	2	1	0	2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1				
<=31.62	1.50	18	222	261	262	243	188	124	78	67	54	41	28	20	16	3	8	4	2	3	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1				
<=34.67	1.54	42	309	311	319	240	198	131	97	80	63	32	27	15	11	7	11	7	5	5	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1				
<=38.02	1.58	44	307	291	256	228	185	148	101	86	54	40	29	25	15	4	4	3	3	4	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1			
<=41.69	1.62	64	315	297	253	232	195	131	100	75	56	50	29	19	26	12	9	5	3	1	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
<=45.71	1.66	48	269	285	207	186	184	134	94	83	73	57	37	28	13	10	6	6	1	0	4	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
<=50.12	1.70	57	260	212	205	195	157	115	79	75	70	43	24	34	16	6	10	5	3	4	3	2	1	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
<=54.95	1.74	66	211	183	185	155	129	97	111	71	62	49	45	29	22	14	6	7	3	2	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
<=60.26	1.78	43	161	164	171	130	125	87	86	74	64	54	38	23	15	12	13	7	3	1	3	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0			
<=66.07	1.82	55	139	125	113	117	96	95	72	60	39	42	28	28	30	10	9	8	5	1	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
<=72.44	1.86	41	104	102	107	81	85	74	70	76	43	38	34	27	19	12	6	1	6	1	1	1	1	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0		
<=79.43	1.90	29	93	86	96	91	86	70	58	71	47	30	38	23	21	8	11	6	5	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
<=87.10	1.94	29	63	87	79	70	79	65	69	43	44	38	34	17	19	8	5	5	4	5	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
<=95.50	1.98	7	37	50	51	70	58	60	64	44	34	49	30	21	10	14	10	4	6	2	1	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0		
<=104.71	2.02	3	23	33	52	46	39	54	39	57	28	35	17	25	26	7	3	5	2	1	2	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0		
<=114.82	2.06	1	11	33	27	32	42	57	48	35	34	28	23	21	12	13	7	11	3	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
<=125.89	2.10	0	12	17	21	21	24	48	37	30	37	22	18	17	19	7	10	3	1	0	1	0	2	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0		
<=138.04	2.14	0	6	10	13	18	18	36	37	28	24	33	16	25	12	12	7	2	0	3	3	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
<=151.36	2.18	1	7	3	11	21	18	29	27	26	16	16	17	19	13	9	6	1	5	4	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0		
<=165.96	2.22	0	1	8	8	5	15	28	20	16	14	23	17	13	14	8	9	3	2	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1			
<=181.97	2.26	1	3	3	4	7	13	22	23	30	12	23	17	14	17	4	6	4	2	2	2	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1		
<=199.53	2.30	0	2	1	4	9	12	7	24	19	17	14	8	14	14	6	9	3	1	2	1	1	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0		
<=218.78	2.34	0	3	1	5	2	4	12	9	13	14	9	13	15	10	8	7	7	2	3	2	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1		
<=239.88	2.38	0	2	0	2	2	4	10	9	10	5	9	9	14	11	5	5	5	4	2	2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1		
<=263.03	2.42	0	2	3	1	2	5	5	12	5	6	15	6	11	9	12	5	2	4	1	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0		
<=288.40	2.46	0	0	1	1	2	0	5	3	6	8	12	9	13	6	7	5	5	5	3	4	3	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
More		0	1	4	5	2	12	24	33	30	25	42	74	96	107	101	80	90	75	61	58	31	36	28	20	22	10	14	6	6	9	3	17								

Figure 6.12: Maximum force versus condition number frequency map. Color and cell number indicate the frequency out of 25,000 simulations. Red are most frequent, green are least.

The majority of high force collisions happen in configurations with higher condition number. Several low force collisions also happen at the higher condition numbers.

If the matrix is not singular, the sensitivity becomes considerably harder to predict from the condition number or the matrix decomposition because all three matrices of the decomposition will affect the visibility of a force in the joint torques. However, the minimum singular value does demonstrate a relationship to the maximum force. If the minimum singular value is very small, we would expect less of the force to be visible as joint torque and thus a higher force is required for detection.

To test the effect of the minimum singular value on the maximum force, the frequency of simulations is plotted against the maximum force and the minimum singular value. (Figure 6.13)

Force [N]	Log ₁₀ Max Force	Minimum Singular Value																				
		0.005	0.01	0.015	0.02	0.025	0.03	0.035	0.04	0.045	0.05	0.055	0.06	0.065	0.07	0.075	0.08	0.085	0.09	0.095	0.1	>0.1
<=19.95	<=1.30	26	31	11	16	16	8	10	13	14	9	13	8	14	6	8	12	10	11	0	5	14
<=21.88	<=1.34	11	11	9	15	14	9	16	7	13	13	17	14	9	25	14	14	22	17	13	16	40
<=23.99	<=1.38	11	14	17	17	23	29	21	22	18	21	24	23	29	21	30	33	35	24	34	24	135
<=26.30	<=1.42	12	16	27	17	37	30	28	33	25	34	32	37	41	50	36	55	40	39	34	43	231
<=28.84	<=1.46	11	31	31	21	38	34	39	35	35	36	44	55	51	50	51	59	54	48	67	73	440
<=31.62	<=1.50	9	20	31	37	54	46	31	48	44	45	49	63	59	71	59	57	73	69	75	59	646
<=34.67	<=1.54	16	28	27	33	44	50	52	46	55	48	57	57	78	66	74	71	58	63	82	81	829
<=38.02	<=1.58	16	13	40	35	54	42	57	52	57	51	60	70	65	57	76	55	73	58	60	46	798
<=41.69	<=1.62	8	35	37	40	52	52	53	49	57	47	53	64	60	73	71	64	52	64	74	63	808
<=45.71	<=1.66	9	26	39	59	54	63	60	38	55	50	68	48	63	65	59	55	48	55	46	54	713
<=50.12	<=1.70	18	24	47	38	48	55	48	53	33	45	65	43	52	51	61	49	54	54	46	53	642
<=54.95	<=1.74	11	35	47	58	44	58	45	52	56	50	45	37	46	37	60	35	47	39	42	37	570
<=60.26	<=1.78	10	33	40	53	55	53	44	53	46	35	40	33	49	47	39	33	45	30	36	40	463
<=66.07	<=1.82	15	32	50	37	45	36	35	42	41	27	42	47	37	30	28	37	27	33	25	26	385
<=72.44	<=1.86	12	21	50	36	43	42	48	41	39	31	31	33	34	21	30	20	25	23	22	27	303
<=79.43	<=1.90	13	32	35	49	34	41	41	40	26	31	33	32	25	31	35	16	28	25	21	25	261
<=87.10	<=1.94	12	25	34	37	40	38	34	29	39	26	26	28	33	27	19	15	16	24	27	17	219
<=95.50	<=1.98	15	27	36	43	45	26	26	40	31	28	26	21	25	11	20	19	21	21	12	18	114
<=104.71	<=2.02	9	22	45	22	37	26	40	27	19	28	22	11	14	13	17	11	10	17	12	9	89
<=114.82	<=2.06	10	31	32	34	25	30	26	17	31	29	19	23	16	12	12	5	8	10	11	4	58
<=125.89	<=2.10	8	23	31	26	30	27	18	18	20	30	16	11	13	6	6	5	6	6	9	5	35
<=138.04	<=2.14	14	25	35	32	27	16	19	21	19	14	17	13	3	7	5	7	4	3	5	3	24
<=151.36	<=2.18	14	20	29	22	17	10	22	12	19	12	12	6	9	6	5	3	3	7	9	1	16
<=165.96	<=2.22	12	19	28	23	19	9	11	15	7	15	11	5	7	7	2	2	1	1	1	3	13
<=181.97	<=2.26	10	21	26	21	22	8	20	14	15	11	13	5	4	5	2	3	2	0	1	1	9
<=199.53	<=2.30	9	22	26	11	15	14	14	9	5	5	3	3	2	4	2	2	1	0	1	0	6
<=218.78	<=2.34	12	23	22	20	7	13	5	9	3	9	3	1	3	1	2	1	0	1	1	0	7
<=239.88	<=2.38	15	17	20	12	8	2	5	9	4	7	3	1	2	2	0	1	1	0	0	0	4
<=263.03	<=2.42	9	21	19	8	13	4	5	4	9	4	0	3	2	0	2	0	0	1	0	1	5
<=288.40	<=2.46	20	16	18	12	11	6	4	4	0	1	2	1	1	0	0	1	0	1	0	0	2
More than 288.40N		424	282	180	77	31	20	25	15	17	19	6	7	3	3	2	0	1	0	0	0	10

Figure 6.13: Maximum force versus minimum singular value frequency map

In Figure 6.13, most of the high force simulations (bottom) had a low minimum singular value (left) and most simulations with higher minimum singular value (right) resulted in a lesser maximum force (top). Like the condition number, the minimum singular value cannot predict the sensitivity of the collision detection.

It is impossible to design a collision detection system which can detect any and all collisions and respond such that forces remain below acceptable levels. However, the condition number or the minimum singular value can be used to identify or avoid configurations where forces are likely to exceed acceptable levels. An implemented

collision detection system may either force the manipulator to avoid those configurations or it may notify the operator that the system is susceptible to dangerous collision forces.

For collision detection implementation, the condition number can be calculated online to indicate when the manipulator is approaching a configuration with directions of insensitivity. Conveniently, these configurations correspond to configurations which are generally avoided for kinematic reasons, e.g. singularities. Avoiding these configurations is also used as a criterion for redundancy resolution in Pryor [2002].

6.3. Chapter Summary

Collisions with the human hand and arm were simulated in this chapter. The results indicate that at low velocities, there is a high probability of effective detection and response. When moving in a downward motion, typical of a pick and place motion, the simulations resulted in safe detection and reaction for velocities of 10mm/s and 25mm/s. at higher velocities the force exceeds the acceptable limit for some of the hand. If the contact area is small, the pressure will be too much.

Simulations were performed for random joint positions and EEF velocities. Some configurations of the manipulator are not sensitive to collisions and would likely result in unsafe collision forces and pressures. However, these configurations are at or near singularities which are frequently avoided for kinematic reasons.

The condition number is used in kinematics to avoid singular configurations. The relationship between condition number and the maximum collision force is analyzed here, too. There appears to be some indication that higher condition numbers result in higher collision forces. However, sometimes, a configuration with a high condition number results in a low force. The minimum condition number was examined as an indicator of maximum collision force. The correlation between the minimum singular

value and the maximum collision force exists, but is also an imperfect indicator of maximum collision force. Some of the collisions with low minimum singular values have low maximum forces.

Parameters such as the model accuracy, maximum acceleration, or the system feedback rate may be improved in the future. Improving these parameters may permit a higher operating velocity yielding similar maximum force distributions. It is impossible, however, to completely eliminate the insensitivities associated with singular configurations.

7. CONCLUSIONS AND FUTURE WORK

Effective collision detection and response is critical as robot application areas expand into human-shared spaces. Some research focuses on implementation using cutting edge hardware and techniques. The goals of this work were to demonstrate collision detection on a position controlled, current feedback industrial manipulator and to evaluate the effectiveness of the system. To achieve these goals, a black box model for torque estimation was developed, collision detection using the estimated torque was demonstrated, and the effectiveness of the collision detection system for human robot interaction was evaluated.

