September 2015

Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices

Yu-Cheng Wang
University of South Florida, yucheng@mail.usf.edu

Scholar Commons Citation

Wang, Yu-Cheng, "Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices" (2015). *Graduate Theses and Dissertations.*
http://scholarcommons.usf.edu/etd/5793

Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices

by

Yu-Cheng Wang

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Mechanical Engineering
Department of Mechanical Engineering
College of Engineering
University of South Florida

Major Professor: Redwan Alqasemi, Ph.D.
Rajiv Dubey, Ph.D.
Kyle Reed, Ph.D.

Date of Approval:
July 2, 2015

Keywords: Redundancy, Manipulator,
Robot, Motion-planning, Force Feedback

ACKNOWLEDGMENTS

I am very grateful to my major professor, Dr. Redwan Alqasemi, for letting me enter the lab and working with him. In the past two years, he gave me a lot of useful suggestions, and shared his knowledge with me generously.

I would like to thank my committees, Dr. Rajiv Dubey and Dr. Kyle Reed. Because of Dr. Rajiv Dubey's suggestion, I had an opportunity to connect with Dr. Redwan Alqasemi, and entered CARRT. Furthermore, Dr. Kyle Reed influenced me in his Haptics course. His teaching made me having the decision to pursue master degree with thesis.

In addition, I want to thank the members in the lab, Andoni Aguirrezabal, Daniel Ashley, Indika Pathirage, Lei Wu, Shangdong Gong, Karan Khokar, Paul Mitzlaff, Mustafa Mashali, and Tianyu Hong. Because of your help, I was able to finish the thesis. I really had a great time with all of you while we were working.

My special thanks go to my friends in Taiwan, Korea, and America. You let me know that I was not here alone. You will always stay with me.

Last but not least, I would like to express my greatest thanks to my parents and family. With your love, I could complete the work without any anxiety. I will always love you same as how you treat me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A teleoperated system of dual redundant manipulator will be controlled in this thesis. The robot used with the dual redundant manipulator in this thesis is Baxter. Baxter's redundant robot arms are 7-degree-of-freedom arms. The problem that will be solved in this thesis is optimization of the 7-degree-of-freedom robot arms. The control algorithm of the 7-degree-of-freedom robot arms will be discussed and built. A simulation program will be built to test the control algorithm. Based on the control algorithm, a teleoperation system will be created for Baxter. The controller used is Omni, which is a six-joint haptic device. Omni will also be used to give force feedback upon collision while the user is controlling the robot. Hence, a collision force feedback system is going to be created and combined with the teleoperation system. The teleoperation system will be tested in common daily applications.

CHAPTER 1: INTRODUCTION

1.1 Motivation

Imagine there is a robot that can perform arm-like motions while being controlled by a person from a different location, there may be some benefits for that person. For instance, the robot can enter dangerous environments to complete tasks instead of the person [1]. Now imagine a more probable scenario, the robot is trying to complete a complex task automatically. If the pre-programmed robot cannot implement the work, a human is needed to control it [2]. Hence, a reliable remote controlled robotic system is necessary to be built to represent a human.

In order to represent a human, the robot's arms should have the same manipulability as a human. This robot should have at least two arms, both 7 degrees of freedom or more. With dual arms, the robot can work effectively in more tasks than a robot with a single arm, such as holding a tray and opening a bottle. And 7-degrees-of-freedom arms allow the robot to finish many tasks without redesigning the kinematics of the arms because a 7-degree-of-freedom arm can achieve more poses. A teleoperated robotic system with dual arm can assist in accomplishing the above task. A 7-degree-of-freedom arm is redundant. Hence, how to optimize its kinematics is an important topic. Additionally, combining teleoperation system with some devices can improve users' experience, such as a haptic device [16], and depth sensor camera [11].

This thesis is going to discuss the control method of a dual 7-degree-of-freedom manipulator and its optimization. Based on the control method, a motion planning system of the manipulator will be built. Motion planning in this thesis is defined as the motion of the robot's joints after considering the commanded Cartesian motion and the given joint limits. The motion planning system computes changes of joints of the manipulator when the end-effector of the manipulator moves from one point to other points. Then, with our input velocities of the end-effector from control device and communication program, the motion planning system can send joint commands to the robot for achieving desired positions, and the robot can also return its current joint angles to the motion planning system. In this way a reliable teleoperation control system can be built. Finally, we will combine the system with force feedback for a better user experience. The platform used in this thesis is a Baxter robot, which is made by Rethink Robotics [30]. This robot has dual 7-degree-of-freedom arm. In the end, a remote controlled Baxter system will be accomplished, and can be controlled to complete some daily activities. The system will be tested with human subjects, and statistical analysis will be performed to find the effectiveness of the control system.

1.2 Thesis Objectives

1. The first objective for this thesis is to develop a control system for the redundant arms of Baxter to move the end-effector of the arm from one point to other points.
2. A simulation program of the arms will be built and tested with the control system.
3. The redundant control system will be optimized to avoid singularities and joint limits by using the simulation program.

4. A communication program will be established in order to interchange information of joint angles between the motion planning system and the physical robot.
5. A motion planning system will be built and tested on the physical platform to confirm accuracy of the control system through the communication program.
6. By adding a velocity input device into the motion planning system, velocities of the end-effector will be obtained.
7. Combining the motion planning system, the velocities input device, and the communication program, a teleoperation system will be created on the platform.
8. A user interface will be developed.
9. A collision force feedback program will be built, and will be combined with the teleoperation system.
10. Human subject testing will be performed, and data will be collected and analyzed.

1.3 Thesis Outline

For this thesis, the background of works that have been done previously in the field of teleoperation, dual redundant arm control, and force feedback application will be introduce in chapter 2. Chapter 3 will discuss the redundant arms control method and its optimization. Chapter 4 will focus on the simulation development with the control method. Chapter 5 will discuss the teleoperation system, and apply the control method on the teleoperation system with collision force feedback. Finally, Chapter 6 will conclude the thesis and discuss future work.

CHAPTER 2: BACKGROUND

2.1 Teleoperation

Why do humans need telerobotics? Sheridan [2] has mentioned that there are many unpredictable tasks that pre-programmable robotics cannot implement; moreover, many working environments are unsuitable for humans. However, comparing human capability with robot's capability is meaningless. Hence, it is more important to study how they can collaborate. In his paper, Sheridan has defined teleoperation as a means to allow a user to execute their own senses and manipulation capability on a remote-controlled device, which also includes telerobotics. Telerobotics technology has been used in various applications, as stated in Sheridan's papers [1] and [2], such as space, undersea, toxic waste cleanup, nuclear power, telediagnosis, telesurgery, aids for the handicapped, terrestrial mining, construction and maintenance, warehousing, firefighting, policing, and military operations.

Niemeyer et al [3] proposed the concept of a teleoperation system, shown in figure 2.1, which can be divided into five parts: human operator, master system, communications, telerobot, and environment. Nevertheless, the communication part causes time delay problems between the human operator and the telerobot. Goto et al [4] have combined visual servo control technique and remote control system in the loop to correct user's desire position. The result shows that because of the improvement of the handling ability of the robot arm motion with visual servo control in the loop, the influence of the communication problem is reduced. Lai et al [5] has implemented an "Adaptive Smith

Predictor" to compensate for the time delay. The experiment in this paper is to remote control a 400W servo motor from 15 km away. The results show that the Adaptive Smith Predictor improves the stability of the motor, but causes a larger overshoot problem.

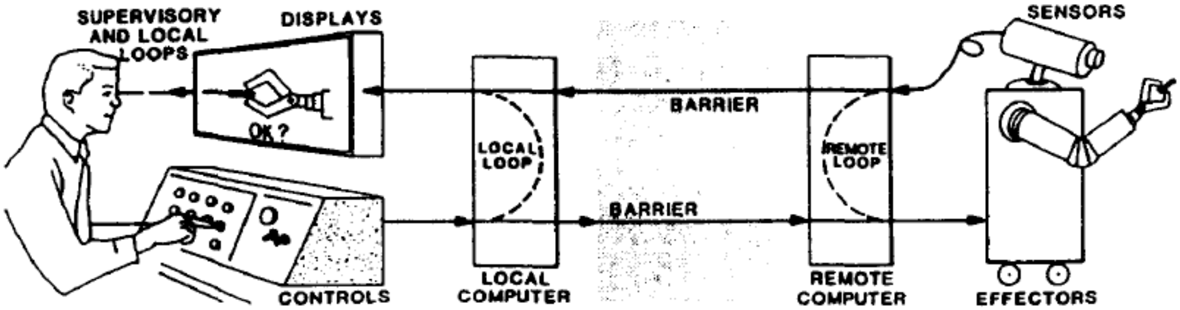


Figure 2.1: Concept of Teleoperation System.

2.2 Dual Arm Control

The researchers at the Royal Institute of Technology [6] have proposed two classifications of dual arm manipulation. The first classification is non-coordinated control, which mean both arms act on two different works, and can be regarded as two single arms. In the second categorization, coordinated control, two arms work on different parts of the same work. According to the existence of physical interaction in coordinated control, the authors have further classified it into bimanual manipulation, with physical interaction, and goal coordinated manipulation, without physical interaction.

Krüger et al [7] have proposed the development of an industrial dual arm robot. They focused on a bimanual manipulation task, and presented an algorithm for compliant motion control. Using impedance control with position control and closed loop environmental transfer functions can have better performance with dual arm manipulation than two single arm controls. Wimböck et al [9] also executed impedance control of both arms in the same system to coordinate in the same task.

The researchers at the California Institute of Technology [8], figure 2.2, and Chiba University [10] have respectively combined dual arm control with a visual system to estimate the robotic arms sate and object state for improving task performance.

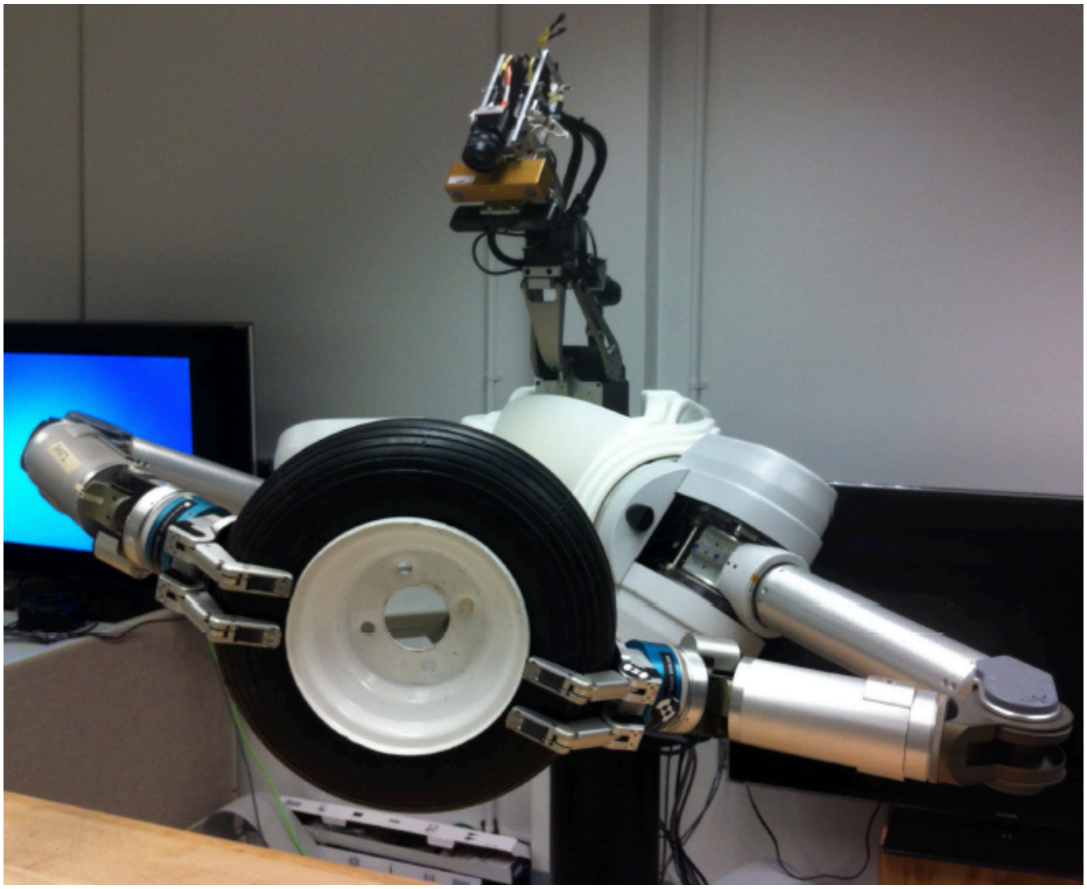


Figure 2.2: A Dual Arm Robot in Gripping Wheel Experiment with Visual System.

2.3 Application of Kinect on Robotics

The Kinect has become a well-known and popular 3D sensor device because of its versatility, competitive price, high efficiency and accurate results [11]. It is composed of one near-infrared laser pattern projector, an IR camera and an RGB camera, as seen in figure 2.3. The RGB camera can show color and pattern on top of the 3D image information. Moreover, the near-infrared laser pattern projector and IR camera gives a set of depth image information; hence, the Kinect can offer a color image with depth information. By using depth information, the user can obtain the distance between the Kinect and any objects shown in the image. Therefore, with those characteristics, Kinect is widely used in the robotics domain.

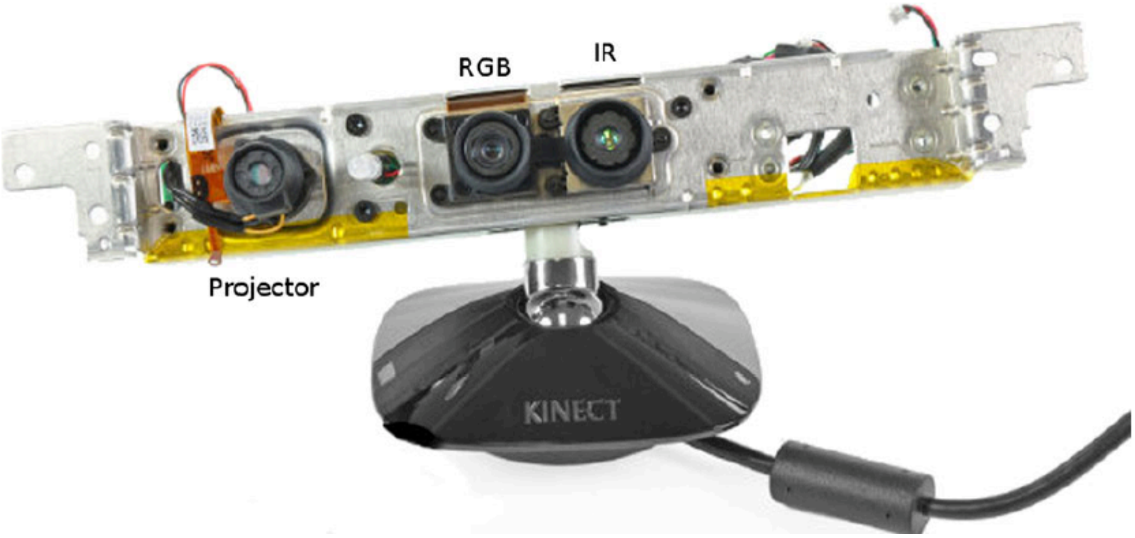


Figure 2.3: The RGB Camera, IR Projector, and IR Camera on the Kinect.

Rakprayoon et al [12] used a Kinect to recognize the manipulator and obstacles when obstacles are in the working space of the manipulator. The authors converted a 3D model of the manipulator to 3D point cloud in the scene. Then, they used least-square method to eliminate the manipulator data points from Kinect coordinates, as shown in Figure 2.4.

Flacco et al [13] has proposed a real time collision avoidance method with the Kinect. Using the depth data, one can estimate the distance between the robot and moving obstacles, as shown figure 2.5. Next, they computed the direction of the obstacle repulsive vector to avoid obstacle collision in their motion task. In addition, Leeper at al [14] [32] have also created real-time environment models using the Kinect sensor to avoid environment collisions in their Constraint-Aware Teleoperation system, shown in figure 2.6. Because it takes time to estimate a collision avoidance task, their joint velocity is limited.