7.1. Summary

A black box model is developed for estimating joint torques from motor currents on a position-controlled industrial manipulator. The model includes velocity dependent and hysteresis losses in the motor and joint. For each joint in the SIA5D serial manipulator, the velocity dependent losses are characterized in terms of the motor current. The current lost is found to be dependent on a power function of the velocity.

$$i_{friction}(\dot{\theta}_{joint}) = sgn(\dot{\theta}_{joint})D|\dot{\theta}_{joint}|^F \quad (7.1)$$

The joint current is the current which is not lost to velocity dependent losses. The joint torque is estimated from the joint current by a second-order polynomial.

$$\tau_{est, \dot{\theta}=0} = f(i - i_{friction}) = -sgn(\dot{\theta}_{joint})Ai_{joint}^2 + Bi_{joint} + C \quad (7.2)$$

If the joint velocity is zero, the hysteresis losses are subtracted from the joint torque. The hysteresis loss is dependent on the joint's last non-zero velocity, $\dot{\theta}_H$.

$$\tau_H = \begin{cases} T_H sgn(\dot{\theta}_H) & |\dot{\theta}| = 0 \\ 0 & |\dot{\theta}| > 0 \end{cases} \quad (7.3)$$

The acceleration dependent inertia term was omitted from this work.

The torque estimation model was validated by comparison to the Newton-Euler predicted joint torques. The manipulator was moved through typical glovebox pick and place motions and semi-random trajectories covering much of the joint position and velocity range. The differences between the estimated and predicted torques were found to have standard deviations of less than 10Nm except joint 2 on the second workspace motion.

Table 7.1: Validation Motion Statistics

	Pick and place		Workspace motion 1		Workspace motion 2	
	Average [Nm]	Std. Dev. [Nm]	Average [Nm]	Std. Dev. [Nm]	Average [Nm]	Std. Dev. [Nm]
Joint 1	4.4391	5.0078	0.6142	9.8146	0.0314	8.1604
Joint 2	0.2607	3.5832	-0.9028	9.6544	-1.9945	12.7070
Joint 3	-5.6133	4.1414	3.0950	6.1881	0.9921	9.0358
Joint 4	1.5958	1.5317	0.1803	8.9397	0.9219	6.3589
Joint 5	-5.6335	1.5256	2.0444	6.3575	-0.3515	5.9359
Joint 6	0.1112	2.8138	0.0000	0.0001	1.1532	6.1437

To gain better understanding of the significance of these values, the model is used to estimate the contact torque – the torque due to a contact force. The estimated contact torque is compared to the torque due a measured contact force. For comparison, the force which would be required to generate the estimated contact torque is compared to the measured contact torque. The difference between these forces is small (less than 5-10N for each test) except during acceleration.

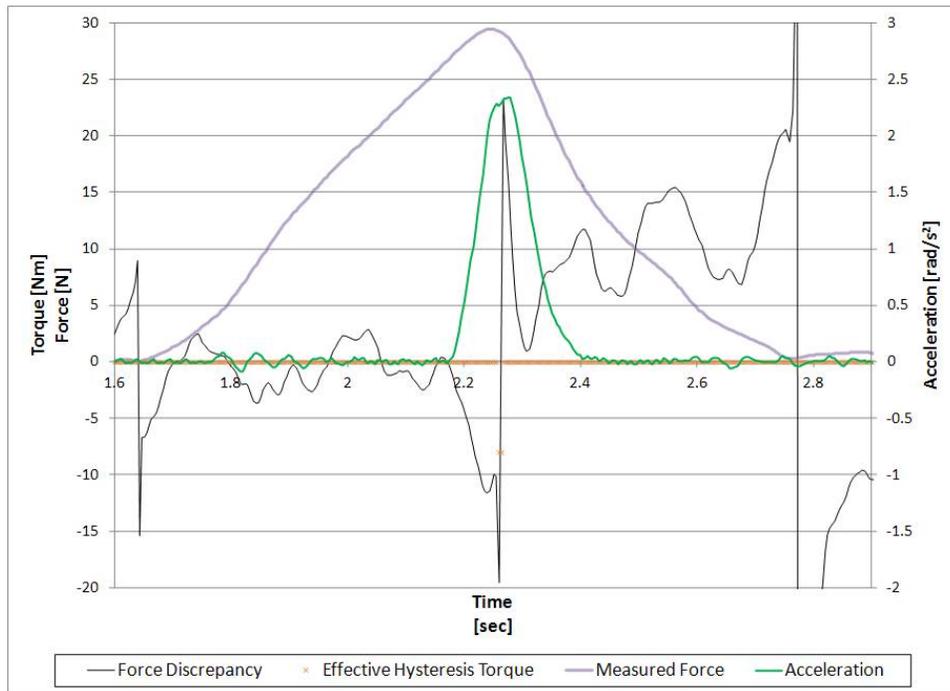


Figure 7.1: Joint 2 force error, hysteresis, and acceleration for 25mm/s EEf velocity on foam

The error increases due to the acceleration and the spike as the velocity changes sign.

The estimated joint torques are used to detect collisions between the manipulator and the environment. The difference between the estimated and predicted torque is monitored to identify when a collision occurs.

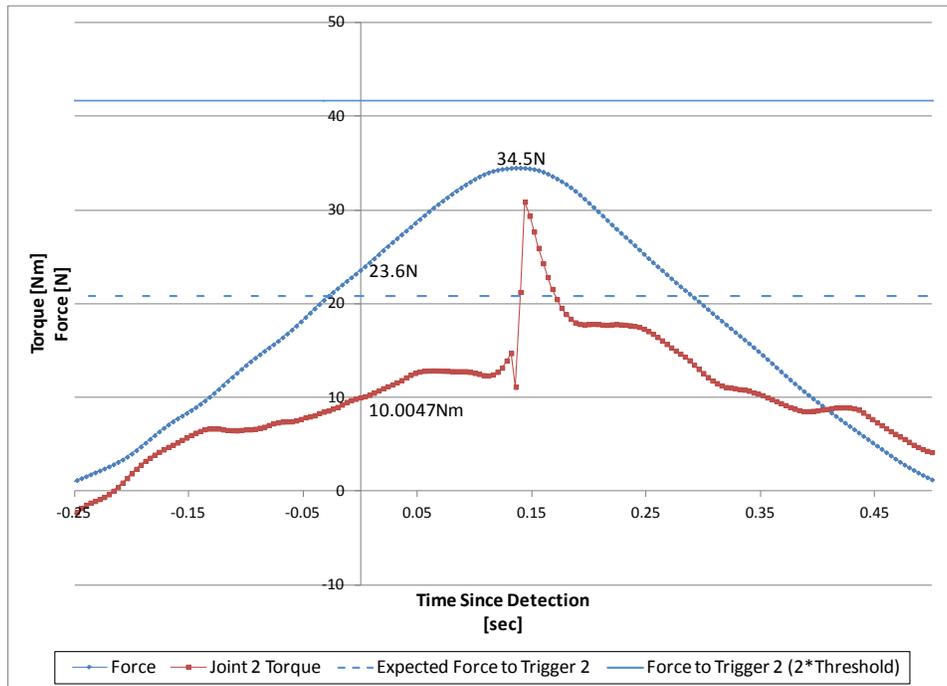


Figure 7.2: Estimated joint 2 torque and measured EEF force at 25mm/s with Plexiglas

Collisions are tested with a variety of objects at different velocities and thresholds. The system detected and responded to collisions with a glovebox glass window and a banana without causing significant damage to either.

Collisions could not be tested directly with a human so a simulator was developed. Characteristics of the manipulator and the collision could be changed to emulate different operating conditions and effects on objects such as a human arm or hand. The simulator is written in C++ using OSCAR and openCV libraries. The user can change parameters and run the simulation hundreds or thousands of times through console text input. Results from the simulations are output to a text file for statistical analysis. The simulator is used to develop a rough design guide which might be used by engineers implementing the collision detection system on another robot or making changes to the system to improve effectiveness. Key parameters of the system are identified and their effect on the maximum collision force is analyzed.

Table 7.2: Relationship between parameters and maximum force

Parameter	Effect of Increasing Parameter	Relationship
Object Stiffness	Increase Force	Linear
EEF Velocity	Increase Force	Quadratic
Detection Thresholds	Increase Force	Linear
Feedback Rate	Increase Force	Power
Maximum Joint Acceleration	Decrease Force	Power

Collisions with the human hand and arm were simulated for pick and place motions. At low end effector velocities, the maximum force remained below acceptable injury limits.

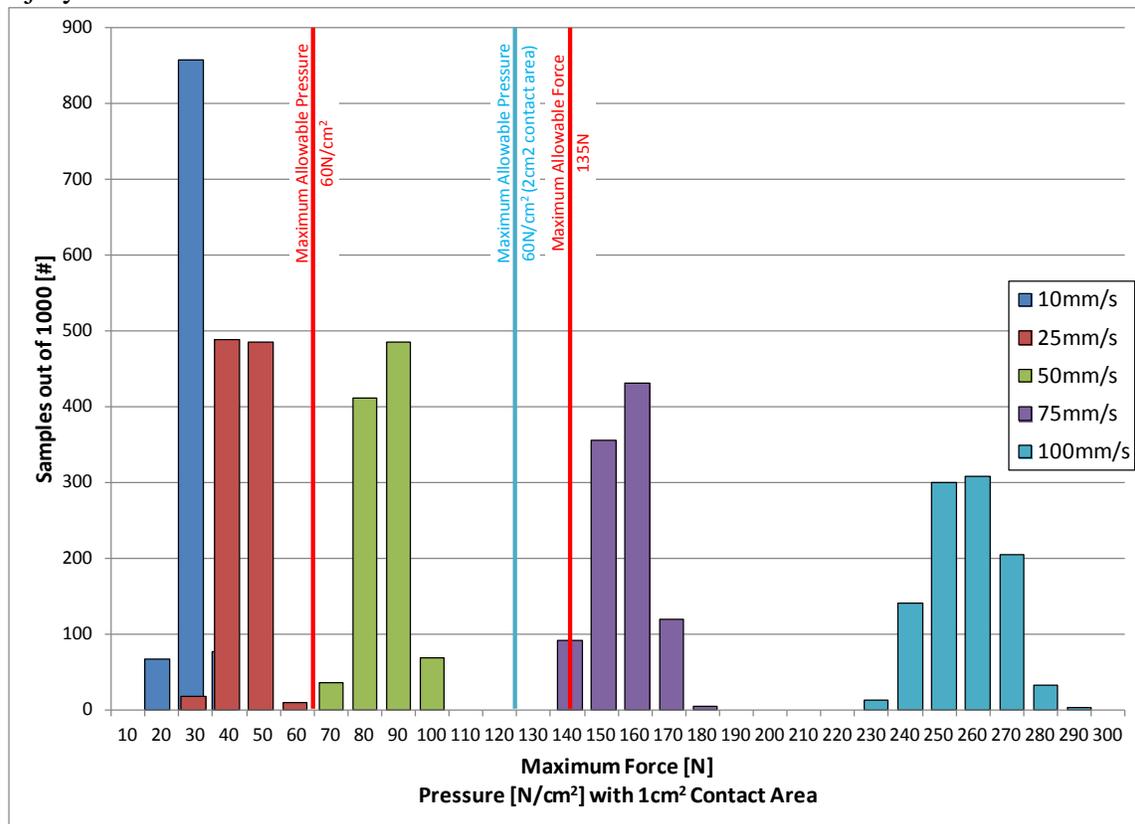


Figure 7.3: Simulated collisions with human hand at different EEF velocities

Stress related limits must be estimated because the size of the contact area cannot be predicted ahead of time. The simulations are also repeated covering the entire workspace of the manipulator. Even at 10mm/s end effector velocity, some collisions result in excessive forces due to limitations of detectability due to configuration.