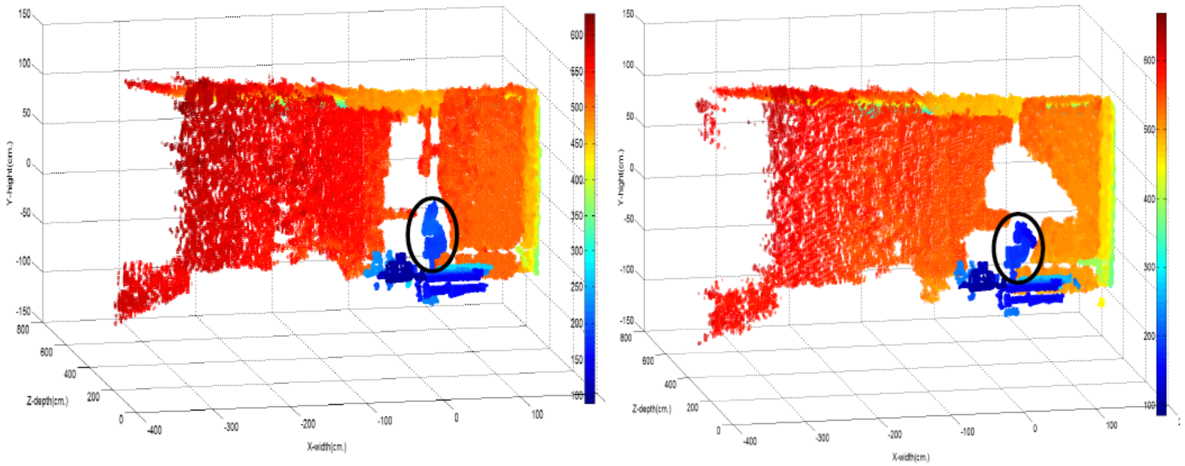


Figure 2.4: The Point Cloud of Manipulator from Kinect in the Scene.

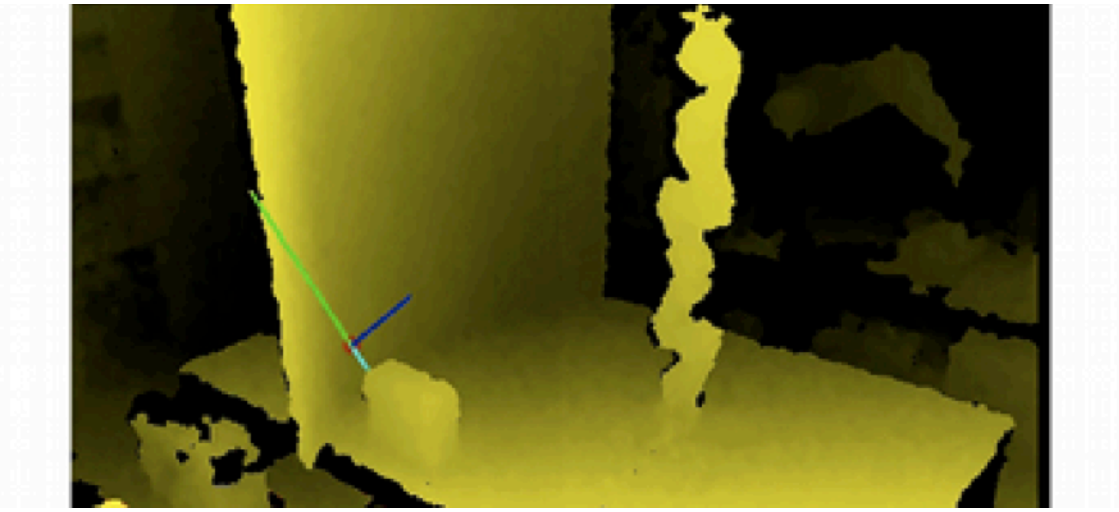


Figure 2.5: The Intensity of Colors Refer to Distance of Obstacle Points.

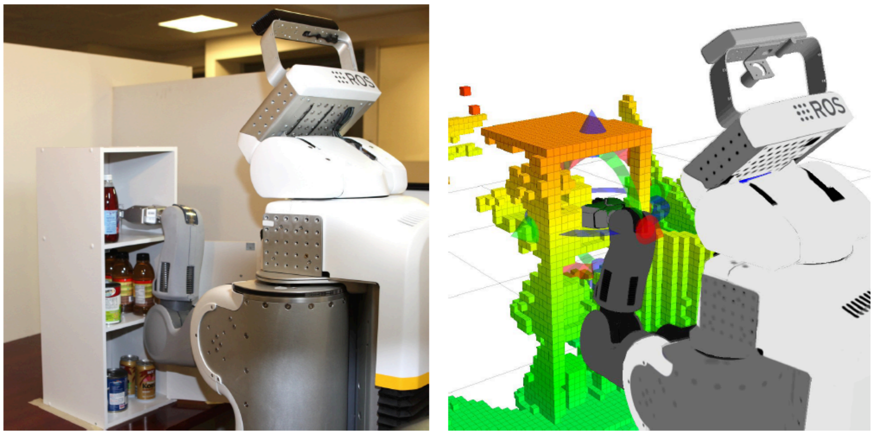


Figure 2.6: A Teleoperated Robot in a Cluttered Environment, and Its Virtual Scene.

2.4 Haptic Teleoperation

Haptics help the user to feel the virtual environment or remote scene [16]. Interactions can be performed with force feedback on the master device. In other words, it can improve the user's remote control experience by adding force feedback from the slave robot with the audio-visual feedback, figure 2.7. Hence, one of the challenges for teleoperation haptics is how to design a local force interface and estimate the local force from the remote environment to the slave robot [17].

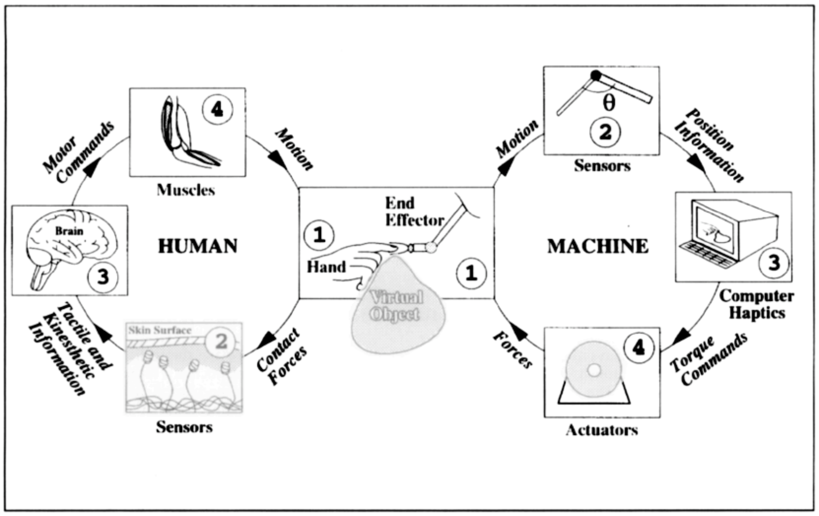


Figure 2.7: Haptic Interaction between Humans and Machines.

Park et al [15] have proposed a teleoperation interface to compute the local force and feedback force. There are three parts of the system. The first part is a virtual spring between the master and slave devices. The second part is a motion control system, and the third is a contact force sensor on the slave device. By using the above parts, the users can remote control the end-effector of a robotic arm via a haptic device with effective force feedback.

Nadrag et al [18] provided the user with force feedback from a slave device, which is a Lina robot, to improve operator grasp. Lina is a mobile wheels robot, shown in figure 2.8, with 12 ultrasonic sensors positioned around the robot and a laser range finder. In their experiment, only seven ultrasonic sensors were used. Lina was tested in various walled environments with and without force feedback. The result showed that force feedback does not always have an advantage in various situations.

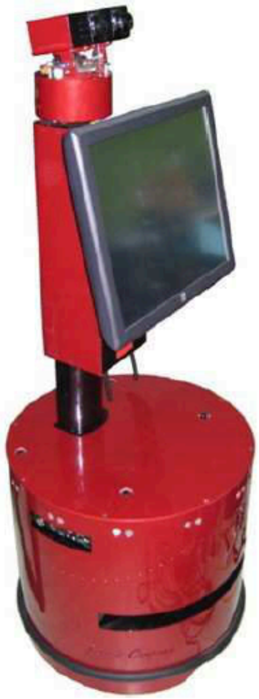


Figure 2.8: The Lina Robot.

Leeper at al [19] have also designed a haptic algorithm using the distance between a virtual gripper's surface and a virtual environment, which is built by a 3D sensor, to avoid collisions. The user remote controls the virtual gripper through a haptic device or mouse on the virtual environment screen, shown in figure 2.9. While the user is creating a path of motion on the virtual gripper, the haptic algorithm allows him or her to feel the force of the virtual scene. After the user confirmed the motion of the end-effector on the virtual scene, the robot would execute the confirmed trajectory. The results of their works show that in a 10 minute period, users grasped more objects with haptic feedback than without haptic feedback.

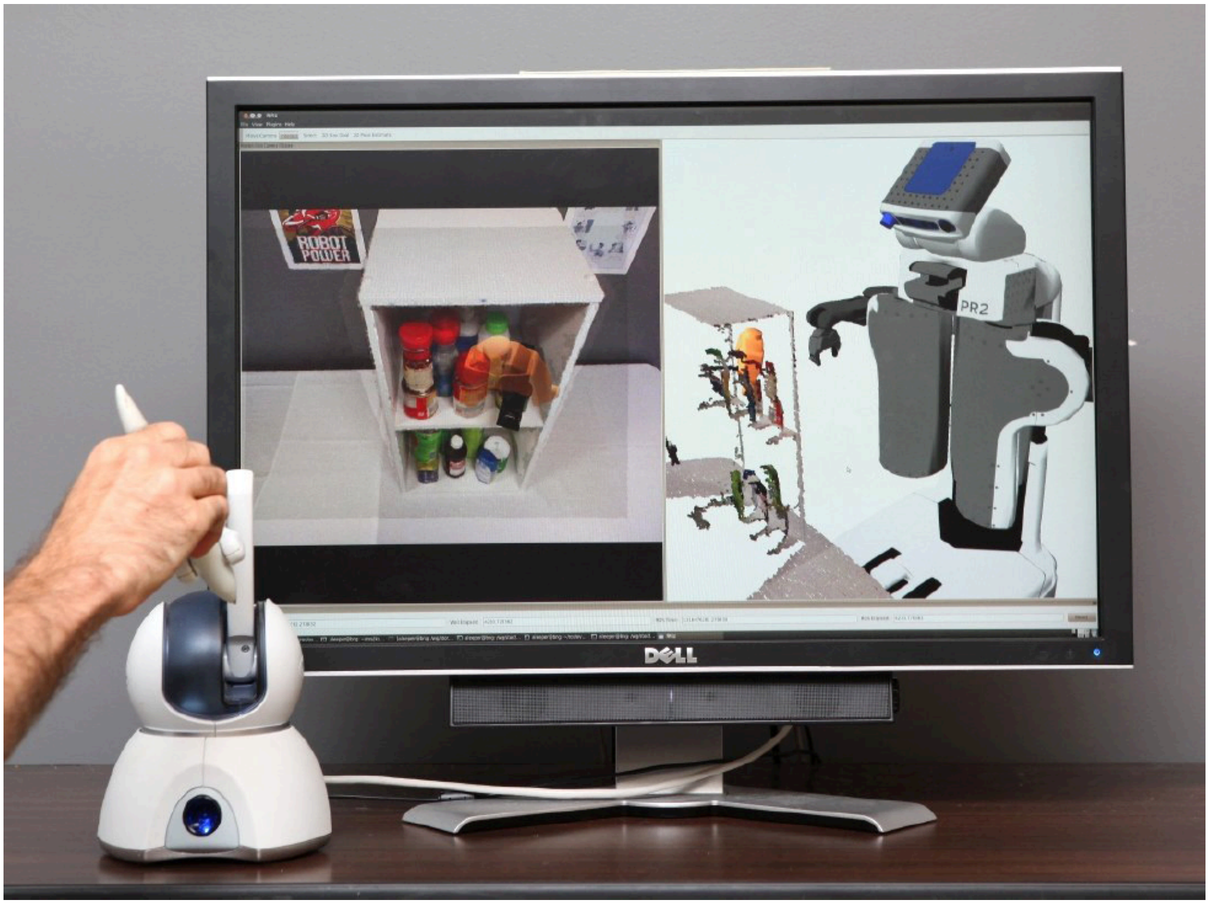


Figure 2.9: The Haptic Device and the Virtual Environment on Screen.

2.5 Redundant Robot Control

In Cartesian space, there are 6 degrees of freedom (DOF), which consist of 3 degrees of freedom for translation and 3 degrees of freedom for rotation. These can be used to describe the movement of a rigid body, such as aircraft, in 3D space. Likewise, the movement of the end-effector of a robot arm can also be described as degrees of freedom in 3D space. Hence, if an end-effector is required to match any particular pose in the working space, the robot arm should be designed with at least 6 degrees of freedom [20]. And then, if a robot arm has more than 6 degrees of freedom, it is call redundancy, which can provide infinite kinematic solutions for performing one end-effector movement. However, even though a redundant robot arm can offer infinite paths for motion planning as a significant benefit, the traditional inverse kinematics is not fit to solve kinematic problems with redundancy because the normal Jacobin matrix size is not suitable to resolve inverse manipulator kinematics. Moreover, singularity points appear in some poses of the robot arm, which can cause singularity problems in resolving inverse kinematics.

Klein et al [21] have discussed a suggestion about joint velocities of redundant kinematics with pseudoinverses, which can also easily use the minimum norm solution to minimize velocity, which can reduce energy consumption. With minimized velocity, we can avoid singularities by limiting the joint velocity when the joint is nearby the singularities. However, the author noted pseudoinverses couldn’t provide joint angle limits. The authors suggested adding maximum and minimum joint stop conditions, separately. Lee et al [22], Huber et al [23], and Leeper et al [19] used the same method to resolve and optimize kinematic redundancy on dual redundant arm manipulators, aerial redundant robot arm manipulation, shown in figure 2.10, and PR2 robot control, respectively.

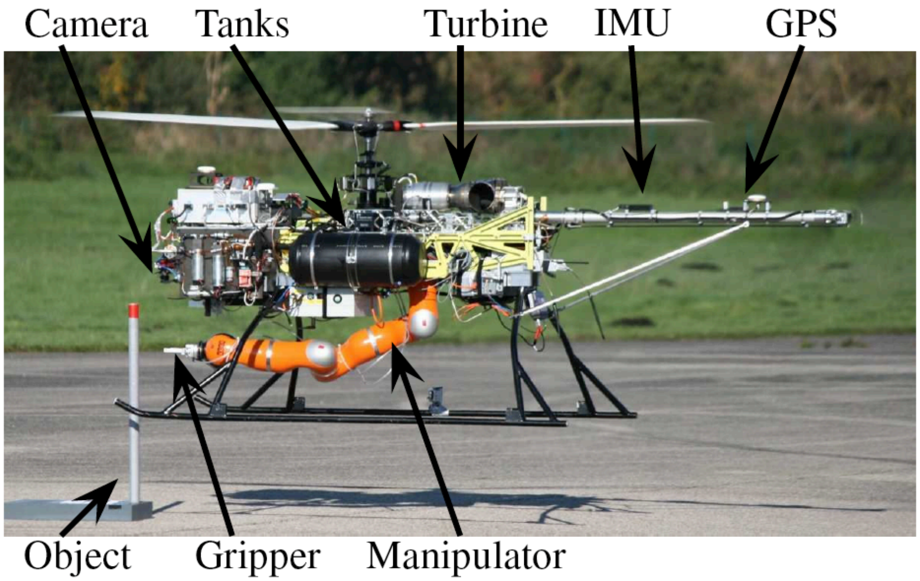


Figure 2.10: A Redundant Industrial Robot Arm on an Aerial Platform.

Chan et al [24] have presented a method to avoid joint angle limit and resolve kinematic redundancy with pseudoinverses. In this method, they added a weighted least-norm solution into pseudoinverse. Then, they treated the weighted matrix as a diagonal matrix, and offered a criteria function related to the upper and lower joint limits. This criteria function would restrain the joint motion if the joint angle is close to the joint angle limits. With the added function, the kinematic redundancy can be resolved, and the joint limits can be avoided.

CHAPTER 3: CONTROL METHOD OF REDUNDANT ROBOTIC ARM

3.1 Introduction

For a healthy person, it is intuitive to grasp an object and place it somewhere. However, for a robot arm, there are a series of computations necessary to accomplish the motion planning. First, the pose and position of the end-effector of the robot arm have to be calculated based on each joint angle and link length in the system. Then, we can calculate the relative position and orientation between the object and base frame of a robot in Cartesian coordinates. After that, the relation between the object and end-effector can be built with a mathematical method in order to create the trajectory for going through on each waypoint. Then, all the joint angels should be obtained with the desired position to achieve the object. Because there are infinite kinematic solutions for redundant manipulators (more than 6 degrees of freedom), the necessary resolution and optimization of a redundant manipulator will be problematic to perform. In this chapter, the control method from manipulator kinematics to inverse manipulator kinematics and redundant robotic arm optimization will be introduced. In this research, the robot platform will be the Baxter from Rethink Robotics [30], which has dual 7-degree-of-freedom arms.

3.2 Description of Link

A link chain of a robot arm is composed of both joints and links. Before the position and orientation of the end-effector from base frame can be computed, each link frame should be properly attached to every link and joint. Craig [20] has introduced the description of link frames and link-joint parameters Figure 3.1 shows the relation between every neighboring link and joint with link-joint parameters and link frames.

With the link-joint parameters, the Denavit-Hartenberg parameters (D-H parameter) can be obtained according to Craig's notation. Figure 3.2 and figure 3.3 show link-joint parameters (D-H parameters) and frame definitions of the right and left arms of Baxter robot, respectively. The DH parameters are defined in tables 3.1 and 3.2. The picture of Baxter is shown in figure 3.4.

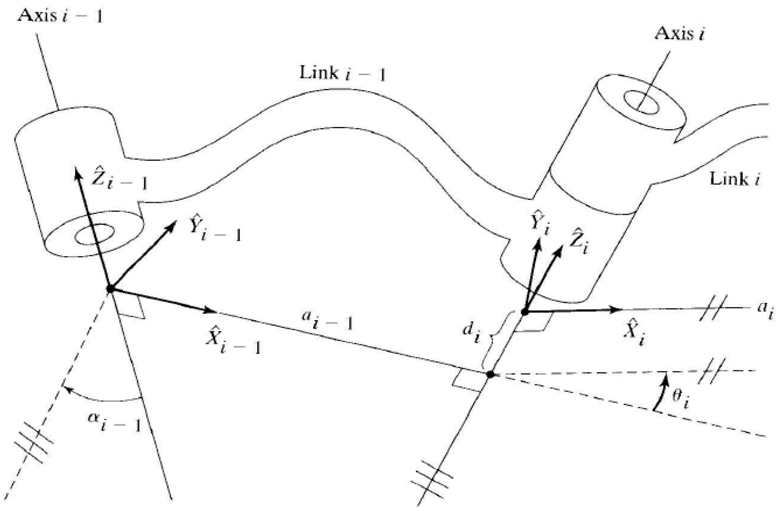


Figure 3.1: Link Frame and Joint Parameters.

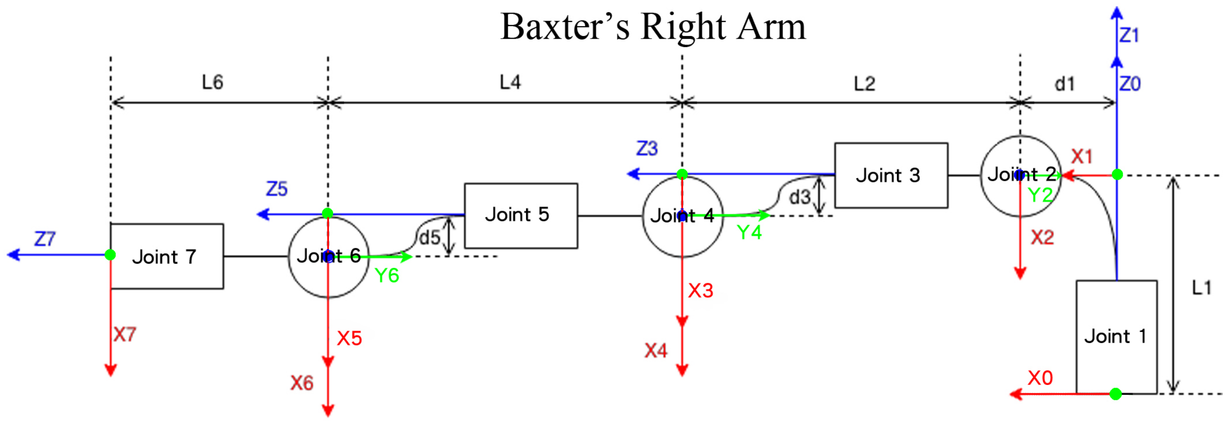


Figure 3.2: Link Parameters and Frames of Baxter's Right Arms.

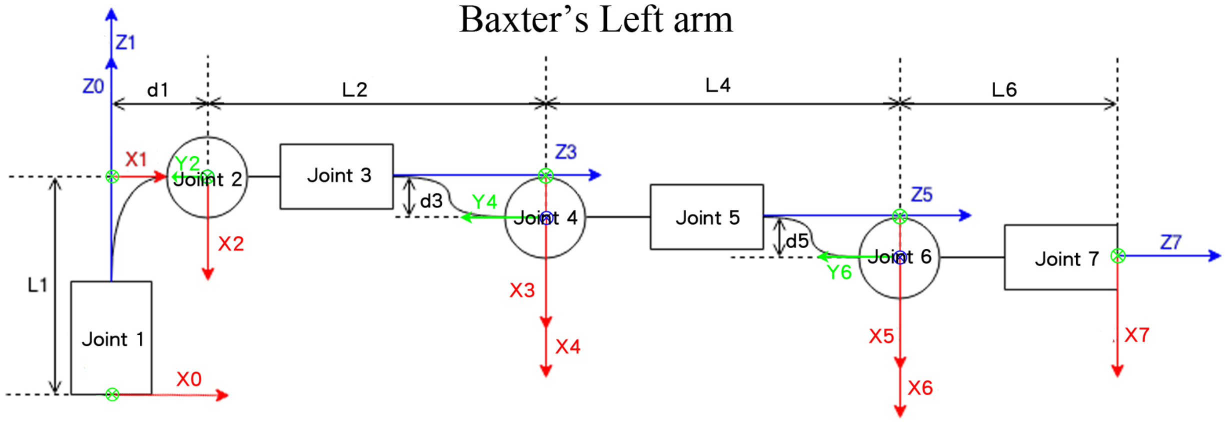


Figure 3.3: Link Parameters and Frames of Baxter's Left Arms.

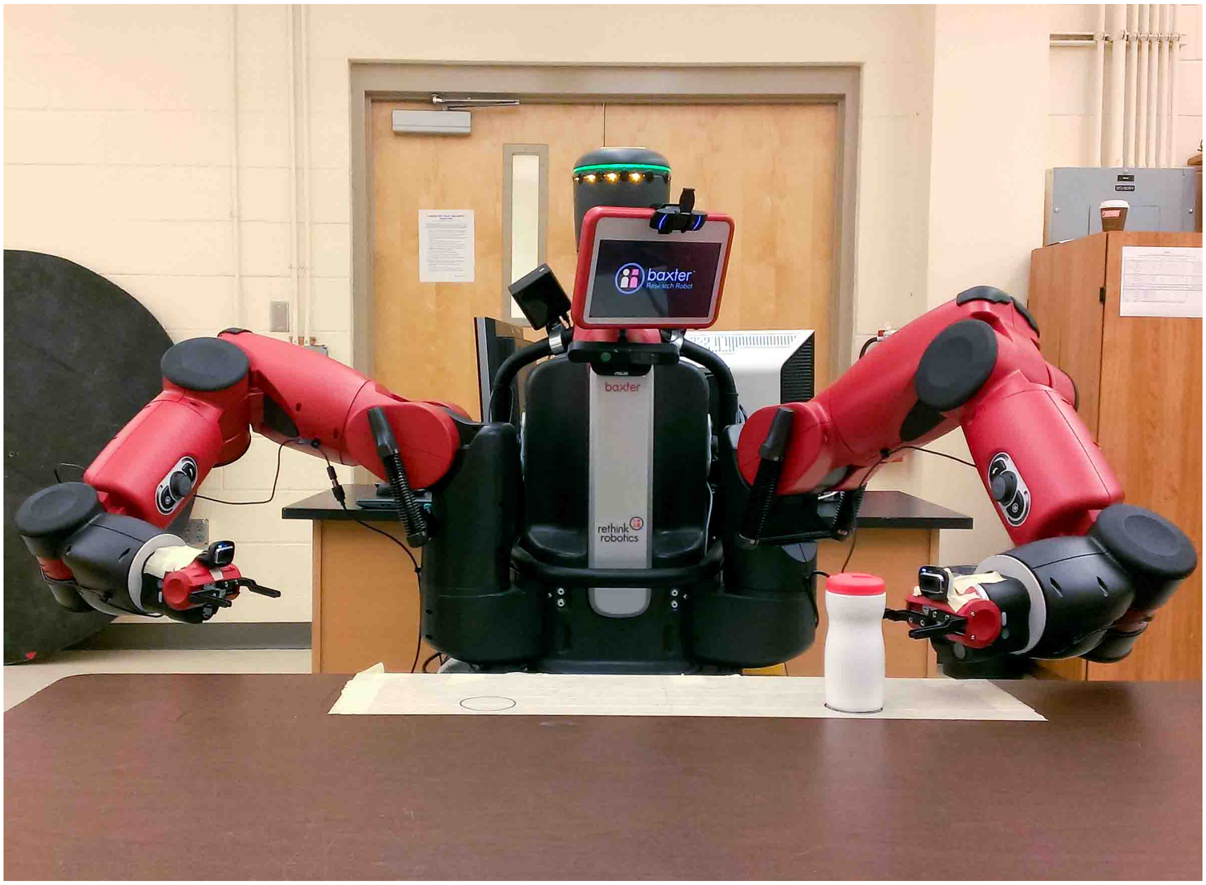


Figure 3.4: Baxter Is Gripping a Bottle.

Table 3.1: D-H Parameter of Baxter’s Right Arm.

i	α_i	a_i	d_i	θ_i
1	$0°$	0	L_1	θ_1
2	$-90°$	d_1	0	θ_2
3	$90°$	0	L_2	θ_3
4	$-90°$	d_3	0	θ_4
5	$90°$	0	L_4	θ_5
6	$-90°$	d_5	0	θ_6
7	$90°$	0	L_6	θ_7

Table 3.2: D-H Parameter of Baxter’s Left Arm.

i	α_i	a_i	d_i	θ_i
1	$0°$	0	L_1	θ_1
2	$-90°$	d_1	0	θ_2
3	$90°$	0	L_2	θ_3
4	$-90°$	d_3	0	θ_4
5	$90°$	0	L_4	θ_5
6	$-90°$	d_5	0	θ_6
7	$90°$	0	L_6	θ_7

3.3 Manipulator Kinematics

D-H parameters show the relation between every neighboring link and joint, where only the joint angles are variables in the system. Hence, after knowing every joint angle of a robot arm, the position and orientation of the end-effector can be acquired with forward kinematics by a homogenous transformation matrix [20], which shows the relation between two frames. The following equation is a form of transformation matrix from the DH parameters:

$$ {}^{i-1}_{\;\;i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1} $$

where c is cosine and s is sine. In the homogenous transformation matrix, the first three rows and columns are a 3x3 rotation matrix that is related to row, pitch, and yaw of frame $\{i\}$ relative to frame $\{i-1\}$. Furthermore, the relative position matrix on x, y, and z axes of the Cartesian coordinates are the first three elements in the fourth column of the homogenous transformation matrix. With the D-H parameter in table 3.1, the homogenous transformation matrices between each neighboring frames of the right arm of Baxter can be laid out as follows:

$$ \begin{aligned} {}^{0}_{1}T_r &= \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & {}^{1}_{2}T_r &= \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & d_1 \\ 0 & 0 & 1 & 0 \\ -s\theta_2 & -c\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\ {}^{2}_{3}T_r &= \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & 0 \\ 0 & 0 & -1 & -L_2 \\ s\theta_3 & c\theta_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & {}^{3}_{4}T_r &= \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 & d_3 \\ 0 & 0 & 1 & 0 \\ -s\theta_4 & -c\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \end{aligned} \tag{3.2} $$

$${}^{4}_{5}T_r = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ 0 & 0 & -1 & -L_4 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad {}^{5}_{6}T_r = \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & d_5 \\ 0 & 0 & 1 & 0 \\ -s\theta_6 & -c\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$${}^{6}_{7}T_r = \begin{bmatrix} c\theta_7 & -s\theta_7 & 0 & 0 \\ 0 & 0 & -1 & -L_6 \\ s\theta_7 & c\theta_7 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Because the left arm of Baxter is same as the right arm, the transformation matrices of the left arm is equal to the transformation matrices of the right arm.

3.4 Relation of Transformation Matrix

In equation 3.2, the homogenous transformation matrix of frame {2} relative to frame {0}, ${}^{0}_{2}T$, can be obtained by multiplying ${}^{0}_{1}T$ and ${}^{1}_{2}T$. By the same logic, the homogenous transformation matrix relates frame {0} and frame {7}, ${}^{0}_{7}T$, is:

$${}^{0}_{7}T = {}^{0}_{1}T \cdot {}^{1}_{2}T \cdot {}^{2}_{3}T \cdot {}^{3}_{4}T \cdot {}^{4}_{5}T \cdot {}^{5}_{6}T \cdot {}^{6}_{7}T \tag{3.3}$$

Then, the relative position and orientation of frame {7} based on frame {0} in Cartesian coordinates can be obtained from equation 3.3. The relation between the base frame and frame {0} of both arms should be calculated in order to obtain the correspondent Cartesian position and orientation relative to the base frame. The relation between the base frame and frame {0} of both arms can be found in figure 3.5 The transformation matrices between frame {0} of the right arm and the base frame is ${}_{base}{}^{0}T_r$, and between zero frame of left arm and base frame is ${}_{base}{}^{0}T_l$.

$${}^{base}_{0}T_r = \begin{bmatrix} 1 & 0 & 0 & 6.4027 \\ 0 & 1 & 0 & -25.9027 \\ 0 & 0 & 1 & 11.8588 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{base}_{0}T_l = \begin{bmatrix} 1 & 0 & 0 & 6.4027 \\ 0 & 1 & 0 & 25.9027 \\ 0 & 0 & 1 & 11.8588 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

Then, the transformation matrices from the base to the end-effectors of both arms are:

$$ {}^{base}_{7}T_r = {}^{base}_{0}T_r \cdot {}^{0}_{7}T_r \;,\; {}^{base}_{7}T_l = {}^{base}_{0}T_l \cdot {}^{0}_{7}T_l \tag{3.5} $$

Moreover, if there is an object, and the homogenous transformation matrix between the object and the base frame $\{base\}$, ${}^{base}_{object}T$, is known, the relative homogenous transformation matrix, ${}_{object}^{7}T$, can be easily computed as:

$$ {}_{object}^{7}T = {}_{base}^{7}T \cdot {}^{base}_{object}T = {}^{base}_{7}T^{-1} \cdot {}^{base}_{object}T \tag{3.6} $$

By using this relation, the homogenous transformation matrix between the end effector and any object frame can be easily obtained.