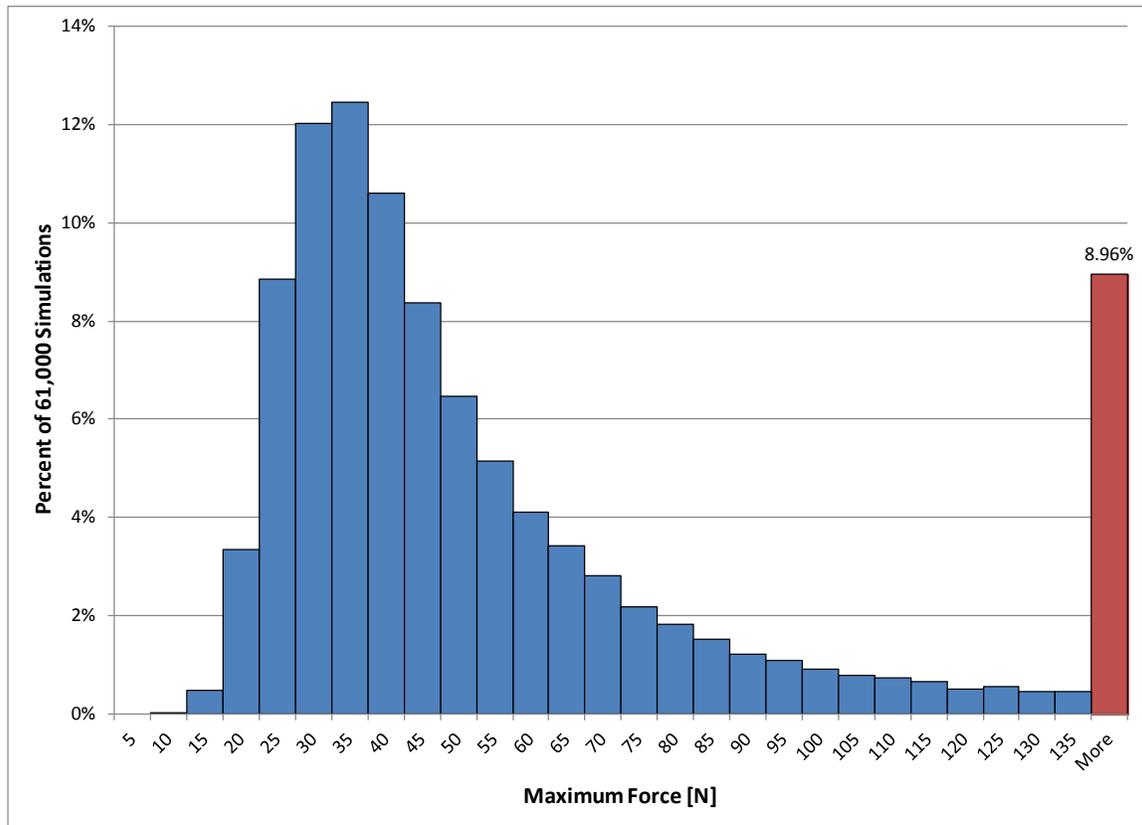


Figure 7.4: Maximum force distribution with random positions and velocities with hand collisions

Simulations provide insight on the safety-related effectiveness of the given manipulator, controller, and collision detection system. In the next section, the objectives outlined in the introduction and the relevant portions of this work are revisited.

7.2. Contributions and Conclusions

The summarized work was performed to meet the objectives outlined in the introduction. The actions and results are highlighted with the objectives they satisfy.

- Map motor “current” measurements to joint torque for an industrial manipulator that is compatible with glovebox manufacturing.

Estimated torques were validated against predicted torques and the contact torque.

- Analytically identify critical points where manipulators are most likely to cause damage.

There are numerous factors, including noise and estimation uncertainty, which influence the likelihood of injury. Condition number is low when the manipulator is insensitive to collisions but not all collisions at low condition number result in high forces.

- Calculate collision response time (time from initial contact until removal of contact) as function of manipulator parameters.

Response time, compression, and force are dependent on the semi-random effects of the estimation error and noise. Simulations capture the response characteristics.

- Model compression and force imparted to relevant body parts during collision.

Response time, compression, and force are dependent on the semi-random effects of the estimation error and noise. Simulations capture the response characteristics.

- Experimentally compare the collision response time, compression distance, and force to model.

Collision simulator was validated against experimental collision data.

- Quantify and/or empirically evaluate system parameters (bandwidth, detection sensitivity, etc.) that impact safety during robot-human collision in a glovebox.

Simulated results demonstrate the effects of changing system parameters. Critical parameters for improving effectiveness are detection threshold and feedback frequency.

It is impossible to guarantee safety in a collision between a robot and human, but simulations over the entire configuration space at low velocities indicate that most collisions can be effectively detected and responded to.

7.3. Future Work

Collision detection was explored through extensive experimentation and simulation. However, it would be useful to develop a means for identifying configurations when the manipulator is insensitive to collisions. In the simulations in this work, if the force was excessive, the condition number of the Jacobian transpose was high and the minimum singular value was low. However these indicators are not perfectly thorough and a more precise means for identifying insensitive configurations would be very useful for real-world implementation.

The goal of this work was to develop an effective model which requires minimal investment for parameter estimation or controller development. Future systems might improve on the proposed system to decrease the detection thresholds and increase the sensitivity. The first parameter to focus on might be the lumped inertia parameter. The lumped inertia is an acceleration related parameter that was not finely tuned for this work. In future works the effects of the lumped inertia might be explored. For example, the acceleration dependent errors plot (Figure 7.5) may show an acceleration dependent hysteresis.

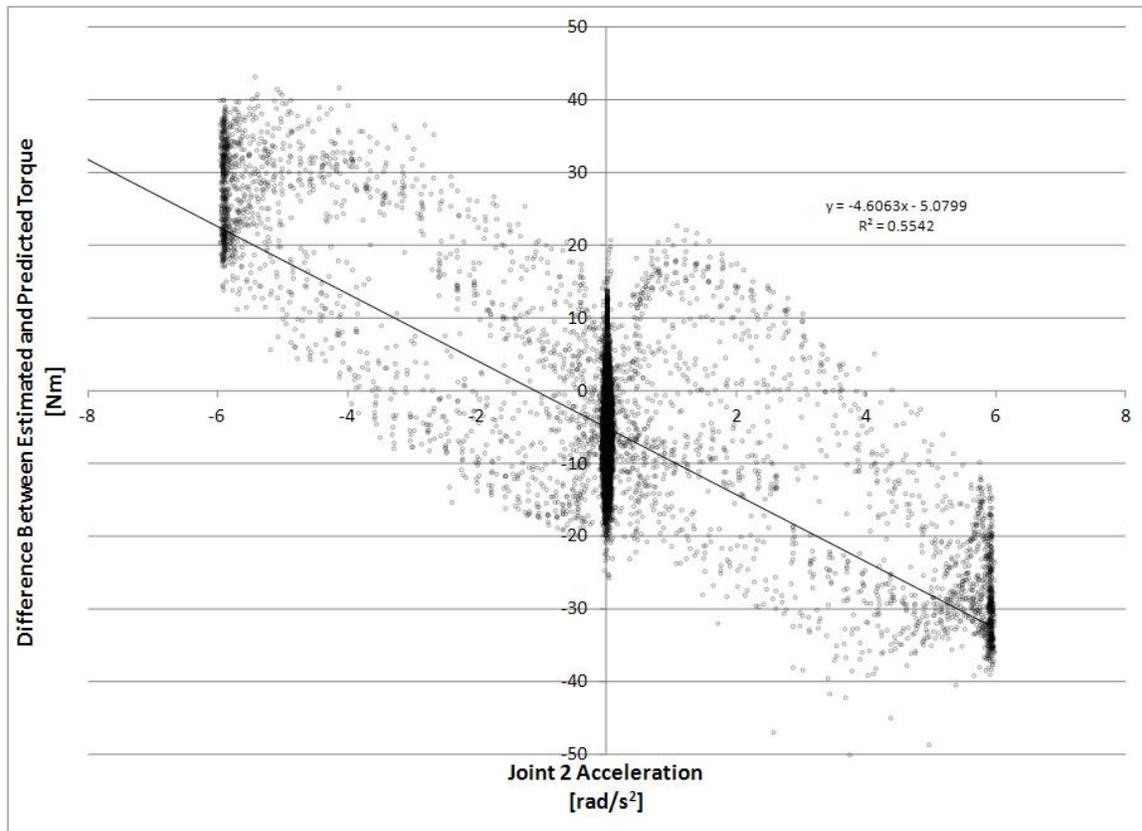


Figure 7.5: Example lumped inertia parameter graph

The motion planner used in the simulations was a very basic simulation of linear end effector motions at a constant velocity because the details of motion planning lie outside of the scope of this work. The motions executed in the simulations section all moved over short distances so motion planning did not play a significant role. In this work, the focus was on the response at and following detection. In many other cases, the particular motion of the manipulator will be more important. Adapting the simulator to accommodate generalized motion planning would expand the application area and would be a useful tool when planning and implementing collision detection. Given a reliable means for predicting collision detection sensitivity, an advanced motion planner may be used to avoid insensitive or seek sensitive configurations.

It would also be interesting to simulate collisions occurring at other locations on the manipulator. Collision prediction/avoidance software using 3D models may be able to calculate the distance to environmental objects. Given the complete environmental model, collisions could be simulated at other points on the manipulator. These can be principally simulated as collisions with a manipulator with fewer degrees of freedom, just as these simulations on a 7-DOF manipulator are a basis for collisions happening at the 7th link of a manipulator with more degrees of freedom. If the manipulator was torque controlled, the additional links would likely affect the ability if the manipulator to stop. Since the manipulator is position controlled, the only concern would be if the other links would come into contact with the object during the stopping portion of the collision response.