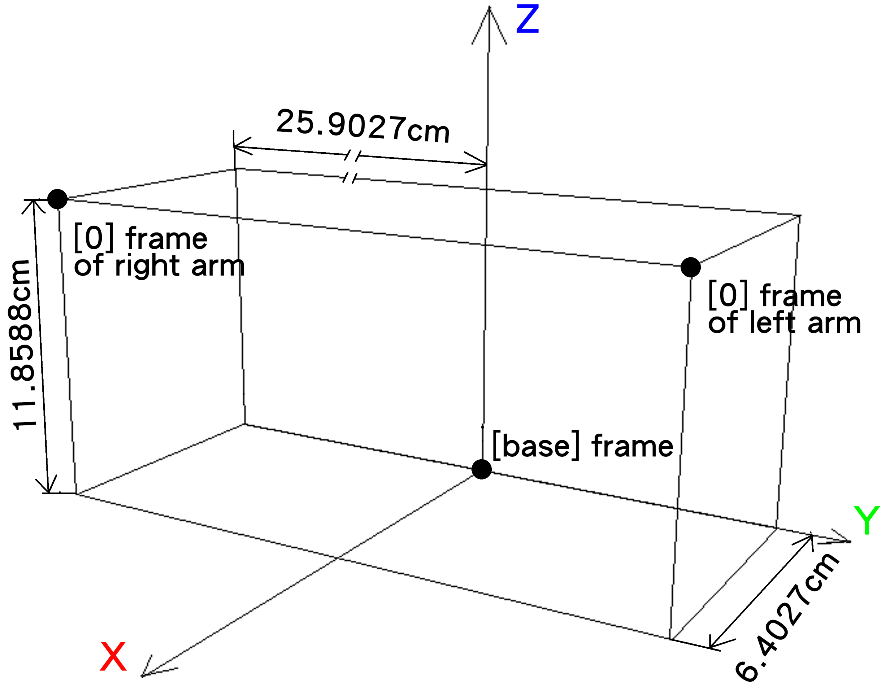


Figure 3.5: Relation of Base Frame and Frame {0}.

Additionally, the difference in the Cartesian position between two homogenous transformation matrices is easy to compute, by subtracting the two position vectors from the homogenous transformation matrices. But the difference of Cartesian orientation is not obvious from the homogenous transformation matrices. The orientation error, e_o, between two homogeneous transformation matrices can be used to obtain the difference of Cartesian orientation. The definition of the orientation error as defined in [26] is:

$$e_o = \frac{1}{2}(n \times n_d + s \times s_d + a \times a_d) \tag{3.7}$$

where n, s, a are each column of the current rotation matrices, n_d, s_d, a_d are each column of the desire rotation matrices. With the error of both position and orientation, a linear trajectory can be calculated.

3.5 Inverse Manipulator Kinematics and Jacobians

If the joint angles are known, the position and orientation of the end-effector of a robot arm can be computed by forward kinematics. However, in the real world, the desired position and orientation, or the desired velocities of the end-effector are usually given for the manipulators, and the proper combination of joint angles or velocities are desired to allow the end-effector to reach the desired point if necessary. Hence, inverse kinematics of manipulators can be used to assist the controller to read and command the joint velocities to achieve the desired point.

Because the linear velocity (v) and angular velocity (w) of neighboring links in a manipulator are relative to each other, the velocity of each link can be computed [20][29]. Figure 3.6 shows the velocity vectors of neighboring links. The relationship for v vectors and w vectors between i and $i + 1$ can be defined as:

$${}^{i+1}w_{i+1} = {}^{i+1}_{\ \ i}R\ {}^{i}w_i + \dot{\theta}_{i+1}\,{}^{i+1}Z_{i+1} \tag{3.8}$$

$${}^{i+1}v_{i+1} = {}^{i+1}_{\ \ i}R(\ {}^{i}v_i +\ {}^{i}w_i \times\ {}^{i}P_{i+1}) \tag{3.9}$$

Here, ${}^{i}w_i$ is the angular velocity for link i relative to frame i, ${}^{i}v_i$ is the velocity of link i relative to frame i, $\dot{\theta}_i$ is joint angular velocity for joint i, ${}^{i+1}_{\ i}R$ is a rotation matrix of frame i relative to frame $i+1$, and ${}^{i}P_{i+1}$ is a position vector of frame $i+1$ relative to frame i .

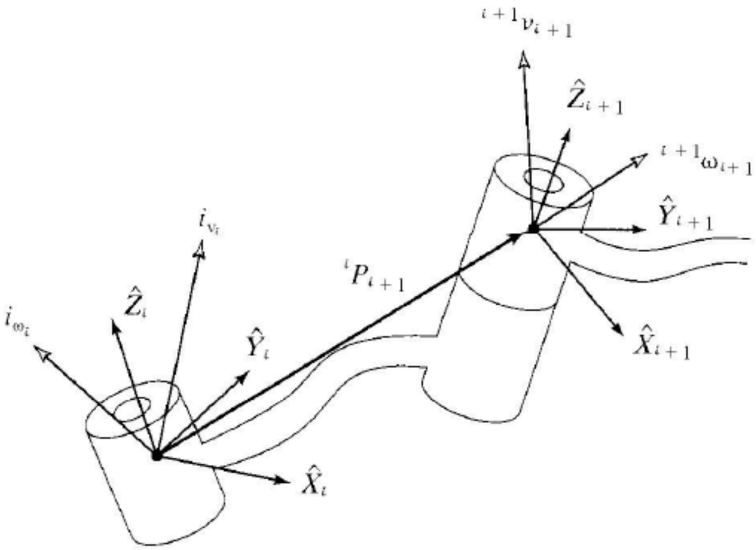


Figure 3.6: Velocity Vectors on Link i and Neighboring Links.

In this research, there are seven joints on each manipulator, so each velocity function can be written with seven independent variables:

$$Y = F(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7) \tag{3.10}$$

where Y is a 6×1 vector, which includes three position and three orientation vectors of the end-effector. Vectors of the end-effector are based on the base frame in the Cartesian system. The vectors can be written as below:

$$\begin{aligned} \delta x &= \frac{\delta f_1}{\delta\theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta\theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta\theta_7}\delta\theta_7, \\ \delta y &= \frac{\delta f_1}{\delta\theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta\theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta\theta_7}\delta\theta_7, \end{aligned} \tag{3.11}$$

$$\begin{aligned}
\delta z &= \frac{\delta f_1}{\delta \theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta \theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta \theta_7}\delta\theta_7, \\
\delta \alpha &= \frac{\delta f_1}{\delta \theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta \theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta \theta_7}\delta\theta_7, \\
\delta \beta &= \frac{\delta f_1}{\delta \theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta \theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta \theta_7}\delta\theta_7, \\
\delta \gamma &= \frac{\delta f_1}{\delta \theta_1}\delta\theta_1 + \frac{\delta f_2}{\delta \theta_2}\delta\theta_2 + \cdots + \frac{\delta f_7}{\delta \theta_7}\delta\theta_7
\end{aligned} \tag{3.11}$$

which can be also described as:

$$\delta X = \frac{\delta F}{\delta \theta} \cdot \delta\theta \tag{3.12}$$

The derivative 6×6 matrix in equation 3.12 is called the "Jacobians matrix", which is a multidimensional matrix. In robot manipulators, a Jacobian matrix defines the relation between Cartesian velocities and joint rates as:

$$V = J(\theta) \cdot \dot{\theta} \tag{3.13}$$

where V are the Cartesian linear and angular velocities, $J(\theta)$ is the Jacobian matrix, and $\dot{\theta}$ are the joint velocities. The V and $J\ (\theta)$ are based on the same frame. On the other hand, by knowing the inverse Jacobian matrix and Cartesian velocities, the joint rates can be calculated, as shown in equation 3.14.

$$\dot{\theta} = J^{-1}(\theta) \cdot V \tag{3.14}$$

However, there are some singularity points in the workspace. Singularities occur when the manipulator of a robot arm is fully stretched out or folded back and the end-effector is near the boundary of the workspace, or when two or more joint axes are lined up. At those moments, the determinant of Jacobian is zero, which causes inverse Jacobian to be infinite. Looking at equation 3.14, when the inverse Jacobian is infinite, joint velocities are also infinite. High joint velocities will cause the robot arm to vibrate and go out of control. Yoshikawa [25] has proposed a manipulability measure:

$$w = \sqrt{\det\left[J(\theta) \cdot J^T(\theta)\right]} \tag{3.15}$$

The Manipulability measure, w, is a scalar value, which becomes zero when singularity occurs, and increases as the manipulator configuration moves away from singular configurations. Hence, w can be considered to be a measure of how far is the manipulator from singular configurations.

3.6 Redundant Arm Optimization

A redundant arm has its benefits and disadvantages. As for the benefits, a redundant arm has more poses available for a given position than non-redundant robot arm. In other words, the redundant arm can reach the fixed orientation and position with more options, such as options for obstacle avoidance. However, the Jacobian matrix for a redundant robot arm is not a square matrix, so the inverse Jacobian matrix of redundant robot arm cannot be computed. For instance, the Baxter has dual 7-degree-of-freedom arms. For each arm, the size of the Jacobian matrix is 6×7, which cannot be inverted. Hence, we can use pseudoinverse to resolve this problem as introduced in [27]. Pseudoinverse is written as follows:

$$J^{\#} = J^T(J \cdot J^T)^{-1} \tag{3.16}$$

where J is a $m\times n$ matrix, $m < n$, and $J^{\#}$ is $n\times m$ matrix. Then, by combining the pseudoinverse, equation 3.14 can be written as follows:

$$\dot{\theta} = J^{\#}(\theta) \cdot V \tag{3.17}$$

For Baxter, $J^{\#}$ is a 7×6 matrix, $\dot{\theta}$ is 7×1 vector, and V is 6×1 vector. But the pseudoinverse cannot resolve the singularity problem, so singularity-robust inverse will be introduced in next section [28].

3.6.1 Singularity-Robust Inverse

Nakamura [28] has proposed the singularity-robust inverse (SR-inverse), which can not only resolve kinematic redundancy, but also the singularity problem. SR-inverse is performed as follows:

$$J^{*}(\theta) = J^{T}(J \cdot J^{T} + k \cdot I)^{-1} \tag{3.18}$$

where I is the identity matrix, and k is a scale factor, which is written as follows:

$$k = \begin{cases} k_0(1 - \dfrac{w}{w_0})^2 & for \quad w < w_0 \\ 0 & for \quad w \geq w_0 \end{cases} \tag{3.19}$$

where k_0 is the scale factor at singular points, and w_0 is the measure of manipulability threshold to recognize measure of manipulability, w in equation 3.15, as the neighborhood of singular points. When w is inside of the neighborhood of singular points, k starts to limit the joint rates. Then, on the other hand, when w is outside of the neighborhood of singular points, SR-inverse is equal to the pseudoinverse because k is zero. For each of Baxter's arms, J^{*} is also a 7×6 matrix, and I is a 6×6 matrix. Therefore, SR-inverse can be applied to inverse redundant kinematics as:

$$\dot{\theta} = J^{*}(\theta) \cdot V \tag{3.20}$$

3.6.2 Weighted Least-Norm Solution for Joint Limit Avoidance

Weighted least-norm solution has to be introduced to avoid joint limits as proposed by Chan et al [24]. The definition of a weighted norm of the joint velocity vector is:

$$\left|\dot{\theta}\right|_W = \sqrt{\dot{\theta}^T \, W \, \dot{\theta}} \tag{3.21}$$

where W is a positive and symmetric weighting matrix, and usually a diagonal matrix for simplicity. Then, the following transformations were introduced in Chan's paper for the purpose of analysis:

$$J_W = JW^{-1/2} \text{ and } \dot{\theta}_W = W^{-1/2}\dot{\theta} \tag{3.22}$$

Using equation 3.22, equation 3.13 can be rewritten as:

$$\dot{x} = J_W \, \dot{\theta}_W \tag{3.23}$$

and equation 3.21 can be rewritten as:

$$\left|\dot{\theta}\right|_W = \sqrt{\dot{\theta}_W^T \dot{\theta}_W} \tag{3.24}$$

Then, equation 3.23 can be also rewritten as:

$$\dot{\theta}_W^* = J^*{}_W \, \dot{x} \tag{3.25}$$

By using the second part of equation (3.22), the weighted least-norm of joint rate is:

$$\dot{\theta}_W = W^{-1/2}\dot{\theta}^* \tag{3.26}$$

Using pseudoinverse, the weighted least-norm solution can be obtained as following:

$$\dot{\theta}_W = W^{-1}J^T[JW^{-1}J^T]^{-1}\dot{x} \tag{3.27}$$

Here, the authors defined the weighting matrix, W, to be a diagonal $n \times n$ matrix, which is represented in the following form:

$$W = \begin{bmatrix} 1 + \left|\frac{\partial H(\theta)}{\partial \theta_1}\right| & 0 & \cdots & 0 \\ 0 & 1 + \left|\frac{\partial H(\theta)}{\partial \theta_2}\right| & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 + \left|\frac{\partial H(\theta)}{\partial \theta_n}\right| \end{bmatrix} \tag{3.28}$$

where $\partial H / \partial \theta$ is a performance criterion function that is defined to limit joint motion as follows:

$$\frac{\partial H(\theta_i)}{\partial \theta_i} = \frac{\left(\theta_{i,max} - \theta_{i,min}\right)^2 (\theta_i - \theta_{i,max} - \theta_{i,min})}{4(\theta_{i,max} - \theta_i)^2 (\theta_i - \theta_{i,min})^2} \tag{3.29}$$

where θ_i is the current joint angle, $\theta_{i,max}$ is the maximum joint limit, and $\theta_{i,min}$ is the minimum joint limit for joint i. In equation 3.29, when the current joint angle is in the middle between maximum and minimum of joint limits, the value is zero. On the other hand, when the joint angle is equal to joint limits, the value of the performance criterion function goes to infinity. The value of performance criterion function is shown in figure 3.7. This way, with this function, if each joint angle is in the middle span of its joint limits, the weighting factor is one. In contrast, if the joint angle is close to its joint limits, the value of the weighting factor goes higher. As the weighting factor goes higher, the joint rate goes to lower for that joint. When the current joint angle achieves the joint limit, the weighting factor equals to infinity, and the joint rate is zero. Therefore, the joint angle can be prevented from crossing its joint limits by the heavy weighting factor, and moves freely when it is in the middle span of joint range. But when the joint is moving towards the middle of the joint range from the joint limit, the joint motion should not be penalized by the performance criterion function because of the heavy weighting factor. For resolving the above condition problems, Alqasemi [29] has proposed four conditions by using the performance criterion function. The conditions are defined as following:

1. When the current joint is inside of the joint range, and is moving towards the joint limit, the performance criterion function should be processed the same as equation 3.29. Then the following joint motion is under penalizing condition.
2. When the current joint is inside of the joint range, and moving away from the joint limit, the performance criterion function is equal to zero. After that, the joint can move freely.
3. If the present joint is outside of the joint range, and it is moving away from joint limits to outside of the joint range, the infinite value of performance criterion function should be given as the weighting factor. While the infinite value is applied, the future joint motion is not allowed.
4. If the present joint is outside of the joint range, and moving toward the middle of the joint range, the performance criterion function is equal to zero. Then, the future joint motion can move freely and back to the range of the joint.

With the above conditions, the performance of the joint is suitable until it is very close to the joint limit because of these reasons.

When the joint is very close to the joint limit, the above conditions are insufficient for the joint limit avoidance. The following paragraphs are the explanations. First, only the current and previous joint angles can be compared. Equation 3.29 can only input the current angles, and combined with the SR-inverse and velocities of the end-effector, it can compute the next joint angles. Then, we compare the current and previous joint angles to check the joint motion trend. That means the current and previous joints states are used to determine future joints trend, and apply the joint limit avoidance. It may cause low accuracy of joint limit avoidance. For example, assume there is a joint angle, θ_{time}, is moving towards its limit, shown in figure 3.8. At time i, the joint angle is θ_i, and it is still moving forward to its limit. The next joint angle, θ_{i+1}, is computed by θ_i using equation 3.29. At time $i+1$, θ_{i+1} is applied to equation 3.29 to obtain θ_{i+2}. At this moment, θ_{i+1} is larger than θ_i, and the weighting factor for θ_{i+2} is greater than that for θ_{i+1}. If at time $i+1$, the joint switched direction towards the middle of the joint range, there is no indication for it. The weighting factor for θ_{i+2} is from the above first condition, not the second condition. This way, the weighting factor for θ_{i+2} is heavier than it should be, as shown in Figure 3.8.