APPENDIX

The simulator code is included here.

```
/*
  This program is designed and written by Kyle Schroeder to simulate the
  safety of collision detection given uncertainties.
  The program simulates an industrial position controller with a desired
  end effector cartesian velocity. The controller for the joint
  positions is simulated at 5x the simulation speed.
  Bandwidth is simulated by choosing a delta t in thousandths of a
  second, e.g. 1000Hz = 0.001s is the highest bandwidth simulation
  currently permitted.
  */

#define USERANDPOS
#include "CommonIncludes.h"

void CalculateConditionNumber(FKJacobian *fkj, Vector jointPosition,
Vector &detectionConditionNumber, Vector handVel)
{
  Matrix Jacobian(6, DOF);
  Matrix JacobianTrans(DOF,6);
  Matrix JacobianTransInv(6,DOF);
  bool bLeftInv;
  Jacobian = *fkj->GetJacobian(jointPosition);
  Jacobian.t(JacobianTrans);
```

```

//calculate condition number
cv::Mat JacTransCV(DOF, 6, CV_64F);
cv::Mat JacTransInvCV(6,DOF,CV_64F);

//calculate inverse of condition number with Jacobian Transpose (I
think THIS is what we want, because we want the relationship between
the forces and the zero torque space)
for(int r = 0; r < DOF; r++)
{
    for(int c = 0; c < 6; c++)
    {
        JacTransCV.at<double>(r,c) = JacobianTrans.at(r,c);
    }
}
cv::SVD jactranssvd(JacTransCV);
double JTminNorm = 0;
JTminNorm = cv::invert(JacTransCV, JacTransInvCV, CV_SVD);
Matrix UMat(jactranssvd.u.size.p[0],jactranssvd.u.size.p[1]);
Matrix VtMat(jactranssvd.vt.size.p[0],jactranssvd.vt.size.p[1]);
Matrix DMat(jactranssvd.w.size.p[0],jactranssvd.w.size.p[0]);
for(int j = 0; j < DMat.nRow(); j++)
{
    DMat.at(j,j) = jactranssvd.w.at<double>(j,0);
}
for(int r = 0; r < jactranssvd.vt.size.p[0]; r++)
{
    for(int c = 0; c < jactranssvd.vt.size.p[1]; c++)
    {
        VtMat.at(r,c) = jactranssvd.vt.at<double>(r,c);
    }
}
for(int r = 0; r < jactranssvd.u.size.p[0]; r++)
{
    for(int c = 0; c < jactranssvd.u.size.p[1]; c++)
    {
        UMat.at(r,c) = jactranssvd.u.at<double>(r,c);
    }
}

//If Vt transforms from our original space to a space where the
eigenvalues scale the components, then they are transformed again by U
to the final output space, the joint torque space...
//A force along the same axis as the least of the "eigenvalues" can be
transformed back to the force space and the angle compared.

//The vector in the V space will be in the Z direction.
Vector transformedForce(6);
transformedForce = VtMat*handVel;

fstream forceout("Torque RowSpace.txt", ios::app);

```

```

Vector    nullForce(6);
Vector    outputTorque(DOF);
for(int dir = 0; dir < 6; dir++)
{
    for(int j = 0; j < 6; j++)
    {
        nullForce.at(j) = 0;
    }
    nullForce.at(dir) = 1;
    nullForce = VtMat.t()*nullForce;
    outputTorque = JacobianTrans*nullForce;
    forceout << dir+1 << "\t" << DMat.at(dir,dir) << "\t";
    for(int j = 0; j < DOF; j++)
    {
        forceout << outputTorque.at(j) << "\t";
    }
    forceout << "\t";
}
forceout << endl;
forceout.close();

```

```

detectionConditionNumber.at(0) = 1.0/JTminNorm;
detectionConditionNumber.at(1) = jactranssvd.w.at<double>(0,0);
detectionConditionNumber.at(2) = jactranssvd.w.at<double>(1,0);
detectionConditionNumber.at(3) = jactranssvd.w.at<double>(2,0);
detectionConditionNumber.at(4) = jactranssvd.w.at<double>(3,0);
detectionConditionNumber.at(5) = jactranssvd.w.at<double>(4,0);
detectionConditionNumber.at(6) = jactranssvd.w.at<double>(5,0);
detectionConditionNumber.at(7) = transformedForce.at(0);
detectionConditionNumber.at(8) = transformedForce.at(1);
detectionConditionNumber.at(9) = transformedForce.at(2);
detectionConditionNumber.at(10) = transformedForce.at(3);
detectionConditionNumber.at(11) = transformedForce.at(4);
detectionConditionNumber.at(12) = transformedForce.at(5);
detectionConditionNumber.at(13) = jactranssvd.w.size.p[0];
}

```

```

void GetNextHandDontMoveJ7(FKJacobian *fkj, Vector
&currentJointPositions, Vector &currentJointVelocities, Vector
&currentJointAccelerations, Vector maxJointAccelerations, Vector
desiredHandVelocity)
{
    //the joint position controller is still simulated to run at 5x the
simulation speed.
    int    numLoopCounts = 5;
    double controlDeltaT = simulationDeltaT/numLoopCounts;
    Vector interJointPos(currentJointPositions);
    Vector interJointVel(currentJointVelocities);
    Vector interJointAcc(currentJointAccelerations);
    for(int loopCount = 0; loopCount < numLoopCounts; loopCount++)
    {

```

```

Matrix  Jacobian(6, DOF);
Matrix  RedJac(6,6);
Matrix  RedJacInv(6,6);
bool    bLeftFlag;
Jacobian = *fkj->GetJacobian(interJointPos);
for(int r = 0; r < 6; r++) //only use the 6x6 jacobian because joint
7 is locked in these examples.
{
    for(int c = 0; c < 6; c++)
    {
        RedJac.at(r,c) = Jacobian.at(r,c);
    }
}
GetSingularInverse(RedJac, RedJacInv, bLeftFlag, 1e-7);
Vector  desiredJointVelocities(6);
Vector  desiredJointAcceleration(6);
Vector  accelerationRatio(6);
double  minRatio = 1;
desiredJointVelocities = RedJacInv*desiredHandVelocity;
//identify the joint with the minimum acceleration ratio
for(int j = 0; j < 6; j++)
{
    desiredJointAcceleration.at(j) = (desiredJointVelocities.at(j) -
interJointVel.at(j))/controlDeltaT;
    accelerationRatio.at(j) =
maxJointAccelerations.at(j)/fabs(desiredJointAcceleration.at(j));
    if(accelerationRatio.at(j) < minRatio)
    {
        minRatio = accelerationRatio.at(j);
    }
}
//scale each joint acceleration so the joint with the 'max'
acceleration is accelerating at it's maximum
for(int j = 0; j < 6; j++)
{
    interJointAcc.at(j) = minRatio*desiredJointAcceleration.at(j);
    //if the change in velocity is greater than the current difference
between the desired and actual....
    if(fabs(interJointAcc.at(j)*controlDeltaT) >
fabs(interJointVel.at(j)-desiredJointVelocities.at(j)))//the change in
velocity will be greater than the difference between the desired and
current
    {
        //then just set the current velocity equal to the desired velocity
interJointVel.at(j) = desiredJointVelocities.at(j);
    }else
    {
        interJointVel.at(j) += interJointAcc.at(j)*controlDeltaT;
    }
    interJointPos.at(j) += interJointVel.at(j)*controlDeltaT;
}
}
currentJointPositions = interJointPos;

```

```

    currentJointVelocities = interJointVel;
    currentJointAccelerations = interJointAcc;
}

bool ModelCollisionForce(Vector desiredHandVel, HandPose
initHandPosition, HandPose actualHandPosition, Vector stiffness,
HandPose collisionLocation, Vector &collisionForce)
{
    if(collisionForce.GetSize() != 6)
    {
        cout << "Error: Incorrect vector size.\nPlease use collisionForce
size 6.\n";
        return false;
    }
    for(int dir = 0; dir < 3; dir++)
    {
        //check for collision in each direction.
        if(fabs(actualHandPosition.at(dir) - initHandPosition.at(dir)) >
fabs(collisionLocation.at(dir) - initHandPosition.at(dir)) &&
fabs(desiredHandVel.at(dir)) > 0)
        {
            //cout << "Force exists" << endl;
            collisionForce.at(dir) =
stiffness.at(dir)*fabs(actualHandPosition.at(dir) -
collisionLocation.at(dir));
        } else
        {
            collisionForce.at(dir) = 0;
        }
    }
    return true;
}

void measureTorque(IDSANewtonEuler *idsane, Vector jointPosition,
Vector jointVelocity, Vector jointAcceleration, Vector &measuredTorque,
Vector collisionForce=Vector(6))
{
    idsane->SetJointPosition(jointPosition);
    HandPose collisionLoad(Orientation::FixedXYZ);
    for(int dir = 0; dir < 6; dir++)
    {
        collisionLoad.at(dir) = collisionForce.at(dir);
    }
    measuredTorque = idsane->GetJointTorques(jointVelocity,
jointAcceleration, collisionLoad);
}

bool IsCollisionDetected(Vector predictedTorque, Vector measuredTorque,
Vector jointThresholds)

```

```

{
  for(int j = 0; j < DOF; j++)
  {
    if(fabs(predictedTorque.at(j) - measuredTorque.at(j)) >
jointThresholds.at(j))
    {
      return true;
    }
  }
  return false;
}

```

```

bool DetectCollisions(IDSANewtonEuler *idsane, Vector jointPosition,
Vector jointVelocity, Vector jointAcceleration, Vector collisionForce,
Vector jointThresholds, Vector &predictedTorque, Vector
&measuredTorque, Vector randomTorqueErrors)
{
  measureTorque(idsane, jointPosition, jointVelocity, jointAcceleration,
predictedTorque);
  measureTorque(idsane, jointPosition, jointVelocity, jointAcceleration,
measuredTorque, collisionForce);
  for(int j = 0; j < DOF; j++)
  {
    measuredTorque.at(j) += randomTorqueErrors.at(j);
  }
  return IsCollisionDetected(predictedTorque, measuredTorque,
jointThresholds);
}

```

```

bool CalculateTriggerForce(IDSANewtonEuler *idsane, Vector
jointPosition, Vector thresholds, Vector &triggerForce)
{
  for(int dir = 0; dir < 3; dir++)
  {
    HandPose unitForce(Orientation::FixedXYZ);
    Vector unitLoadTorque(DOF);
    unitForce.at(dir) = 1;
    idsane->SetJointPosition(jointPosition);
    idsane->GetLoadTorques(unitForce, unitLoadTorque);
    double maxRatio;
    double tempRatio;
    maxRatio = 1e10;
    for(int j = 0; j < DOF; j++)
    {
      if(fabs(unitLoadTorque.at(j)) > 1e-3)
      {
        tempRatio = thresholds.at(j)/fabs(unitLoadTorque.at(j));
        if(tempRatio > 1e-3 && tempRatio < maxRatio)
        {
          maxRatio = tempRatio;
        }
      }
    }
  }
}

```

```

    }
  }
}
triggerForce.at(dir) = maxRatio;
}
return true;
}

```

```

void InputParameters(Vector &initJointPosition, double
&bandwidthDeltaT, double &contactArea, double &filterCutoffFrequency,
Vector &initDesiredHandVelocity, Vector &jointThresholds, Vector
&maxAccelerations, Vector &meanTorqueErrors, int &noSimulations, Vector
&StdevTorqueErrors, Vector &stiffness)
{
  int UserInputDataFlag = -1;
  do
  {
    cout << "Initial joint positions are currently " <<
initJointPosition*RadToDeg << "degrees. Enter 1 to change it, 0 to keep
it.\n";
    cin >> UserInputDataFlag;
  } while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
  if(UserInputDataFlag)
  {
    for(int j = 0; j < DOF; j++)
    {
      do
      {
        cout << "Please enter the initial position of joint " << j+1 << "
IN DEGREES.\nCurrently: " << initJointPosition.at(j)*RadToDeg << endl;
        cin >> initJointPosition.at(j);
      } while(fabs(initJointPosition.at(j)) > 180);
    }
    initJointPosition *= DegToRad;
  }

  do
  {
    cout << "Controller Feedback Delta T is currently " <<
bandwidthDeltaT << ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
  } while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
  if(UserInputDataFlag)
  {
    do
    {
      cout << "Please enter the controller feedback delta T\nCurrently: "
<< bandwidthDeltaT << endl;
      cin >> bandwidthDeltaT;
    } while(bandwidthDeltaT <= 0);
  }
}