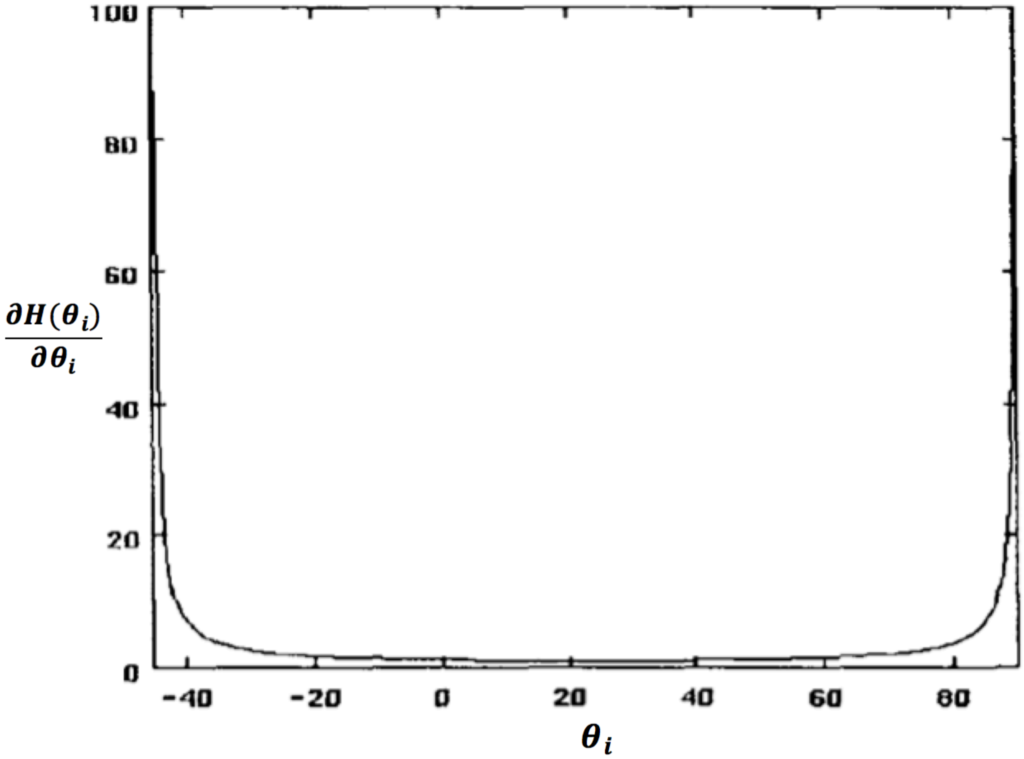


Figure 3.7: Value of Performance Criterion Function.

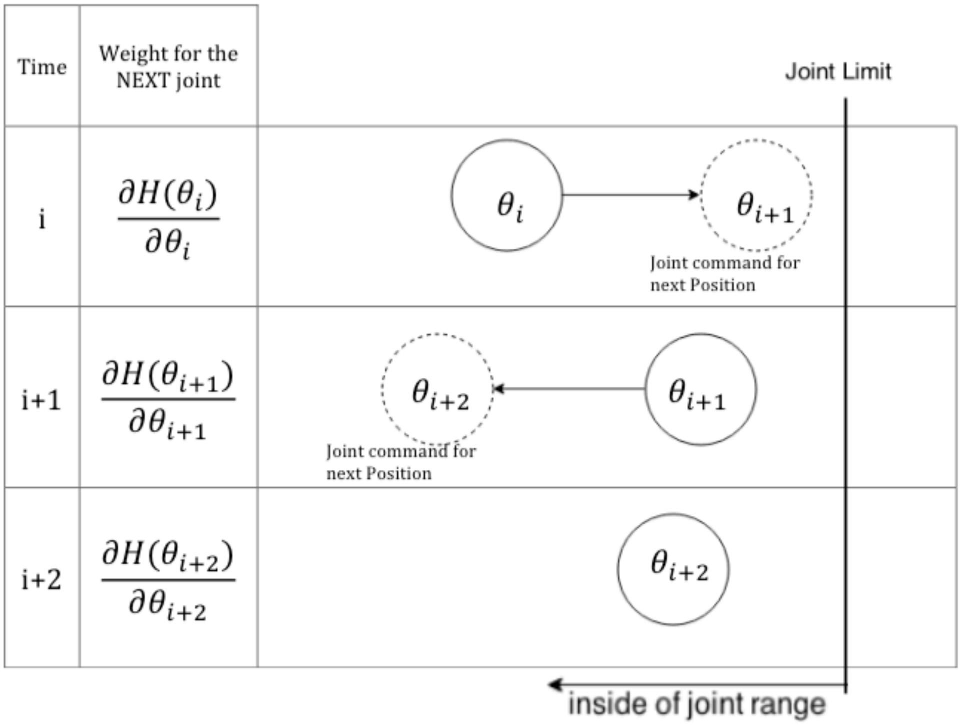

Time
Weight for the NEXT joint
Joint Limit
i
$\frac{\partial H(\theta_i)}{\partial \theta_i}$
θ_i
θ_{i+1}
Joint command for next Position
i+1
$\frac{\partial H(\theta_{i+1})}{\partial \theta_{i+1}}$
θ_{i+2}
θ_{i+1}
Joint command for next Position
i+2
$\frac{\partial H(\theta_{i+2})}{\partial \theta_{i+2}}$
θ_{i+2}
inside of joint range

Figure 3.8: Example of Influence of Weight for Next Joint in Motion Planning.

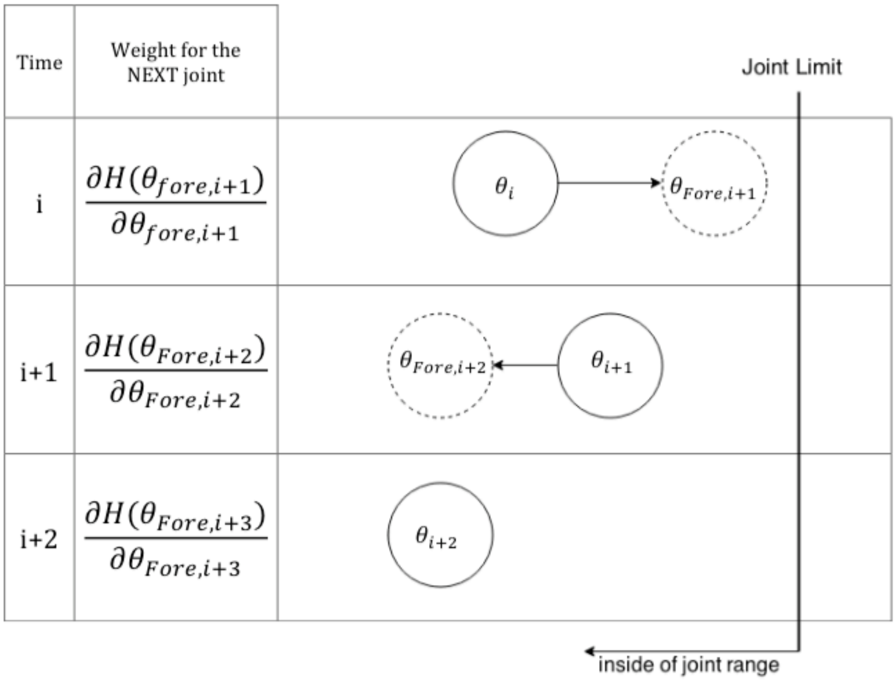
Second, excessively heavy weighting factors may cause a joint lock problem. For the same reason as the first problem, the performance criterion function for the next joint is based on the current joint positions, when the joint goes very close to the joint limit, the weighting factor is very heavy, which causes the weight to be very high for the future joint angles, and the future joint may be hard to move. In this situation, if the joint is trying to move towards the middle of its joint range from the joint limits, it cannot move. The heavy weights may cause the future joint angles to be the same as the current joint angles, and there is no condition when the previous joint angles equal the current joint angles. Therefore, the joint will be locked for a while, and will be hard to move away from the angle that is very close to joint limits.

A good way to resolve those problems is to pre-compute the next joint movements, analyze the trend of the joint by using current and forecast joint angles, and use the forecasted joints in the performance criterion function. Then, use the pre-performance criterion value in equation 3.27 to prevent from achieving or crossing the joint limits with new conditions.

3.6.2.1 Next Joint Motion Forecasting

For getting better results of joint limit avoidance, forecasting the next step of the joint is important. In equation 3.13, once V is obtained, the joint rates can be computed with the current joint angles. By using the joint rates and current angles, the next joint angles can also be established. The weighting matrix should be really applied for the next joint angle by forecasting the joint motion, θ_{fore}, not by the current joint because the current joint angle cannot represent the future joint trend. Hence, forecasting the joint angles and the weighting factor with equation 3.27 should be applied to resolve the problems in 3.6.2. This way, the joints are not allowed to be over their limits or locked when the joints are moving towards the middle of the joint range, as shown in figure 3.9.

To obtain the forecast joint angles, the joint rates and current angles are applied in equation 3.27 without any weighting factor, which means the weight is zero in SR-inverse, and the joint can move without any penalization. This way, the trend of the future joint motion can be exhibited. Then, by using these forecast joint angles and their relations with current joint angles, weighting factors can be obtained. Here, the forecast joint can be considered as an original joint command. The obtained weighting factors will be applied to SR-inverse with the current joint angles to compute the next joint angles.



Time	Weight for the NEXT joint
i	$\dfrac{\partial H(\theta_{fore,i+1})}{\partial \theta_{fore,i+1}}$
i+1	$\dfrac{\partial H(\theta_{Fore,i+2})}{\partial \theta_{Fore,i+2}}$
i+2	$\dfrac{\partial H(\theta_{Fore,i+3})}{\partial \theta_{Fore,i+3}}$

Figure 3.9: Example for Joint Motion Forecasting and Its Weight

3.6.2.2 New Conditions of Joint Limits Avoidance

With the trends of the joint, and pre-performance criterion value, the conditions of the joint limit avoidance have to be reconsidered. The first three conditions are similar to the original conditions.

1. When the current joint, θ_i, and the forecast joint, θ_{fore}, are in the joint range, and θ_{fore} is closer to the joint limits than θ_i, the weight in equation 3.28 is given as the result of equation 3.29 for the following joint, θ_{i+1}. Then θ_{i+1} is under the penalizing condition.
2. No matter where θ_i is, if θ_{fore} is closer to the middle of joint range than θ_i, the weight is given as zero to allow $\theta_{i+1}$to move towards the middle of joint range without any penalizing conditions.

3. No matter where θ_i is, if θ_{fore} is outside of the joint range or on the joint limits, the weight is given as infinity to avoid crossing the joint limits for θ_{i+1}.

In the above conditions, we only consider the conditions of moving joints. What if the current joint and the forecast joint are the same? That means no motion is commanded for that joint at that step. However, every changed weight can affect the results of the other joint angles in the weighted least-norm solution. That means if there is one joint that is penalized by the weight matrix, the other joints in this system will be influenced to change their joint angles because of their weights relative to the penalized joint. Hence, if the forecast joint position is the same as the current joint position, because of the relative joint weight, the result of the next joint angle will be influenced. Moreover, when a joint is very close to its limit, if the joint is under heavy weight, the possibility of whether a joint is crossing its limit or not has to be considered carefully. Because of the above situations, when the current joint and the forecast joint are the same, conditions can be considered as follows:

4. Based on the previous conditions and problems, the rest of the conditions are about the case when the current joint angle and the forecast joint angle are the same. In that case, the original joint rate command is zero. But, in this situation, the weight of the other related joints can influence the joint rate. That causes the joint rate to carry another value instead of zero, and has a possibility to make the joint close to the joint limit. Hence, the rest of the conditions have to be considered in this case. There are two parts of the rest of the conditions, which are separated by a boundary. The boundary of a joint angle is set to distinguish a very-close-to-joint-limit area and a not very-close-to-joint-limit area in the joint

range. Because the related joints may influence the joint rate, when θ_i is equal to θ_{fore}, and θ_i is not in very-close-to-joint-limit area, if the weight of the joint is given as zero, θ_{i+1} may not be the same as θ_i. θ_{i+1} may move towards the joint limit. But when the joint is not in very-close-to-joint-limit area, then there is still some room for the joint to move in the joint range. Hence, the weight is given as the result of equation 3.29 to ensure the result of θ_{i+1} won't be far from the original joint command.

5. When θ_i is in the very-close-to-joint-limit area, and θ_i and θ_{fore} are the same, there is no much room for the joint to move around because the joint may cross the joint limit. Therefore, the weight is given as infinite to avoid the influence of the weights of the related joints. This way, we can ensure that θ_i and θ_{i+1} will be the same, and won't achieve the limits by the effects of other joint weights.

As an example, we consider the two-link robot arm. By using the result of the Jacobian in frame zero, $J(\theta) = \begin{bmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix}$, of the two-link robot arm in [20] (5.67), we can combine the Jacobian with the weight matrices (equation 3.27). If three weight matrices are given as:

$$W_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.30}$$

$$W_2 = \begin{bmatrix} \infty & 0 \\ 0 & 1 \end{bmatrix} \tag{3.31}$$

$$W_3 = \begin{bmatrix} 1 & 0 \\ 0 & \infty \end{bmatrix} \tag{3.32}$$

and assuming that $l_1 = 10m$, $l_2 = 10m$, $\theta_1 = 30°$, $\theta_2 = -45°$, and $v = [10 \quad 30]^T$, $J^*{}_W$ can be computed as below:

$$J^{*}{}_{W_1} = \begin{bmatrix} -0.1366 & 0.0366 \\ 0.2591 & 0.0341 \end{bmatrix} \tag{3.33}$$

$$J^{*}{}_{W_2} = \begin{bmatrix} 0 & 0 \\ 0 & 0.1250 \end{bmatrix} \tag{3.34}$$

$$J^{*}{}_{W_3} = \begin{bmatrix} 0 & 0.0625 \\ 0 & 0 \end{bmatrix} \tag{3.35}$$

With $J^{*}{}_{W}$, the joint rate can also be obtained as:

$$\dot{\theta}_{W_1} = [-15.3495 \quad 207.0669]^T \tag{3.36}$$

$$\dot{\theta}_{W_2} = [0 \quad 214.8592]^T \tag{3.37}$$

$$\dot{\theta}_{W_3} = [107.4296 \quad 0]^T \tag{3.38}$$

Here, the first element of the joint rate matrix, $\dot{\theta}_W$, is the joint rate of θ_1, and the second element is the joint rate of θ_2. It is clear that because of the different heavy weight matrices, the result of $J^{*}{}_{W}$ will be different. And in equation 3.27, if $J^{*}{}_{W}$ is changed, and the requirement of the velocity are the same, the results of joint rates will also be changed.

With the above conditions and methods, the joints can move more freely and smoothly between the limits, easily move back to the joint range from outside of joint limits, and will be stopped when they are going to reach their limits. Figure 3.10 shows the joint position of five of the above conditions and very-close-to-joint-limit area.

3.7 Control Frames

From section 3.4, the position and orientation relation between the desired target and the current end-effector poses can be obtained based on the base frame using a transformation matrix. Hence, the direction of velocity and Jacobian, which are given in equation 3.20 are also based on the base frame. However, when the users are teleoperating the end-effectors, it's usually instinctive to control based on the base frame. The users can comprehend and have more intuitive control if objects' position and pose are relative to the end-effector frame when they are doing teleoperation [29]. Hence, the control frame mode should be easy to switch between the base frame mode and the end-effector frame mode. Then the users can switch the control basis mode for different tasks. In [20], the Jacobian in different frames can be switch from the base frame to the end-effector frame as the equation below:

$$ {}^{end}J = \begin{bmatrix} {}^{base}_{end}R^T & 0 \\ 0 & {}^{base}_{end}R^T \end{bmatrix} \cdot {}^{base}J \qquad (3.39) $$

With ${}^{end}J$, the users can input the velocity vectors, which are based on the end-effector frame, to control the end-effector relative to its own frame.

3.8 Safety Conditions

To use the control algorithm on the physical platform, Alqasemi [29] has suggested adding safety conditions in the system for avoiding joint limits. Two of the suggestions that are going to be used in the system are described below:

1. If a joint reaches its limit, stop the joint.

$$ if\ q_i > q_{i,max}\ orq_i < q_{i,min}, \dot{q}_{i,command} = 0 \qquad (3.40) $$

2. If a joint reaches its velocity limit, slow down the joint, which is also useful when singularity happens and the arms go out of control.

$$ if\ |\dot{q}_i| \geq |\dot{q}_{i,max}|, \dot{q}_{i,command} = sign(\dot{q}_{i,command}) \cdot \dot{q}_{i,max} \qquad (3.41) $$

With the safety condition, the control system can be reliable and avoid physical damage on the platform.