```

```

do
{
    cout << "Simulation Delta T is currently " << simulationDeltaT << ".
Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    do
    {
        cout << "Please enter the simulation delta T\nCurrently: " <<
simulationDeltaT << endl;
        cin >> simulationDeltaT;
    }while(simulationDeltaT <= 0);
    }

do
{
    cout << "Stiffness of the object is currently " << stiffness << ".
Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < 6; j++)
    {
        do
        {
            cout << "Please enter the desired stiffness of direction " << j+1
<< ".\nCurrently: " << stiffness.at(j) << endl;
            cin >> stiffness.at(j);
        }while(stiffness.at(j) < 0);
    }
}

do
{
    cout << "Desired hand velocity is currently " <<
initDesiredHandVelocity << ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < 6; j++)
    {
        do
        {
            cout << "Please enter the desired hand velocity for direction " <<
j+1 << ".\nCurrently: " << initDesiredHandVelocity.at(j) << endl;
            cin >> initDesiredHandVelocity.at(j);
        }while(fabs(initDesiredHandVelocity.at(j)) > 1.0);
    }
}

```

```

}

do
{
    cout << "Filter cutoff frequency is currently " <<
filterCutoffFrequency << ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    do
    {
        cout << "Please enter the desired filter cutoff
frequency.\nCurrently:\n";
        cin >> filterCutoffFrequency;
    }while(filterCutoffFrequency < 0);
}

do
{
    cout << "Maximum accelerations are currently " << maxAccelerations <<
". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < DOF; j++)
    {
        do
        {
            cout << "Please enter the desired maximum acceleration for joint "
<< j+1 << ".\nCurrently: " << maxAccelerations.at(j) << endl;
            cin >> maxAccelerations.at(j);
        }while(maxAccelerations.at(j) <= 0.0);
    }
}

do
{
    cout << "Detection thresholds are currently " << jointThresholds <<
". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < DOF; j++)
    {
        do
        {
            cout << "Please enter the desired error threshold for joint " <<
j+1 << ".\nCurrently: " << jointThresholds.at(j) << endl;
            cin >> jointThresholds.at(j);
        }while(jointThresholds.at(j) <= 0);
    }
}

```

```

    }
}

do
{
    cout << "The contact area, in SQUARE MILLIMETERS is " << contactArea
<< ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    do
    {
        cout << "Please enter the desired contact area.\n";
        cin >> contactArea;
    }while(contactArea < 0);
}

for(int j = 0; j < DOF; j++)
{
    meanTorqueErrors.at(j) = 0.0;
    StdevTorqueErrors.at(j) = jointThresholds.at(j)/5.5;
}

do
{
    cout << "Number of simulations is currently " << noSimulations << ".
Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    do
    {
        cout << "Please enter the desired number of simulations.\nCurrently:
" << noSimulations << endl;
        cin >> noSimulations;
    }while(noSimulations <= 0);
}

do
{
    cout << "Mean torque estimation errors currently " <<
meanTorqueErrors << ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < DOF; j++)
    {
        do
        {

```

```

        cout << "Please enter the desired mean torque error for joint " <<
j+1 << ".\nCurrently: " << meanTorqueErrors.at(j) << endl;
        cin >> meanTorqueErrors.at(j);
    }while(meanTorqueErrors.at(j) == 0 || fabs(meanTorqueErrors.at(j)) >
jointThresholds.at(j));
    }
}

do
{
    cout << "Torque error standard deviation is currently " <<
StdevTorqueErrors << ". Enter 1 to change it, 0 to keep it.\n";
    cin >> UserInputDataFlag;
}while(UserInputDataFlag != 0 && UserInputDataFlag != 1);
if(UserInputDataFlag)
{
    for(int j = 0; j < DOF; j++)
    {
        do
        {
            cout << "Please enter the desired standard deviation for torque
error of joint " << j+1 << ".\nCurrently: " << StdevTorqueErrors.at(j)
<< endl;
            cin >> StdevTorqueErrors.at(j);
        }while(StdevTorqueErrors.at(j) == 0);
    }
}

cout << "Enter 1 to initialize the stats file." << endl;
int dummyIntEnter = 0;
cin >> dummyIntEnter;
if(dummyIntEnter == 1)
{
    fstream StatsOut("StatOutput.txt", ios::out);
    StatsOut << "Timestamp\tCollision Time\tMax Time\tSimulation Delta
T\tBandwidth Delta T\tFilter Cutoff\t";
    for(int j = 0; j < DOF; j++)
    {
        StatsOut << "Initial Joint Position " << j+1 << "\t";
    }
    for(int dir = 0; dir < 6; dir++)
    {
        StatsOut << "EEF Velocity ";
        switch(dir)
        {
            case 0:
                StatsOut << "X";
                break;
            case 1:
                StatsOut << "Y";
                break;
            case 2:
                StatsOut << "Z";

```

```

    break;
case 3:
    StatsOut << "Alpha";
    break;
case 4:
    StatsOut << "Beta";
    break;
case 5:
    StatsOut << "Gamma";
    break;
}
StatsOut << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Max Acc " << j+1 << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Mean Error " << j+1 << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Error Standard Deviation " << j+1 << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Threshold " << j+1 << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Torque Error at Detection " << j+1 << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << "Random Torque at Detection " << j+1 << "\t";
}
for(int j = 0; j < 6; j++)
{
    StatsOut << "Stiffness in direction ";
    switch(j)
    {
    case 0:
        StatsOut << "X";
        break;
    case 1:
        StatsOut << "Y";
        break;
    case 2:
        StatsOut << "Z";
        break;
    case 3:
        StatsOut << "Alpha";

```

```

        break;
    case 4:
        StatsOut << "Beta";
        break;
    case 5:
        StatsOut << "Gamma";
        break;
    }
    StatsOut << "\t";
}
for(int j = 0; j < 6; j++)
{
    StatsOut << "Max Force in Direction ";
    switch(j)
    {
    case 0:
        StatsOut << "X";
        break;
    case 1:
        StatsOut << "Y";
        break;
    case 2:
        StatsOut << "Z\tMax Force Magnitude";
        break;
    case 3:
        StatsOut << "Alpha";
        break;
    case 4:
        StatsOut << "Beta";
        break;
    case 5:
        StatsOut << "Gamma";
        break;
    }
    StatsOut << "\t";
}
for(int j = 0; j < 6; j++)
{
    StatsOut << "Max Collision Deflection ";
    switch(j)
    {
    case 0:
        StatsOut << "X";
        break;
    case 1:
        StatsOut << "Y";
        break;
    case 2:
        StatsOut << "Z";
        break;
    case 3:
        StatsOut << "Alpha";
        break;

```

```

    case 4:
        StatsOut << "Beta";
        break;
    case 5:
        StatsOut << "Gamma";
        break;
    }
    StatsOut << "\t";
}
StatsOut << "Simulation number\tMax Pressure\t";
for(int j = 0; j < 14; j++)
{
    StatsOut << "Condition Number " << j+1 << "\t";
}
for(int j = 0; j < 6; j++)
{
    StatsOut << "Collision Hand ";
    switch(j)
    {
    case 0:
        StatsOut << "X";
        break;
    case 1:
        StatsOut << "Y";
        break;
    case 2:
        StatsOut << "Z";
        break;
    case 3:
        StatsOut << "Alpha";
        break;
    case 4:
        StatsOut << "Beta";
        break;
    case 5:
        StatsOut << "Gamma";
        break;
    }
    StatsOut << "\t";
}
StatsOut << endl;
StatsOut.close();
}
}

```

```

int main(int argc, char *argv[])
{
    //seed random numbers
    srand (time(NULL));

    int      RepeatCalcs;
    do

```

```

{
  //This program will execute the various necessary functions in a
  method. The goal is to write all the data to files but not to carry
  vectors of Vectors all the way through.
  Vector      jointThresholds(DOF);
  Vector      maxAccelerations(DOF);
  Vector      jointPosition(DOF);
  Vector      initJointPosition(DOF);
  Vector      jointVelocity(DOF);
  Vector      jointAcceleration(DOF);
  Vector      filtJointAcc(DOF);
  Vector      actualJointVelocity(DOF);
  Vector      actualJointAcceleration(DOF);
  HandPose    actualHandPosition(Orientation::FixedXYZ);
  Vector      actualHandVelocity(6);
  Vector      actualHandAcceleration(6);
  Vector      actualHandPositionVector(6);
  Vector      integralError(6);
  Vector      collisionForce(6);
  Vector      triggerForce(6);
  Vector      forceAtDetection(6);
  Vector      errorAtDetection(DOF);
  Vector      predictedTorque(DOF);
  Vector      measuredTorque(DOF);
  Vector      maxForce(6);
  Vector      randomTorqueErrors(DOF);
  Vector      RandomTorqueAtDetection(DOF);
  JointVector PositionJointVector(DOF);
  HandPose    collisionLocation(Orientation::FixedXYZ);
  Vector      stiffness(6);
  FKJacobian  *fkj;
  IKJReconfig<> *ikj;
  IDNewtonEuler *idne;
  IDSANewtonEuler *idsane;
  HandPose    collisionHand(Orientation::FixedXYZ);
  HandPose    desiredHandVelocity(Orientation::FixedXYZ);
  HandPose    initDesiredHandVelocity(Orientation::FixedXYZ);
  FiniteDifference calcFDVelAcc(DOF, bandwidthDeltaT);
  Vector      FilteredUncertainty(DOF);

  double      contactArea = 100.0; //contact area in square millimeters
  bool        bErrorGreaterThanThreshold = false;
  int         noSimulations = 1000;
  double      filterCutoffFrequency = 30;
  Vector      initialConditionNumber(14);
  Vector      meanTorqueErrors(DOF);
  Vector      StdevTorqueErrors(DOF);
  Vector      maxCollisionDeflection(6);
  Vector      detectionConditionNumber(14); //doesn't need to be a
  vector except that it gets pushed into the output vectors this way.
  Vector      collisionHandVector(6);

```

```

//set initial position of manipulator. This might, in future, be
changed to allow input of general configuration, either joint or hand
position
initJointPosition.at(0) = 0;
initJointPosition.at(1) = -30;
initJointPosition.at(2) = 0;
initJointPosition.at(3) = 70;
initJointPosition.at(4) = 0;
initJointPosition.at(5) = -40;
initJointPosition.at(6) = 0;
initJointPosition *= OSCAR::DegToRad;

//set joint position JointVector for use in ikjreconfig object
for(int j = 0; j < DOF; j++)
{
    PositionJointVector.at(j) = jointPosition.at(j);
}
//initialize OSCAR objects and import default robot data.
fkj = new FKJacobian("C:\\KyleDataFiles\\sia5.dh");
ikj = new IKJReconfig<>(PositionJointVector, fkj);
idne = new IDNewtonEuler(DOF, "C:\\KyleDataFiles\\sia5.cgm",
"C:\\KyleDataFiles\\sia5.in", Vector3(0.0, 0.0, -9.81));
idsane = new IDSANewtonEuler(idne, fkj);
ReadThresholds("C:\\KyleDataFiles\\thresholds.th", DOF,
jointThresholds);
ReadMaxAccelerations("C:\\KyleDataFiles\\maxAcceleration.macc", DOF,
maxAccelerations);

//Get the initial hand position
fkj->GetHandPose(jointPosition)->Get(actualHandPosition);
//set defaults for collision location, stiffness of object, and
desired hand velocity
initDesiredHandVelocity.at(2) = -0.05;
for(int dir = 0; dir < 6; dir++)
{
    if(fabs(initDesiredHandVelocity.at(dir)) > 0)
    {
        collisionHand.at(dir) = actualHandPosition.at(dir) +
sign(initDesiredHandVelocity.at(dir))*fabs(GetNormalRandomNumber(0.0,
initDesiredHandVelocity.at(dir)/10.0));// +
initDesiredHandVelocity.at(dir) + GetNormalRandomNumber(0.0,
initDesiredHandVelocity.at(dir)/10.0);
    }else
    {
        collisionHand.at(dir) = actualHandPosition.at(dir) - 0.1;
    }
    stiffness.at(dir) = 75000;
}

InputParameters(initJointPosition, bandwidthDeltaT, contactArea,
filterCutoffFrequency, initDesiredHandVelocity, jointThresholds,

```

```
maxAccelerations, meanTorqueErrors, noSimulations, StdevTorqueErrors,
stiffness);
```

```
for(int repeat = 0; repeat < noSimulations; repeat++)
{
    FiniteDifference *HandVelAccCalc;
    ButterLPFilter *handVelFilter;
    ButterLPFilter *handAccFilter;
    vector<double> simulationTimeVector;
    vector<double> bandwidthTimeVector;
    vector<Vector> dummyVect;
    double maxTime = 5.0; //seconds
    double collisionTime;
    bool bCollisionDetected = false;
    bool bRespondedToCollision = false;

    bCollisionDetected = false;
    bRespondedToCollision = false;
    int simTime = 0;
    collisionTime = 0;
    maxTime = 5;

    //reinitialize all the variables that need reset.
#ifdef USERANDPOS
    jointPosition = initJointPosition;
    //or randomize joint position
    /*--Randomize joint positions --*/
    initJointPosition.at(0) = GetUniformRandomNumber(-180,180);
    initJointPosition.at(1) = GetUniformRandomNumber(-110,110);
    initJointPosition.at(2) = GetUniformRandomNumber(-170,170);
    initJointPosition.at(3) = GetUniformRandomNumber(0,205);
    initJointPosition.at(4) = GetUniformRandomNumber(-180,180);
    initJointPosition.at(5) = GetUniformRandomNumber(-110,110);
    initJointPosition.at(7) = 0;
    initJointPosition *= DegToRad;
    jointPosition = initJointPosition;
    /*--End randomize joint positions --*/

    /*--Randomize hand velocity --*/
    {
        double velMag = GetNormalRandomNumber(0.010, 0.002);
        double velPsi = GetUniformRandomNumber(0, 180)*DegToRad; //Angle
down from initial axis (if zero, force is in z direction)
        double velPhi = GetUniformRandomNumber(-179, 180)*DegToRad;
//Angle about the initial axis (if psi is 90 and this is 0, in x
direction, if psi is 90 and this is 90, in y direction)
        initDesiredHandVelocity.at(0) = velMag*sin(velPsi)*cos(velPhi);
        initDesiredHandVelocity.at(1) = velMag*sin(velPsi)*sin(velPhi);
        initDesiredHandVelocity.at(2) = velMag*cos(velPsi);
    }
#endif
}
#endif
```

```

//set joint position JointVector for use in ikjreconfig object
for(int j = 0; j < DOF; j++)
{
    PositionJointVector.at(j) = jointPosition.at(j);
    randomTorqueErrors.at(j) = 0.0;
}
for(int dir = 0; dir < 6; dir++)
{
    desiredHandVelocity.at(dir) = initDesiredHandVelocity.at(dir);
}

fkj->GetHandPose(jointPosition)->Get(actualHandPosition);
HandPose initHandPosition = actualHandPosition;

HandVelAccCalc = new FiniteDifference(6, simulationDeltaT);
handVelFilter = new ButterLPFilter(simulationDeltaT,
filterCutoffFrequency, 6);
handAccFilter = new ButterLPFilter(simulationDeltaT,
filterCutoffFrequency, 6);

for(int dir = 0; dir < 6; dir++)
{
    actualHandVelocity.at(dir) = desiredHandVelocity.at(dir);
    actualHandAcceleration.at(dir) = 0.0;
    actualHandPositionVector.at(dir) = actualHandPosition.at(dir);
    maxForce.at(dir) = 0.0;
    maxCollisionDeflection.at(dir) = 0.0;
    if(fabs(initDesiredHandVelocity.at(dir)) > 0)
    {
        collisionHand.at(dir) = actualHandPosition.at(dir) +
sign(initDesiredHandVelocity.at(dir))*fabs(GetNormalRandomNumber(0.0,
initDesiredHandVelocity.at(dir)/10.0));// +
initDesiredHandVelocity.at(dir) + GetNormalRandomNumber(0.0,
initDesiredHandVelocity.at(dir)/10.0);
    }else
    {
        collisionHand.at(dir) = actualHandPosition.at(dir) - 0.1;
    }
    collisionHandVector.at(dir) = collisionHand.at(dir);
}
HandPose actualHandVelHP;
for(int dir = 0; dir < 6; dir++)
{
    actualHandVelHP.at(dir) = actualHandVelocity.at(dir);
}
JointVector actualJointVelocityJV(DOF);
ikj->SetJointPosition(PositionJointVector);
CalculateConditionNumber(fkj, jointPosition, initialConditionNumber,
collisionForce);
ikj->GetJointVelocity(actualHandVelHP, actualJointVelocityJV);
for(int j = 0; j < DOF; j++)

```

```

{
  actualJointVelocity.at(j) = actualJointVelocityJV.at(j);
  actualJointAcceleration.at(j) = 0.0;
  RandomTorqueAtDetection.at(j) = 0.0;
  errorAtDetection.at(j) = 0.0;
}

//calculate hand position (and velocity)
HandVelAccCalc->Initialize(actualHandPositionVector);
handVelFilter->Initialize(actualHandVelocity);
handAccFilter->Initialize(actualHandAcceleration);

do
{
  simulationTimeVector.clear();
} while(!simulationTimeVector.empty());
do
{
  bandwidthTimeVector.clear();
} while(!bandwidthTimeVector.empty());
do
{
  dummyVect.clear();
} while(!dummyVect.empty());

//Get the initial hand position
fkj->GetHandPose(jointPosition)->Get(actualHandPosition);

do
{

  //simulates industrial controller on joint position, cycling
  several times for each simulation cycle, which happens several times
  for each bandwidth cycle
  GetNextHandDontMoveJ7(fkj, jointPosition, actualJointVelocity,
  actualJointAcceleration, maxAccelerations, desiredHandVelocity);

  //calculate hand position (and velocity)
  fkj->GetHandPose(jointPosition)->Get(actualHandPosition);
  for(int dir = 0; dir < 6; dir++)
  {
    actualHandPositionVector.at(dir) = actualHandPosition.at(dir);
  }
  HandVelAccCalc->GetAccVel(actualHandPositionVector,
actualHandVelocity, actualHandAcceleration);
  //filter hand vel and acc estimates
  handVelFilter->Update(actualHandVelocity);
  handAccFilter->Update(actualHandAcceleration);

  //calculate eef load

```

```

    ModelCollisionForce(initDesiredHandVelocity, initHandPosition,
actualHandPosition, stiffness, collisionHand, collisionForce);

    //estimate the trigger force based on configuration and joint
thresholds
    CalculateTriggerForce(idsane, jointPosition, jointThresholds,
triggerForce);

    CalculateConditionNumber(fkj, jointPosition,
initialConditionNumber, collisionForce);

    //generate random error
    for(int j = 0; j < DOF; j++)
    {
        randomTorqueErrors.at(j) =
GetNormalRandomNumber(meanTorqueErrors.at(j), StdevTorqueErrors.at(j));
    }

    bErrorGreaterThanThreshold = false;
    bErrorGreaterThanThreshold = DetectCollisions(idsane,
jointPosition, actualJointVelocity, actualJointAcceleration,
collisionForce, jointThresholds, predictedTorque, measuredTorque,
randomTorqueErrors);

    //only detect collisions at bandwidth time samples.
    if(((simTime))%(int)(bandwidthDeltaT/simulationDeltaT) == 0)
    {
        //Detect collisions
        if(bErrorGreaterThanThreshold)
        {
            if(!bCollisionDetected)
            {
                //Collision??
                //If so, change the joint acceleration to max decelerate.
                bCollisionDetected = true;
                RandomTorqueAtDetection = randomTorqueErrors;
                CalculateConditionNumber(fkj, jointPosition,
detectionConditionNumber, collisionForce);
                //find maximum joint velocity
                double fastestJointVel = 0;
                double accelerationOfJoint = 0;
                for(int j = 0; j < DOF; j++)
                {
                    if(fabs(actualJointVelocity.at(j)) > fastestJointVel)
                    {
                        fastestJointVel = actualJointVelocity.at(j);
                        accelerationOfJoint = maxAccelerations.at(j);
                    }
                }
                collisionTime = ((double)simTime)*simulationDeltaT;
                maxTime = collisionTime + 0.05 +
fastestJointVel/accelerationOfJoint*1.5;

```

```

        for(int j = 0; j < DOF; j++)
        {
            errorAtDetection.at(j) = measuredTorque.at(j) -
predictedTorque.at(j);
        }
        for(int dir = 0; dir < 6; dir++)
        {
            forceAtDetection.at(dir) = fabs(collisionForce.at(dir));
        }
    }
}

if(bCollisionDetected && !bRespondedToCollision)
{
    bRespondedToCollision = true;
    for(int dir = 0; dir < 3; dir++)
    {
        desiredHandVelocity.at(dir) *= -1.0;
    }
}

if(bCollisionDetected)
{
    for(int dir = 0; dir < 6; dir++)
    {
        //determine maximum force of collision
        if(fabs(collisionForce.at(dir)) > maxForce.at(dir))
        {
            maxForce.at(dir) = fabs(collisionForce.at(dir));
        }

        //determine maximum deflection during collision
        for(int dir = 0; dir < 6; dir++)
        {
            if(fabs(collisionHand.at(dir) - actualHandPosition.at(dir)) >
maxCollisionDeflection.at(dir))
            {
                maxCollisionDeflection.at(dir) = fabs(collisionHand.at(dir) -
actualHandPosition.at(dir));
            }
        }
    }
}

//update time
simTime++;

} while(((double)simTime)*simulationDeltaT <=
maxTime+simulationDeltaT);

stringstream datestampss;

```

```

time_t    rawTime;
struct tm  *currTime;
time(&rawTime);
currTime = localtime(&rawTime);
datestampss << currTime->tm_year+1900 << "-";
if(currTime->tm_mon+1 < 10)
{
    datestampss << "0";
}
datestampss << currTime->tm_mon+1 << "-";
if(currTime->tm_mday < 10)
{
    datestampss << "0";
}
datestampss << currTime->tm_mday << " ";
if(currTime->tm_hour < 10)
{
    datestampss << "0";
}
datestampss << currTime->tm_hour << "-";
if(currTime->tm_min < 10)
{
    datestampss << "0";
}
datestampss << currTime->tm_min << "-";
if(currTime->tm_sec < 10)
{
    datestampss << "0";
}
datestampss << currTime->tm_sec;

```

```

fstream StatsOut("StatOutput.txt", ios::app);
StatsOut << datestampss.str() << "\t";
StatsOut << collisionTime << "\t";
StatsOut << maxTime << "\t";
StatsOut << simulationDeltaT << "\t";
StatsOut << bandwidthDeltaT << "\t";
StatsOut << filterCutoffFrequency << "\t";
for(int j = 0; j < DOF; j++)
{
    StatsOut << initJointPosition.at(j) << "\t";
}
for(int dir = 0; dir < 6; dir++)
{
    StatsOut << initDesiredHandVelocity.at(dir) << "\t";
}
for(int j = 0; j < DOF; j++)
{
    StatsOut << maxAccelerations.at(j) << "\t";
}
for(int j = 0; j < DOF; j++)