3.9 Summary

In this chapter, the control methods of Baxter's redundant robotic arms are introduced. With the D-H parameter, the parameters of every link and joint can be defined for Baxter system. Then, transformation matrices can be computed. The transformation matrix describes the relation between two neighboring links for every frame with a variable joint angle, and relates each frame to calculate their position and orientation relative to each other. Jacobian matrices define the relations between Cartesian velocities and joint rates in forward and inverse kinematics. Pseudoinverse was generated to optimize the problem of redundant manipulators, and SR-inverse is further used to overcome the singularity problem. Combining SR-inverse and weighted least-norm solution offers an optimized solution to avoid joint limits. Forecasting the next joint angle with current conditions can be more efficient to avoid joint limits. Additionally, the method for switching the control reference frame was also generated for users' better experience in teleoperation.

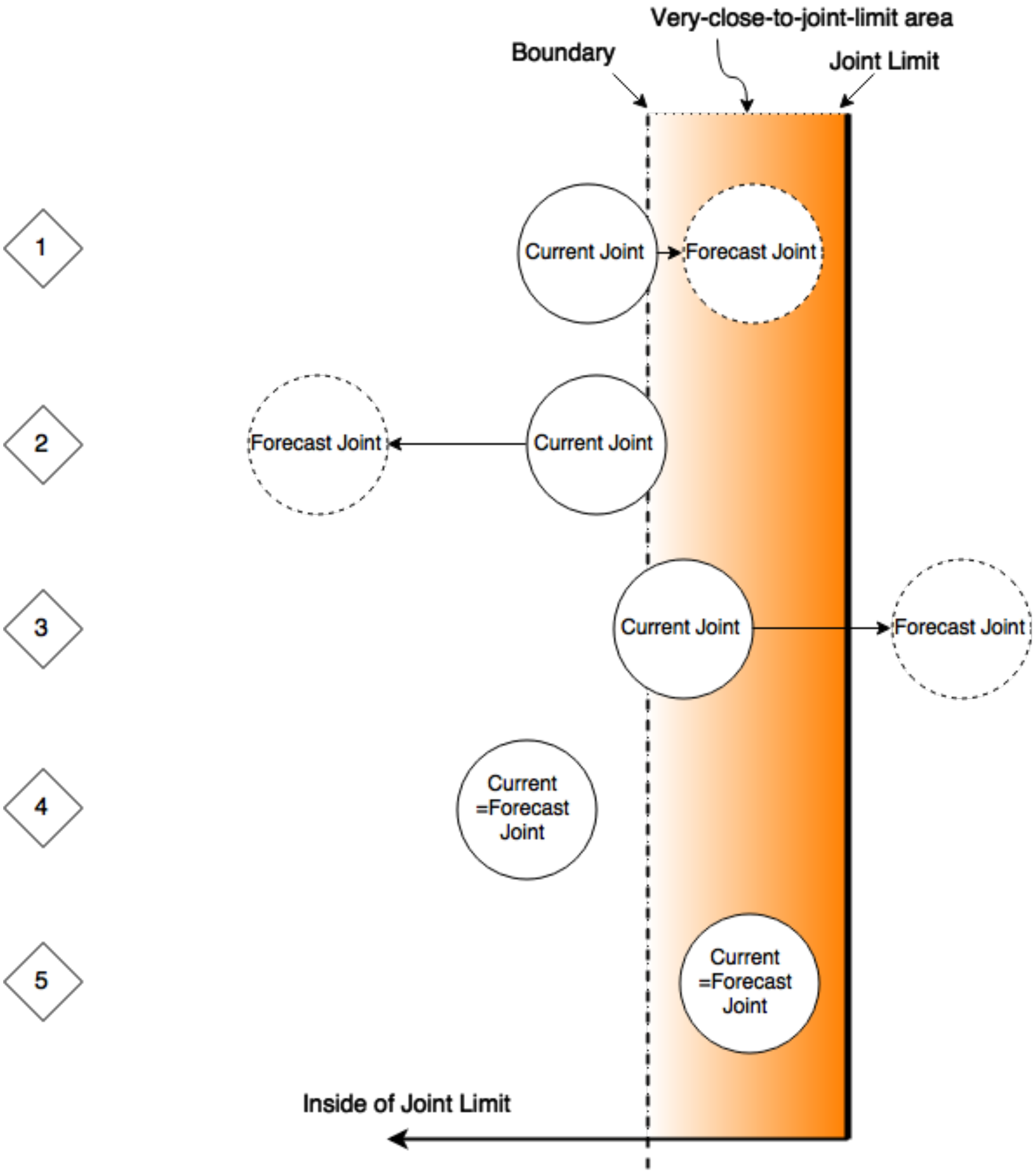


Figure 3.10: Image of Joint Position of the 5 New Conditions.

CHAPTER 4: SIMULATION

4.1 Introduction

Before the control method is applied on the platform for teleoperation, it has to be simulated for detecting errors and improving the code, and confirming the process. In teleoperation, users will give a desired position and orientation as a target for the end-effector. A motion planning system has to be built for the teleoperation system. The motion planning system can give commands of joint angles for moving the end-effector of the arm from one point to the target. Only if the whole process of motion planning and results are stable in the simulation, the method can be applied on the physical platform to avoid damage to the arms. For optimizing the singularity problem, the k_0 and w_0 in equation 3.19 have to be found in simulation. Moreover, the parameters from joint one to joint seven of both Baxter's arms are the same, so in the simulation, only the right arm would be tested.

4.2 Program Flowchart of the Next Joint Motion Forecasting Method

In order to implement the program for manipulator in simulation, the programming language should have powerful and friendly packages for computing matrices and plotting. Hence, Matlab R2013a is used for this simulation.

In Chapter 3, the control methods are introduced. With the control methods, a motion planning system can be built to give commands of joint angles to move the end-effector of the arm from one point to other desired position and orientation. Before those methods are applied on the platform, the calculation of motion planning is necessary to be tested repeatedly in the simulation for achieving the target. First, after obtaining the D-H parameters and initial transformation matrix of each frame from section 3.2 to 3.4, the desired positions and poses of the end-effector should be confirmed. Then, compute the error of positions and orientations. By using the available linear velocity of end-effector and distance between initial and desired positions, the total run time can be generated. Here, the time increment also has to be set in order to compute the running number of increment loops. With the above information, the trajectory and points can be generated. After all of those tasks, the run time will start at zero. Then, the Jacobian is calculated with the current joint angles. The first waypoint is used as the target for the first loop, and delta of the position and orientation are calculated as the rates between the current and target positions. SR-Inverse is then run without weights in order to obtain the forecasting future joint angles. After getting the future joint motion, they are applied into the performance criterion function to avoid joint limits. Later, SR-Inverse and the optimization with the result from the performance criterion function are run to get the joint rates. We then check or modify the joint rates based on the safety conditions. After that, the new joint commands are used, and the new end-effector transformation matrix is calculated using forward kinematics. Because the new joint angles will be the current angles for the next loop in the simulation, we let the current joint angles equal to the new joint angles, and the current transformation matrix equals to the new end-effector transformation matrix. Therefore, until the running time is achieved, the current joint angles will keep being re-computed for the new waypoint as a new target in the loop. When the running time ends, the current position and orientation can be extracted from the current end-effector transformation matrix. The program flowchart is shown in figure 4.1, and the simulation of Baxter's right arm model is shown in figure 4.2.

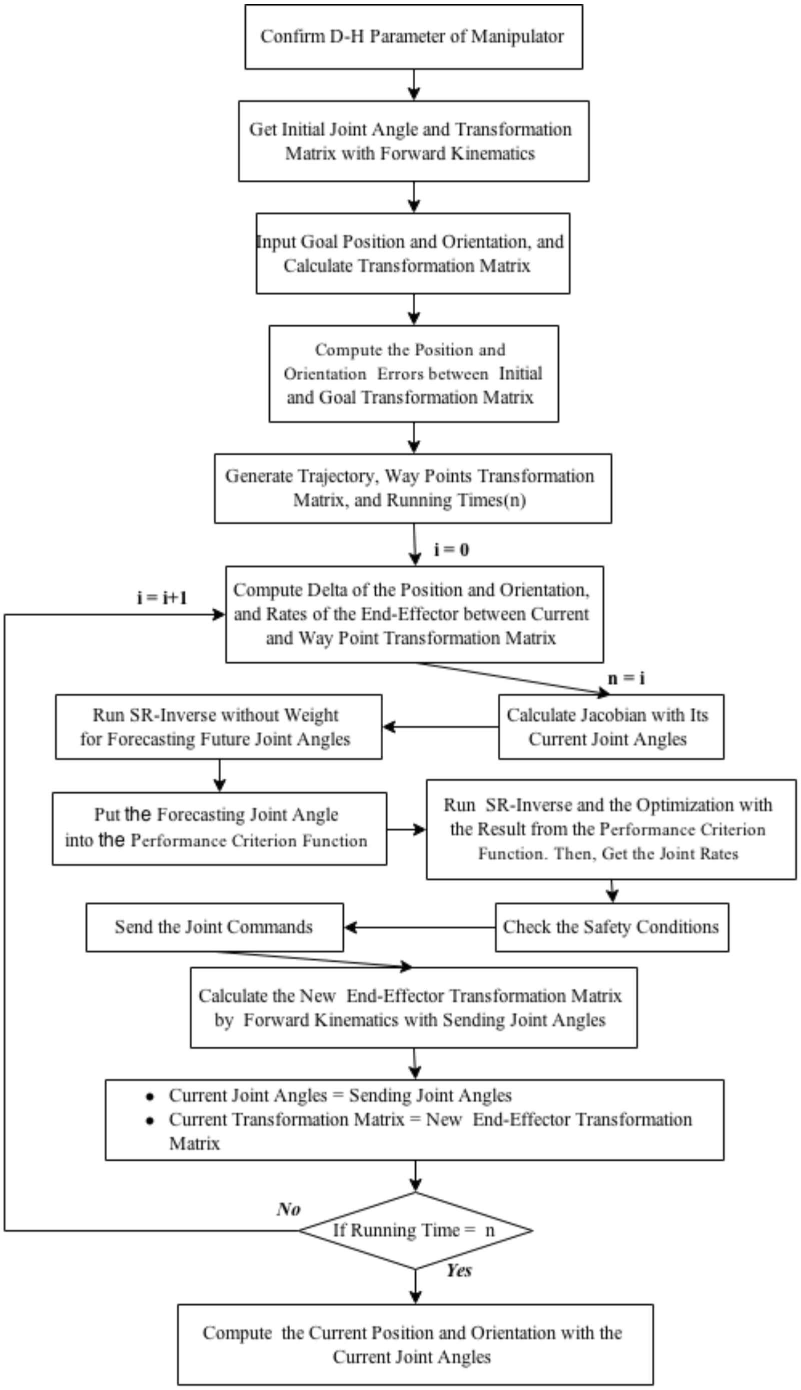

Confirm D-H Parameter of Manipulator
Get Initial Joint Angle and Transformation Matrix with Forward Kinematics
Input Goal Position and Orientation, and Calculate Transformation Matrix
Compute the Position and Orientation Errors between Initial and Goal Transformation Matrix
Generate Trajectory, Way Points Transformation Matrix, and Running Times(n)
i = 0
i = i+1
Compute Delta of the Position and Orientation, and Rates of the End-Effector between Current and Way Point Transformation Matrix
n = i
Calculate Jacobian with Its Current Joint Angles
Run SR-Inverse without Weight for Forecasting Future Joint Angles
Put the Forecasting Joint Angle into the Performance Criterion Function
Run SR-Inverse and the Optimization with the Result from the Performance Criterion Function. Then, Get the Joint Rates
Check the Safety Conditions
Send the Joint Commands
Calculate the New End-Effector Transformation Matrix by Forward Kinematics with Sending Joint Angles
• Current Joint Angles = Sending Joint Angles
• Current Transformation Matrix = New End-Effector Transformation Matrix
No
If Running Time = n
Yes
Compute the Current Position and Orientation with the Current Joint Angles

Figure 4.1: Flowchart of the Simulation Program for the Next Joint Motion Forecasting Method.

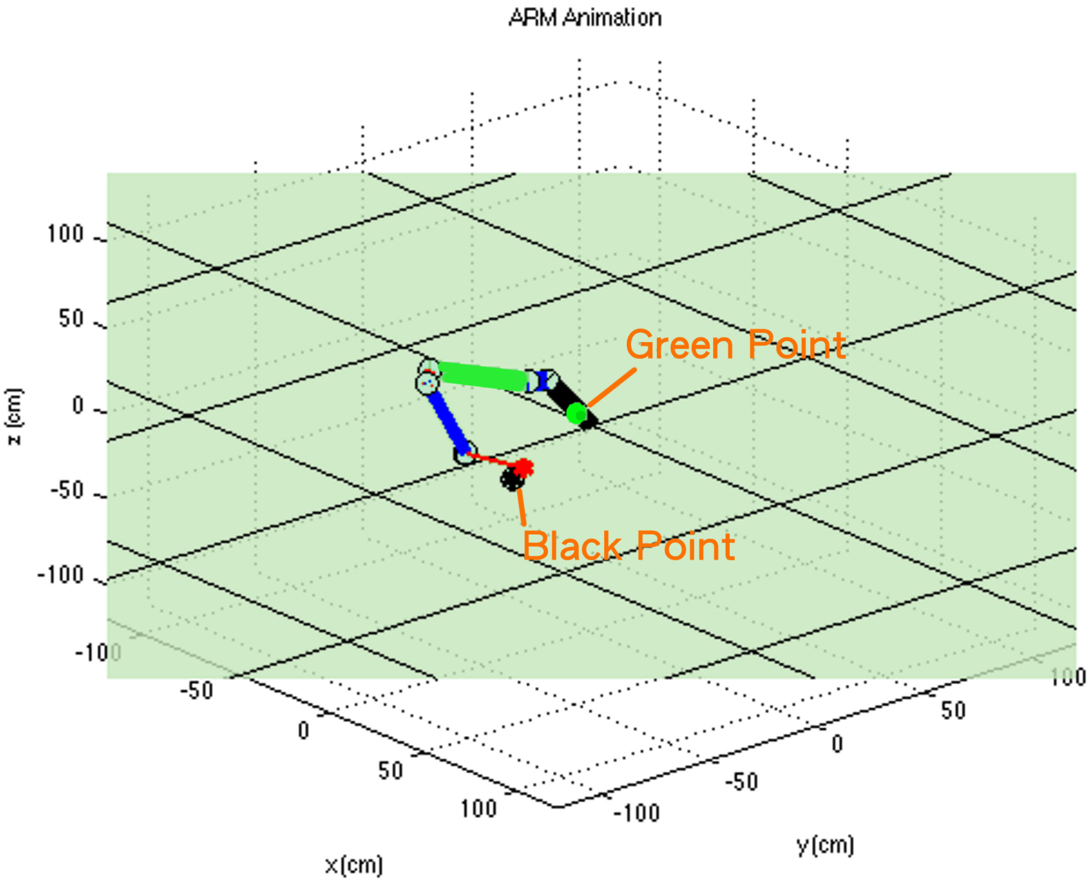


Figure 4.2: Baxter's Right Arm Model while Moving from Black Point to Green Point in Motion Planning Simulation.

4.3 Manipulability Measure and Scale Factor

In order to optimize the redundant manipulator at a singularity, SR-inverse is applied in the inverse kinetics algorithm. In equation 3.18, k is the scale factor. When the manipulability measures, equation 3.15 is too low, this means the manipulator is close to singularities, k starts to work to reduce the joint velocities. But reducing the joint velocities affect the accuracy of the arm. To decrease the effect of reducing the joint velocities, there is a threshold, w_0,after which, the value of the manipulability measure is not acceptable or too close to singularity. This is shown in the equation 3.19. When the manipulability measure is higher than w_0, k is zero. And when the manipulability measure is lower than w_0, k starts to reduce the joint velocities by the results of equation 3.19. Hence, it is critical to choose w_0 and k_0 for k. If k is too high, the accuracy will be low. If k is too low, the joint rates will be too high, and the arm will be unstable. Using the simulation, the ideal w_0 is 2×10^4 and k_0 is 570. Figure 4.3 and figure 4.4 show the results of the manipulability measure when the desired positions of both arms are out of the workspace, which drive the arms to singularity, when the determined values of k_0 and w_0 are used.

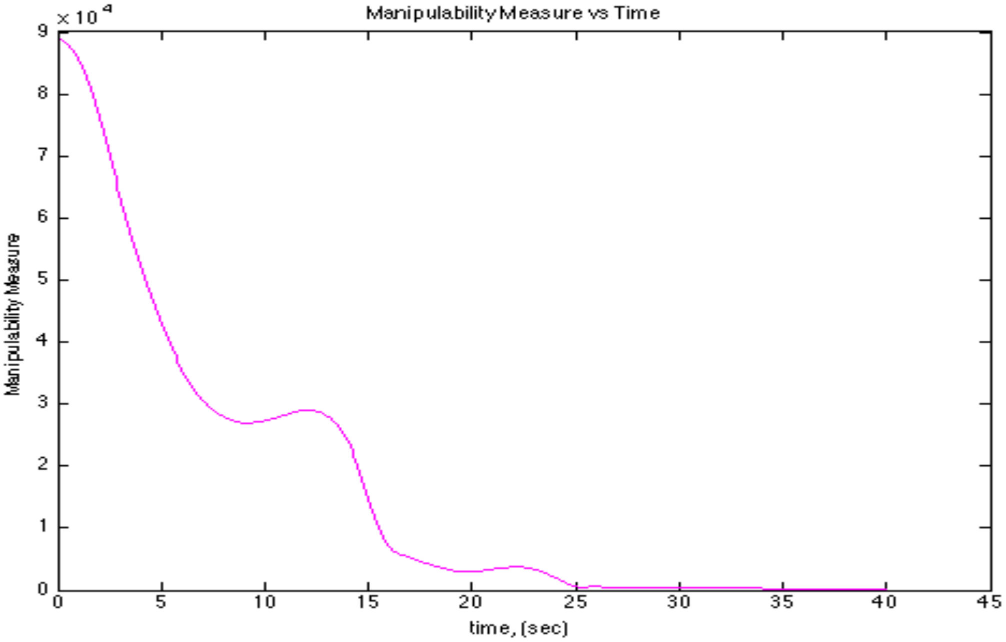


Figure 4.3: Manipulability Measure of Baxter's Right Arm: Goes to Zero When it is Moving Out of the Workspace with the Ideal k_0 and w_0.

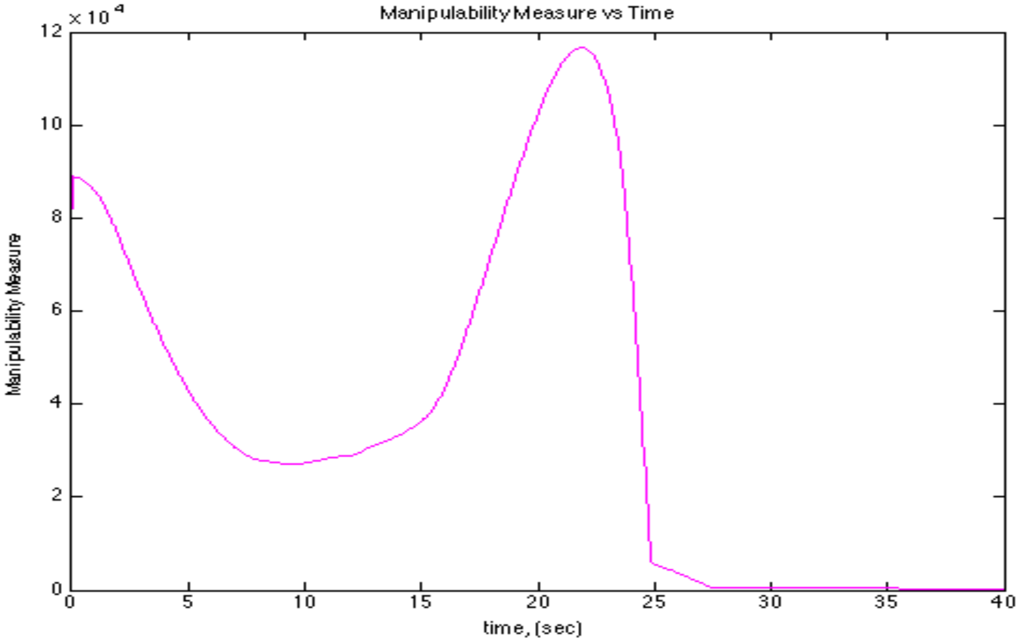


Figure 4.4: Manipulability Measure of Baxter's Left Arm: Goes to Zero When it is Moving Out of the Workspace with the Ideal k_0 and w_0.

4.4 Results and Discussion of Simulation Test

With the new conditions of the performance criterion function and process, the problems in 3.6.2 can be overcome effectively. In the simulation, the results of the condition and process are tested and compared with the other conditions and processes in the motion planning algorithm.

4.4.1 Test of Next Joint Motion Forecasting

The weighted least-norm solution gives a forecast into the future joint angle, and that allows the joint to be penalized if the joint is too close to the joint limits. In this section, before teleoperation testing, the next joint motion forecasting method will be tested with motion planning, and compared with a non- forecasting motion planning in the simulation program. There are five tests to be implemented in different conditions and control methods:

1. The motion planning with non-forecast will be tested without the weight matrix. This case will be the baseline for our comparisons.
2. The motion planning with non-forecast will be tested with weighted least-norm solution.
3. The motion planning with non-forecast will be tested with weighted least-norm solution and the original conditions in chapter 3 section 3.6.2.
4. The motion planning with next joint motion forecasting method will be tested without the weight matrix.
5. The motion planning with next joint motion forecasting method will be tested with weighted least-norm solution.

The tests will be implemented on the right arm of Baxter. In all of these tests, the initial transformation matrix is set as:

$$ {}^{0}_{7}T_i = \begin{bmatrix} 0.2463 & -0.7990 & 0.5485 & 20.2481 \\ -0.4242 & 0.4200 & 0.8023 & -42.6812 \\ -0.8714 & -0.4303 & -0.2355 & 3.4457 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1} $$

and the goal transformation matrix is set as:

$$ T_g = \begin{bmatrix} 0.9361 & 0.3507 & 0.0267 & 30 \\ -0.0819 & 0.1436 & 0.9862 & -30 \\ 0.3420 & -0.9254 & 0.1632 & 50 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2} $$

Moreover, in order to compare the effect of weighted least-norm solution, the initial angle of joint (5) is set on its limit, which is $-175.3°$.

In the first test, the result of the transformation matrix of the end-effector is:

$$ {}^{0}_{7}T = \begin{bmatrix} 0.9361 & 0.3507 & 0.0267 & 30 \\ -0.0819 & 0.1436 & 0.9862 & -30 \\ 0.3420 & -0.9254 & 0.1632 & 50 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4.3) $$

which is the same as the target. However, joint (5) starts to cross the joint limit in the first step, and moves into the joint range at 4.2 seconds, as shown in figure 4.5. The complete run time is 21.1 seconds. Figure 4.6 shows joint rates of Baxter's right arm in the case of test 1.

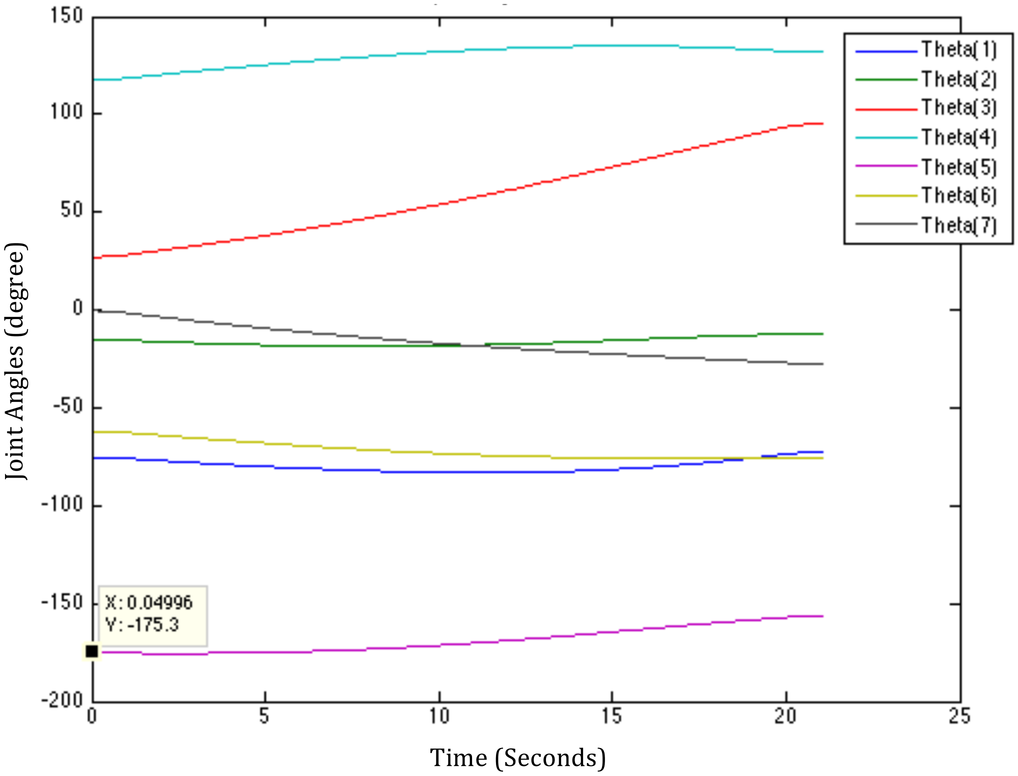


Figure 4.5: The Joint Angles of Baxter's Right Arm in Test 1.

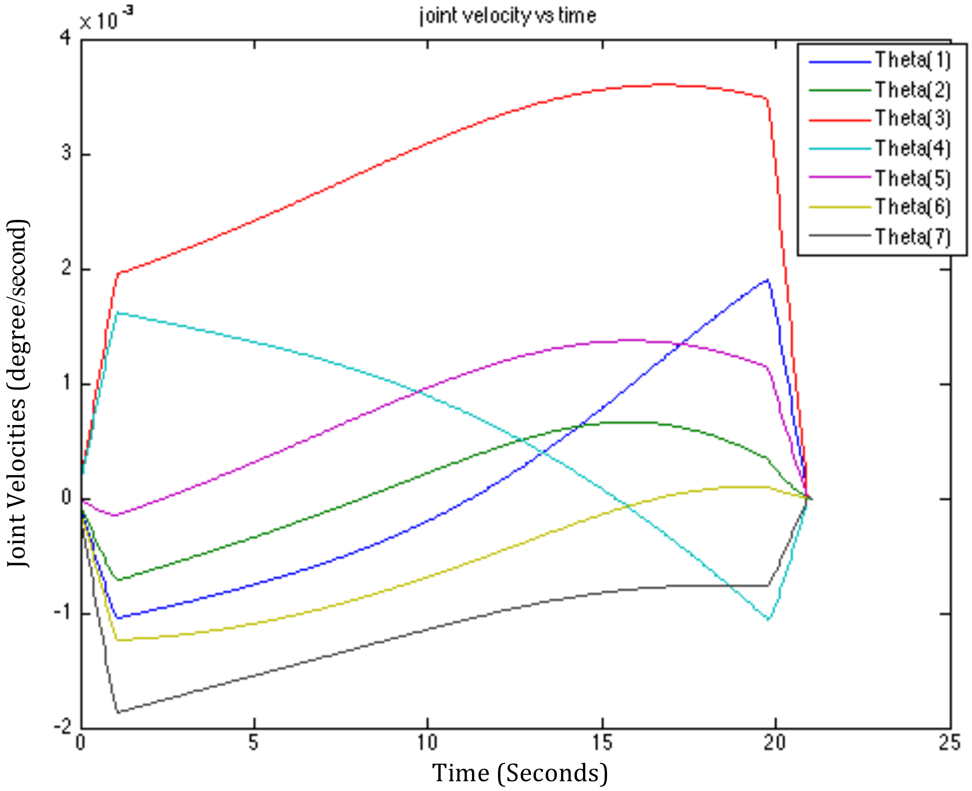


Figure 4.6: The Joint Velocities of Baxter's Right Arm in Test 1.

The second test is to check the result of the non-forecast motion planning with weighted least-norm solution. The complete run time of the second test is 21.1 seconds. The result of the transformation matrix of the end in this test is:

$$ {}^{0}_{7}T = \begin{bmatrix} 0.9344 & 0.3426 & 0.0975 & 29.9698 \\ -0.0738 & -0.0815 & 0.9939 & -30.0307 \\ 0.3484 & -0.9360 & -0.0509 & 49.9841 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4.4) $$

which has small differences between the result and the commanded target. joint (5) is locked by the weight because of its initial joint angle. Figure 4.7 shows the joint angle of joint (5) as it is locked at its initial joint angle. Figure 4.8 shows the joint rate of the joint (5), which is almost zero throughout the simulation. The locked joint (5) may cause a small difference in the resulting matrix.

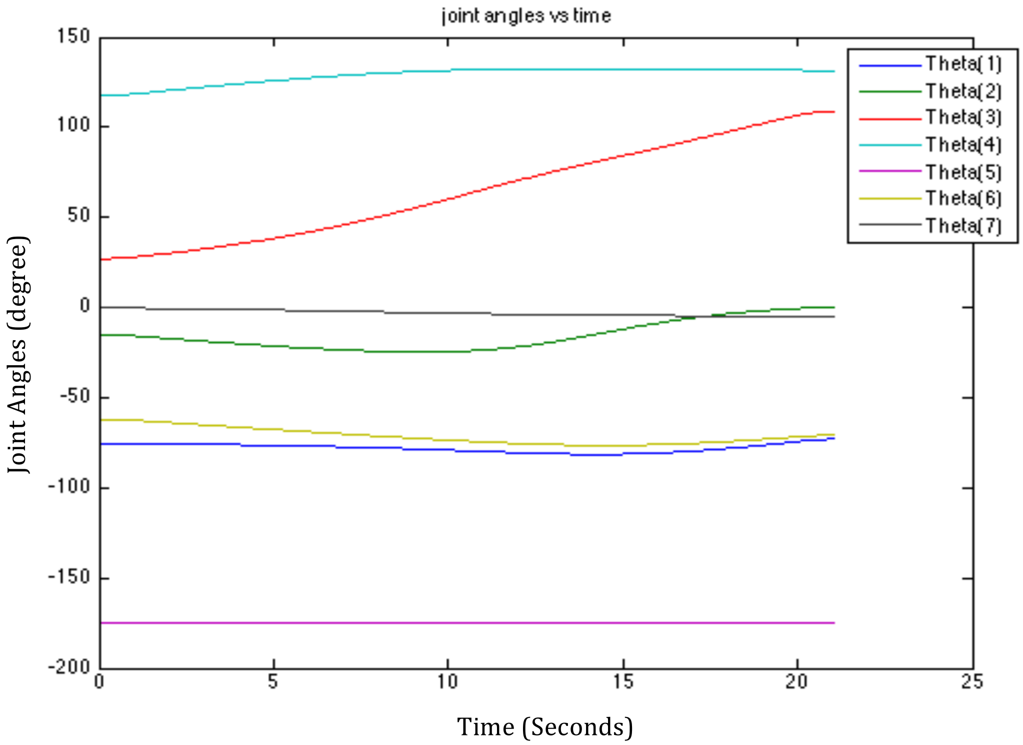


Figure 4.7: The Joint Angles of Baxter's Right Arm in Test 2.