```

```

{
  StatsOut << meanTorqueErrors.at(j) << "\t";
}
for(int j = 0; j < DOF; j++)
{
  StatsOut << StdevTorqueErrors.at(j) << "\t";
}
for(int j = 0; j < DOF; j++)
{
  StatsOut << jointThresholds.at(j) << "\t";
}
for(int j = 0; j < DOF; j++)
{
  StatsOut << errorAtDetection.at(j) << "\t";
}
for(int j = 0; j < DOF; j++)
{
  StatsOut << RandomTorqueAtDetection.at(j) << "\t";
}
for(int j = 0; j < 6; j++)
{
  StatsOut << stiffness.at(j) << "\t";
}
for(int j = 0; j < 6; j++)
{
  StatsOut << maxForce.at(j) << "\t";
  if(j == 2)
  {
    StatsOut <<
sqrt(pow(maxForce.at(0),2.0)+pow(maxForce.at(1),2.0)+pow(maxForce.at(2)
,2.0)) << "\t";
  }
}
for(int j = 0; j < 6; j++)
{
  StatsOut << maxCollisionDeflection.at(j) << "\t";
}
StatsOut << repeat << "\t";
{
  double maxForceMag =
sqrt(pow(maxForce.at(0),2.0)+pow(maxForce.at(1),2.0)+pow(maxForce.at(2)
,2.0));
  StatsOut << maxForceMag/contactArea << "\t";
}
for(int j = 0; j < detectionConditionNumber.GetSize(); j++)
{
  StatsOut << detectionConditionNumber.at(j) << "\t";
}
for(int dir = 0; dir < 6; dir++)
{
  StatsOut << collisionHandVector.at(dir) << "\t";
}
StatsOut << endl;

```

```

StatsOut.close();

delete HandVelAccCalc;
delete handVelFilter;
delete handAccFilter;
if(repeat % 10 == 0 || repeat == noSimulations-1)
{
    cout << repeat << "/" << noSimulations << ": " <<
(double)repeat/(double)noSimulations*100.0 << "%" << endl;
}

}

Sleep(3000);
cout << "\n\n\n\nPlease enter 1 to repeat and 0 to exit.\n";
cin >> RepeatCalcs;
} while(RepeatCalcs != 0);
return 0;
}

```

REFERENCES

- Albu-Schaeffer, Alin, Sami Haddadin, Christian Ott, A. Stemmer, Thomas Wimboeck, and Gerd Hirzinger. "The DLR Lightweight Robot: Design and Control Concepts for Robots in Human Environments." *Industrial Robot: An International Journal*, vol. 35, no. 5, 2007: 376-385.
- Albu-Schaeffer, Alin, Oliver Eiberger, Markus Grebenstein, Sami Haddadin, Christian Ott, Thomas Wimboeck, Sebastian Wolf, and Gerd Hirzinger. "Soft Robotics." *IEEE Robotics and Automation Magazine*, September 2008: 20-30.
- Agile Planet. "AX-I3". Accessed: 28 February 2013. <http://www.agileplanet.com/products/ax-i3>
- Agile Planet. "RLX-I3." <http://www.agileplanet.com/products/rlx-i3>. Access: 5 Jan 2012.
- ANSI/RIA R15.06-1999 American National Standard for Industrial Robots and Robot Systems – Safety Requirements, 1999.
- BGIA – Institut fuer Arbeitsschutz der Deutschen Gesetzlichen Unfallversicherung. "BG/BGIA – Empfehlungen fuer die Gefaehrungsbeurteilung nach Maschinenrichtlinie (Gestaltung von Arbeitsplaetzen mit kollaborierenden Robotern)." U 001/2009 October 2009, Version February 2011.
- Bicchi, Antonio and Giovanni Tonietti. "Fast and 'Soft-Arm' Tactics." *IEEE Robotics and Automation Magazine*. June 2004: 22-33.
- Bicchi, Antonio, Michele Bavaro, Gianluca Boccadamo, Davide De Carli, Roberto Filippini, Giorgio Grioli, Marco Piccagallo, Alessandro Rosi, Riccardo Schiavi, Soumen Sen, and Giovanni Tonietti. "Physical Human-Robot Interaction: Dependability, Safety, and Performance. *IEEE International Workshop on Advanced Motion Control*, 2008: 9-14.
- Craig, John J. "Introduction to Robotics: Mechanics and Control." 3rd Edition. Upper Saddle River, NJ: Pearson Prentice Hall, 2005.
- Cross, Rod. "The Bounce of a Ball." *American Journal of Physics*, vol. 63, no. 3, March 1999: 222-227.
- De Luca, Alessandro, Alin Albu-Schaeffer, Sami Haddadin, and Gerd Hirzinger. "Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006: 1623-1630.
- Desert News. Salt Lake City, Utah. 8 December 1981. "Killer Robot". <http://news.google.com/newspapers?id=1t00AAAIBAJ&sjid=xoMDAAAIBA J&pg=6313,2597702&dq=kenji+urada&hl=en>

- Design Safety Engineering, Inc. "The Status of R15.06 – Robot Standard." Web blog post dated 17 March 2011. Accessed 25 October 2011.
- Dhaouadi, Rached, Fathi H Ghorbel, and Prasanna S Gandhi. "A New Dynamic Model of Hysteresis in Harmonic Drives." *IEEE Transactions on Industrial Electronics*, vol. 50, no. 6, December 2003: 1165-1171.
- Diftler, M.A., J.S. Mehling, M.E. Abdallah, N.A. Radford, L.B. Bridgwater, A.M. Sanders, R.S. Askew, D.M. Linn, J.D. Yamokoski, F.A. Permenter, B.K. Hargrave, R. Platt, R.T. Savely, and R.O. Ambrose. "Robonaut 2 – The First Humanoid Robot in Space." *IEEE International Conference on Robotics and Automation*. Shanghai, China. 2011. May 9-13.
- Duguleana, M., F.G. Barbuceanu, A. Teirelbar, G. Mogan. "Obstacle Avoidance of Redundant Manipulators Using Neural Networks Based Reinforcement Learning." *Robotics and Computer-Integrated Manufacturing* 28. 2012. Pp 132-146.
- Ebert, Dirk M and Dominik D Henrich. "Safe Human-Robot-Cooperation: Image-based Collision Detection for Industrial Robots." *IEEE International Conference on Intelligent Robots and Systems*. Lausanne, Switzerland, 2002: 1826-1831.
- Eitner, Christian, Yuto Mori, Kei Okada, and Masayuki Inaba. "Task and Vision Based Online Manipulator Trajectory Generation for a Humanoid Robot." *IEEE-RAS International Conference on Humanoid Robots*. Daejeon, Korea, 2008: 293-298.
- EuroNCAP. "Frontal Impact Testing Protocol." Version 5.1. June 2011.
- Foster, Chad, Pradeep Ashok, Daniel J. Cox, Delbert Tesar, Pete Pittman, Cameron J. Turner. "Computer Model and Simulation of a Glove Box Process." *American Nuclear Society 9th Topical Meeting on Robotics and Remote Systems*. March 4-8, 2001.
- Gadd, C W. "Use of Weighted Impulse Criterion for Estimating Injury Hazard." *10th Stapp Car Crash Conference*. New York, 1966: 164-174.
- Haddadin, Sami, Alin Albu-Schaeffer, and Gerd Hirzinger. "Safety Evaluation of Physical Human-Robot Interaction via Crash-Testing." *Robotics: Science and Systems Conference*, 2007: 217-224.
- Haddadin, Sami, Alin Albu-Schaeffer, Alessandro De Luca, and Gerd Hirzinger. "Collision Detection and Reaction: A Contribution to Safe Physical Human-Robot Interaction." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008: 3356-3363.
- Haddadin, Sami, Alin Albu-Schaeffer, and Gerd Hirzinger. "The Role of the Robot Mass and Velocity in Physical Human-Robot Interaction – Part I: Non-constrained Blunt Impacts." *IEEE International Conference on Robotics and Automation* 2008: 1331-1338.

- Haddadin, Sami, Alin Albu-Schaeffer, Mirko Frommberger, and Gerd Hirzinger. "The Role of the Robot Mass and Velocity in Physical Human-Robot Interaction – Part II: Constrained Blunt Impacts." IEEE International Conference on Robotics and Automation 2008: 1339-1345.
- Haddadin, Sami, Alin Albu-Schaeffer, Mirko Frommberger, Juergen Rossmann, and Gerd Hirzinger. "The 'DLR Crash Report': Towards a Standard Crash-Testing Protocol for Robot Safety – Part I: Results." IEEE International Conference on Robotics and Automation 2009: 272-279.
- Haddadin, Sami, Alin Albu-Schaeffer, Mirko Frommberger, Juergen Rossmann, and Gerd Hirzinger. "The 'DLR Crash Report': Towards a Standard Crash-Testing Protocol for Robot Safety – Part II: Discussions." IEEE International Conference on Robotics and Automation 2009: 280-287.
- Haddadin, Sami, Sven Parusel, Rico Belder, Joern Vogel, Tim Rokahr, Alin Albu-Schaeffer, and Gerd Hirzinger. "Holistic Design and Analysis for the Human-Friendly Robotic Co-Worker." IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010: 4735-4742.
- Haddadin, Sami, Alin Albu-Schaeffer, and Gerd Hirzinger. "Soft-Tissue Injury in Robotics." IEEE International Conference on Robotics and Automation, 2010: 3426-3433.
- Hansen, S.T., H.J. Andersen, and T. Bak. "Practical Evaluation of Robots for Elderly in Denmark – an Overview." ACM/IEEE International Conference on Human-Robot Interaction (HRI). 2010.
- Harden, Troy A. "Minimum Distance Influence Coefficients for Obstacle Avoidance in Manipulator Motion Planning." PhD Dissertation. The University of Texas at Austin. 2002.
- Harden, Troy, Chetan Kapoor, and Delbert Tesar. "Experimental Evaluation of a Criteria-Based Obstacle Avoidance Scheme." ASME Design and Engineering Technical Conference. DETC99/DAC-8657. Sept 1999.
- Hauschild, J-P, G.R. Heppler, and J.J. McPhee. "Friction Compensation of Harmonic Drive Actuators." 6th International Conference on Dynamics and Control of Systems and Structures in Space, Liguria, Italy, July, 2004: 683-692.
- Heinzmann, Jochen and Alexander Zelinsky. "Quantitative Safety Guarantees for Physical Human-Robot Interaction." International Journal of Robotics Research, 2003: 479-504.
- Ikuta, Koji and Makoto Nokata. "General Evaluation Method of Safety for Human-care Robots." IEEE International Conference on Robotics and Automation, May 1999: 2065-2072.

- Ikuta, Koji, Makoto Nokata, and Hideki Ishii. "General Danger-Evaluation Method of Human-Care Robot Control and Development of Special Simulator." IEEE International Conference on Robotics & Automation, 2001: 3181-3188.
- Ikuta, Koji, Kideki Ishii, and Makoto Nokata. "Safety Evaluation Method of Design and Control for Human-Care Robots." International Journal of Robotics Research, vol. 22, no. 5, May 2003: 281-297.
- iRobot. "NEW! iRobot Roomba 790". Accessed 4 March 2013. <http://store.irobot.com/product/index.jsp?productId=12991591&cp=2501652&s=A-ProductAge&parentPage=family>
- ISO 10218-1 Robots for Industrial Environments – Safety Requirements – Part 1: Robot. 2006.
- Je, Hwan-Jook, Jun-Young Baek, and Min-Cheol Lee. "A Study of the Collision Detection of Robot Manipulator without Torque Sensor." ICROS-SICE International Joint Conference 2009. Fukuoka, Japan, 2009: 4468-4471.
- Jung, David and Alexander Zelinsky. "Whisker-based Mobile Robot Navigation." Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. 1996.
- Kapoor, C. and D. Tesar. "A Reuseable Operational Software Architecture for Advanced Robotics (OSCAR)." The University of Texas at Austin. Report to U.S. Dept. of Energy, Grant No. DE-FG01 94EW37966 and NASA Grant No. NAG 9-809. 1998.
- Kawada Industries Inc. "Mechatronics: Next-generation industrial robot NEXTAGE." <http://global.kawada.jp/mechatronics/nextage.html> Accessed: 6 Jan 2012.
- Khatib, O. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." International Journal of Robotics Research. 1986. 5(1):90-98.
- Kravets, David. "Jan. 25, 1979: Robot Kills Human." Wired Magazine Online. <http://www.wired.com/thisdayintech/2010/01/0125robot-kills-worker/> Published: 25 January 2010. Accessed: 1 February 2012.
- Lim, Hun-ok and Kazuo Tanie. "Human Safety Mechanisms of Human-Friendly Robots: Passive Viscoelastic Trunk and Passively Movable Base." International Journal of Robotics Research, 2000: 307-335.
- Local, The. "Robot Attacked Swedish Factory Worker." The Local, Sweden's News In English. <http://www.thelocal.se/19120/20090428/> Published: 28 April 2009. Accessed: 1 February 2012.
- Lu, Shujun and Jae H Chung. "Collision Detection Enabled Weighted Path Planning: A Wrist and Base Force/Torque Sensor Approach." IEEE International Conference on Advanced Robotics. Seattle, 2005: 165-170.

- Maciejewski, A. and C. Klein. "Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments." *International Journal of Robotics Research*. 1985. 4(3):109-117.
- Marks, Paul. "Robots with skin enter our touchy-feely world: *New Scientist*." April 19, 2010. <http://www.newscientist.com/article/mg20627566.800-robots-with-skin-enter-our-touchyfeely-world.html> (accessed May 2010).
- Matsumoto, Taishi and Kazuhiro Kosuge. "Collision Detection of Manipulator Based on Adaptive Control Law." *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Como, Italy, 2001: 177-182.
- Meng, Q. and M.H. Lee. "Learning and Control in Assistive Robotics for the Elderly." *IEEE Conference on Robotics, Automation, and Mechatronics*. Singapore. 2004. Dec 1-3.
- Morinaga, Shinya and Kazuhiro Kosuge. "Collision Detection System for Manipulator Based on Adaptive Impedance Control Law." *IEEE International Conference on Robotics and Automation*. Taipei, Taiwan, 2003: 1080-1085.
- Mukhopadhyay, S.C. and G.S Gupta. "Sensors and Robotic Environment for Care of the Elderly." *IEEE International Workshop on Robotics and Sensors Environments*. Ottawa, Canada. 2007. October 12-13.
- Nathan, Alan M. "Dynamics of the Baseball—Bat Collision." *American Journal of Physics*. Vol 68, Issue 11, November 2000: 979-990.
- NIOSH Centers for Disease Control and Prevention (CDC) – The National Institute for Occupational Safety and Health (NIOSH). "Fatal Accident Summary Report—Die Cast Operator Pinned by Robot." FACE 8420. 1984.
- NIOSH Centers for Disease Control and Prevention (CDC) – The National Institute for Occupational Safety and Health (NIOSH). "Machine Operator Crushed by Robotic Platform." FACE 99NE017. 1999.
- NIOSH Centers for Disease Control and Prevention (CDC) – The National Institute for Occupational Safety and Health (NIOSH). "Mold Setter's Head Struck by a Cycling Single-side Gantry Robot." Investigation #01MI002. 2001.
- Nokata, Makoto, Koji Ikuta, and Hideki Ishii. "Safety-Optimizing Method of Human-Care Robot Design and Control." *IEEE International Conference on Robotics and Automation*, 2002: 1991-1996.
- Oberer, Susanne and Rolf Dieter Schraft. "Robot-Dummy Crash Tests for Robot Safety Assessment." *IEEE International Conference on Robotics and Automation*. April 2007: 2934-2939.
- Ogorodnikova, Olesya. "How Safe the Human-Robot Coexistence Is? Theoretical Presentation." *Acta Polytechnica Hungarica*, vol. 6, no. 4, 2009: 51-74.

- Pittman, Pete C., Torsten Staab, Dave Nelson, Bill Santistevan, and Wendel Brown. "Automation of the LANL Aries Lathe Glovebox." American Nuclear Society 9th Topical Meeting on Robotics and Remote Systems. March 4-8, 2001.
- Pryor, Mitch and Delbert Tesar. "Complex Task Completion with Redundant Serial Manipulators." *Master's Thesis*, The University of Texas at Austin, 1999.
- Pryor, Mitch. "Task-based Resource Allocation for Improving the Reusability of Redundant Manipulators." *PhD Dissertation*, The University of Texas at Austin, 2002.
- Ralph, Scott K and Dinesh K Pai. "Detection and Localization of Unmodeled Manipulator Collisions." IEEE International Conference on Intelligent Robots and Systems. Pittsburg, Pennsylvania, 1995: 504-509.
- Rethink Robotics. "Baxter's Capabilities". Accessed: 27 February 2013. <http://www.rethinkrobotics.com/index.php/products/baxter/>
- Saenz, Aaron. "A Drop-in Solution for Replacing Human Labor? Kawada's Nextage Robot." Singularity Hub. <http://singularityhub.com/2011/12/09/a-drop-in-solution-for-replacing-human-labor-kawadas-nextage-robot/> Published: 9 Dec 2011. Accessed: 20 Dec 2011.
- Schroeder, Kyle A. "On the Use of Generalized Force Data for Kinematically Controlled Manipulators." Master's Thesis. The University of Texas at Austin. 2010.
- Schroeder, Kyle A., Mitch Pryor. "On the use of Joint Torque Sensors for Collision Detection in a Confined Environment." ANS 3rd International Joint Topical Meeting on Emergency Preparedness & Response and Robotics & Remote Systems. Knoxville, TN. August, 2011.
- Schroeder, Kyle A., Mitch Pryor. "Framework for use of Generalized Force and Torque Data in Transitional Levels of Autonomy." 4th International Conference on Intelligent Robotics and Applications. Aachen, Germany. December 2011.
- Seyfferth, Wolfgang and Jorge Angeles. "A Mechanical Model for Robotic Joints with Harmonic Drives." TR-CIM-94-13, February 8, 1995. Department of Mechanical Engineering & Centre for Intelligent Machines, McGill University, Montreal, Quebec, Canada.
- Seyfferth, Wolfgang, A J Maghzal, and Jorge Angeles. "Nonlinear Modeling and Parameter Identification of Harmonic Drive Robotic Transmissions." IEEE International Conference on Robotics and Automation 1995: 3027-3032.
- Spencer, Andrew, Mitch Pryor, Chetan Kapoor, and Delbert Tesar. "Collision Avoidance Techniques for Tele-Operated and Autonomous Manipulators in Overlapping Workspaces." IEEE International Conference on Robotics and Automation, Pasadena, CA. May 19-23. 2008.

- Steinfeld, Bryan C. "Criteria Based Evaluation of Stopping Trajectories in Serial Manipulators." Master's Thesis. The University of Texas at Austin. 2009.
- Suita, Kazutsugu, Hiroyasy Ikeda, Yoji Yamada, Nuiro Tsuchida, Noboru Sugimoto, and Koji Imai. "A Failure-to-Safety 'Kyozon' System with Simple Contact Detection and Stop Capabilities for Safe Human-Autonomous Robot Coexistence." IEEE International Conference on Robotics and Automation, 1995: 3089-3096.
- Swint, Ethan Baggett. "Collision Detection and Obstacle Avoidance for Robotic Manipulators." Masters Thesis. The University of Texas at Austin, 2004.
- Taghirad, H.D. and P.R. Belanger. "An Experimental Study on Modelling and Identification of Harmonic Drive Systems." IEEE Conference on Decision and Control 1996: 4725-4730.
- Taghirad, H.D and P.R. Belanger. "A Nonlinear Model for Harmonic Drive Friction and Compliance." IEEE International Conference on Robotics and Automation 1998: 1-6.
- Takakura, Shinji, Toshiyuki Murakami, and Kouhei Ohnishi. "An Approach to Collision Detection and Recovery Motion in Industrial Robot." Annual Conference of IEEE Industrial Electronics Society. Philadelphia, Pennsylvania, 1989: 421-426.
- Tuttle, Timothy D and Warren P Seering. "A Nonlinear Model of a Harmonic Drive Gear Transmission." IEEE Transactions on Robotics and Automation, vol. 12, no. 3, June 1996: 368-374.
- Uchiyama, Masaru and Kosei Kitagaki. "Dynamic Force Sensing for High-Speed Robot Manipulation Using Kalman Filtering Techniques." Conference on Decision and Control. Tampa, Florida, 1989: 2147-2152.
- Uehara, Y., et al. "A Handbook for Safety Engineering." Corona Publishing, Tokyo. Pp. 1063. 2002. (In Japanese).
- Versace, John. "A Review of the Severity Index." SAE 710881. 15th Stapp Car Crash Conference. 1971: 771-796.
- Walker, Ian D. "Impact Configurations and Measures for Kinematically Redundant and Multiple Armed Robot Systems." IEEE Transactions on Robotics and Automation, vol. 10, no. 5, October 1994: 670-683.
- Willow Garage. "Robots | Willow Garage". Accessed: 4 March 2013. <http://www.willowgarage.com/robot/overview>
- Yamada, Yoji, Yasuhiro Hirasawa, Shengyang Huang, Yoji Umetani, and Kazutsugu Suita. "Human-Robot Contact in the Safeguarding Space." IEEE/ASME Transactions on Mechatronics, vol. 2, no. 4, December 1997: 230-236.
- Yaskawa Motoman. "SIA5D and SIA50D Robot Models Added to Motoman's SIA-Series Lineup Provide Lean, Powerful 7-Axis Arms for New Era of Automation."

- <http://www.motoman.com/motomedia/pr/SIA%20Series.pdf>. Published: March 2010. Accessed: 5 Jan 2012.
- Zheng, Yuan F and Fred R Sias. "Two Robot Arms in Assembly." IEEE International Conference on Robotics and Automation, 1986: 1230-1235.
- Zinn, Michael, Oussama Khatib, Bernard Roth, and J. Kenneth Salisbury. "Playing it Safe." IEEE Robotics and Automation Magazine, June 2004: 12-21.
- Zollo, Loredana, Bruno Siciliano, Cecilia Laschi, Giancarlo Teti, and Paolo Dario. "Experimental Comparative Evaluation of Compliant Control Schemes for an Anthropomorphic Personal Robot." American Control Conference, Anchorage, AK. May 8-10, 2002.