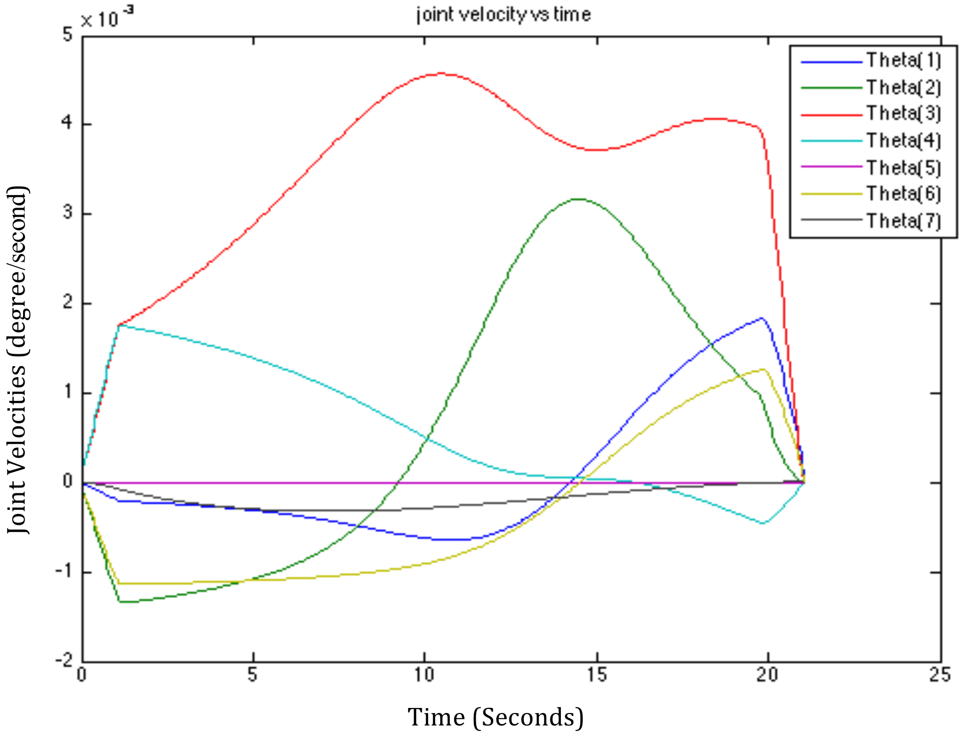


Figure 4.8: The Joint Velocities of Baxter's Right Arm in Test 2.

The non-forecasting motion planning will be tested with weighted least-norm solution and the original conditions in the third test. The result of the test is as follows:

$$ {}_{7}^{0}T = \begin{bmatrix} 0.9301 & 0.3617 & 0.0639 & 30.0002 \\ -0.1273 & 0.1542 & 0.9798 & -29.9987 \\ 0.3445 & -0.9195 & 0.1895 & 49.9995 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5} $$

The resulting pose of the third test is different from the commanded target. Joint (5) is also locked throughout the test due to the weight. Figure 4.9 shows the joint angle of joint (5), which does not change through the entire process. Figure 4.10 shows the joint velocity of joint (5), which is zero from the beginning to the end of the test.

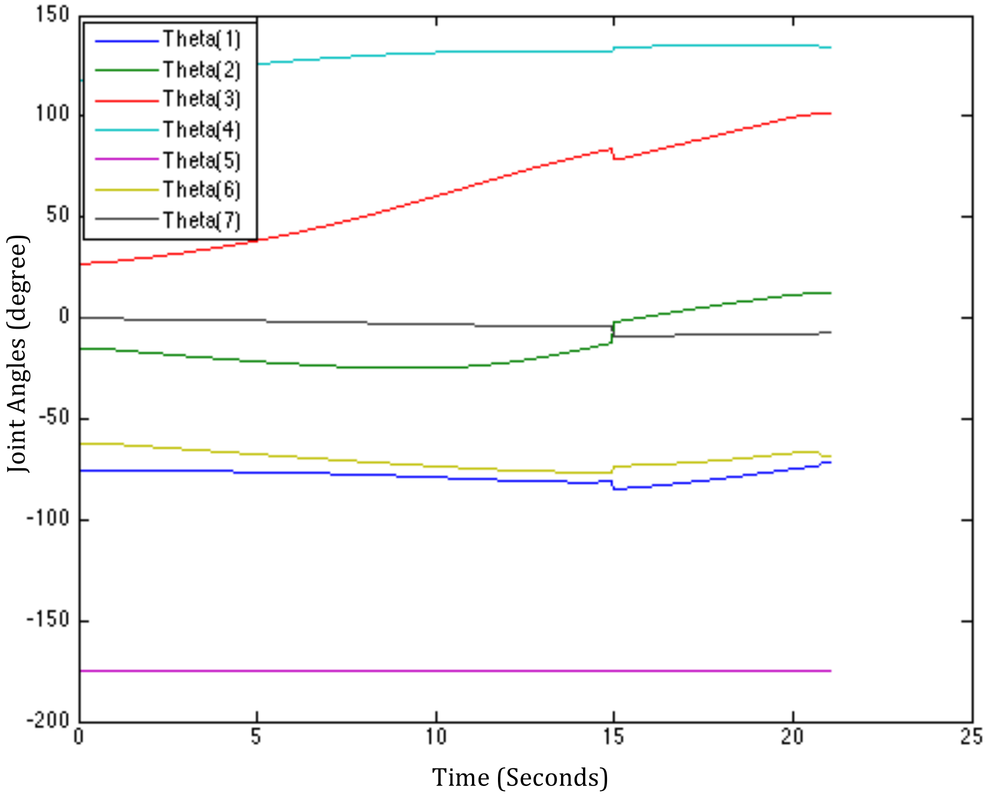


Figure 4.9: The Joint Angles of Baxter's Right Arm in Test 3.

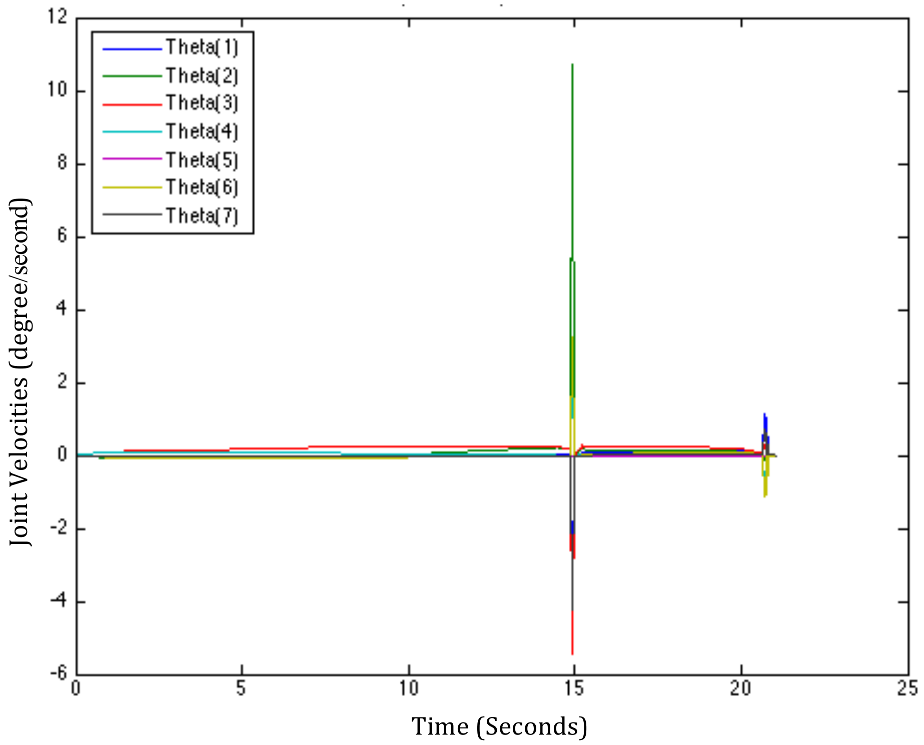


Figure 4.10: The Joint Angles of Baxter's Right Arm in Test 3.

In the fourth test, the next joint motion forecasting method is tested without the weighted least-norm solution. The result of the end effector's pose in the third test is:

$$ {}_7^0T = \begin{bmatrix} 0.9361 & 0.3507 & 0.0267 & 30 \\ -0.0819 & 0.1436 & 0.9862 & -30 \\ 0.3420 & -0.9254 & 0.1632 & 50 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.6} $$

The result is similar to the commanded target and the first test. The complete run time is 21.1 seconds. In the first step, joint (5) starts to cross the joint limit, and it moves back to the joint range at 4.2 seconds, as shown in figure 4.11. Figure 4.12 shows the joint rates in the fourth test.

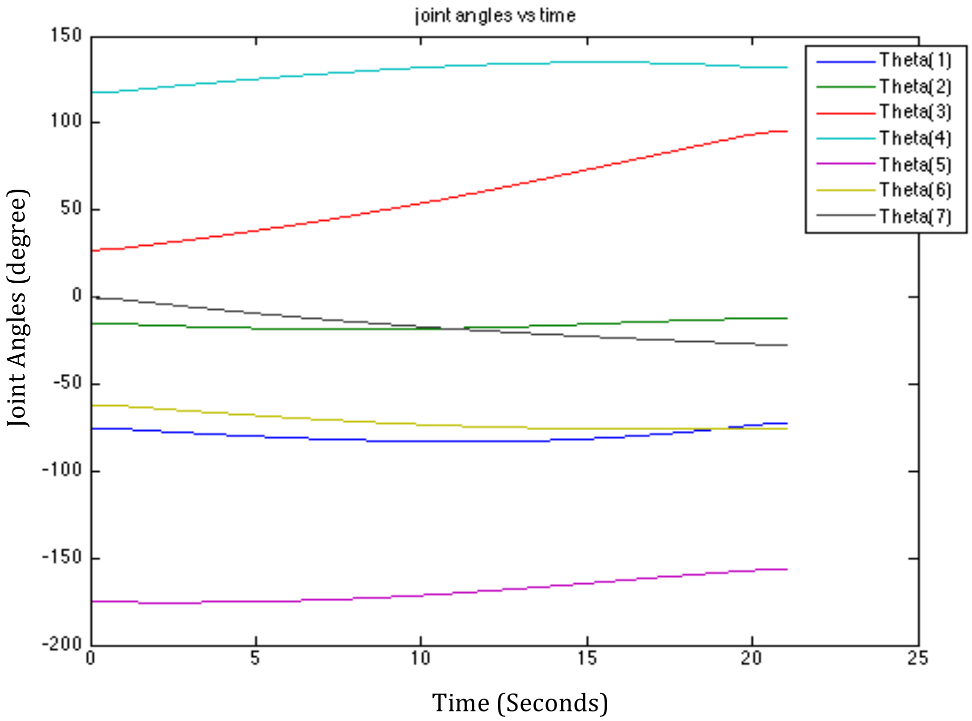


Figure 4.11: The Joint Angles of Baxter's Right Arm in Test 4.

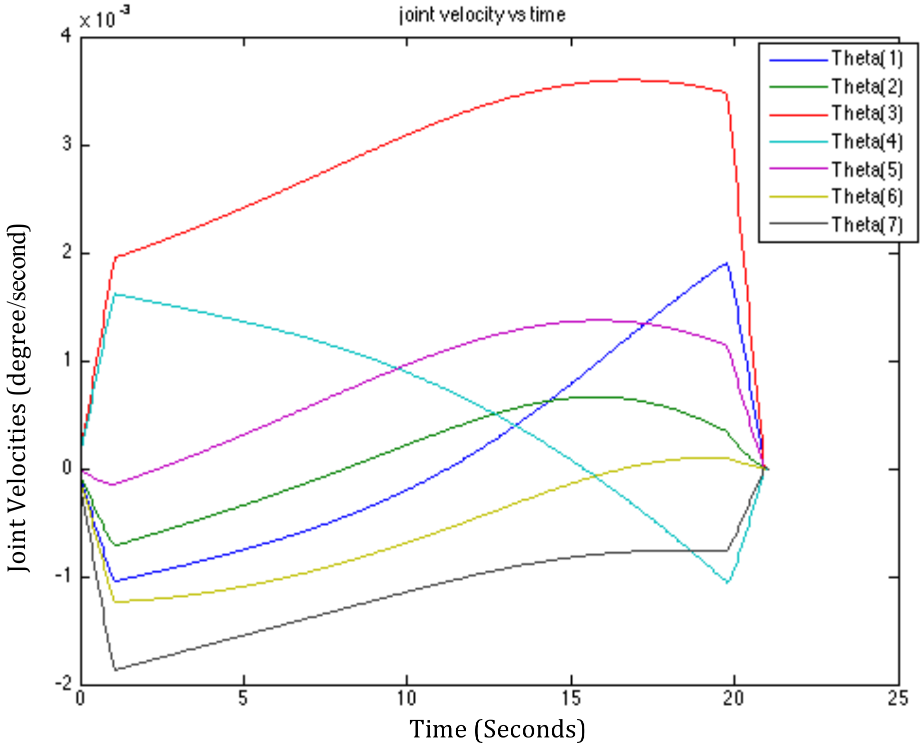


Figure 4.12: The Joint Velocities of Baxter's Right Arm in Test 4.

The last test is to implement the next joint motion forecasting method with the weighted least-norm solution. In this test, the result of the end-effector's transformation matrix is:

$$ {}^{0}_{7}T = \begin{bmatrix} 0.9376 & 0.3379 & 0.0821 & 29.9806 \\ -0.0300 & -0.1565 & 0.9872 & -30.0208 \\ 0.3465 & -0.9281 & -0.1365 & 49.9922 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7} $$

which has small differences between the resulting matrix and the commanded target. But the differences are smaller than those in test 2. Joint (5) is not locked by the weight, and it moves back to the joint range at 1.5 seconds. The end of the joint angle of joint (5) is 175.1°. The complete run time is 21.1 seconds. Figure 4.13 shows the joint angles. Because the difference in joint (5) between the initial angle and the end angle is small, the line of joint (5) looks horizontal. But in figure 4.14, the joint velocity of joint (5) increases. Also, because of the increased joint rate, the differences between the resulting matrix and the target matrix are smaller than that difference in test 2.

In those five tests, it is clear that the results of the "no forecast motion planning" without the weighted least-norm solution, and the "next joint motion forecasting method" without the weighted least-norm solution, are the same. But with the weighted least-norm solution, the next joint motion forecasting method can offer a better solution to stay away from the joint limit than the motion planning with no forecasting. In the next step, the new conditions will be applied to the next joint motion forecasting method.

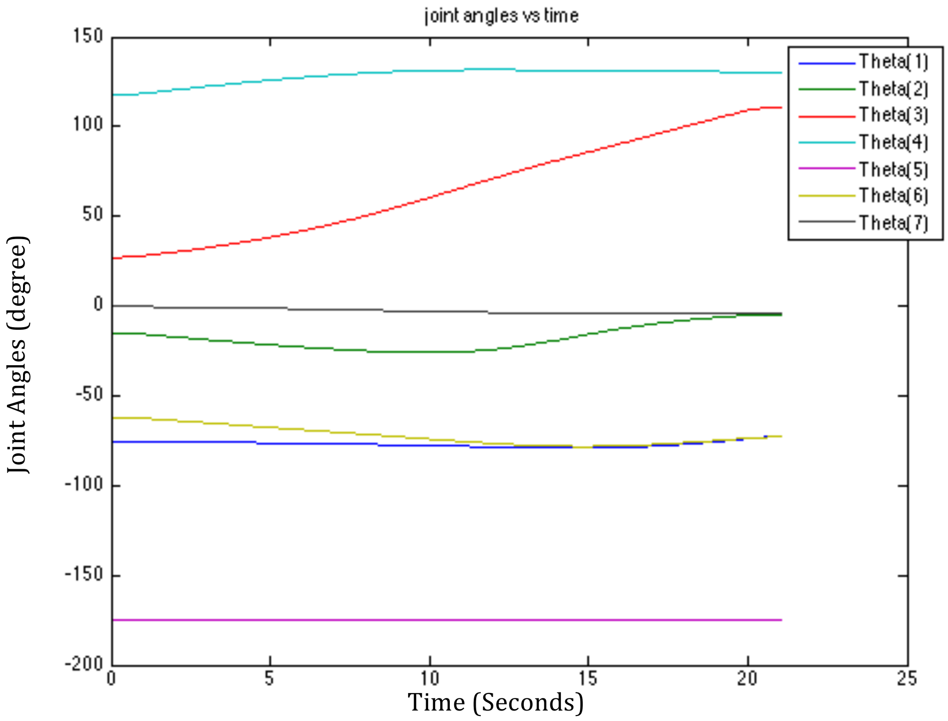


Figure 4.13: The Joint Angles of Baxter's Right Arm in Test 5.

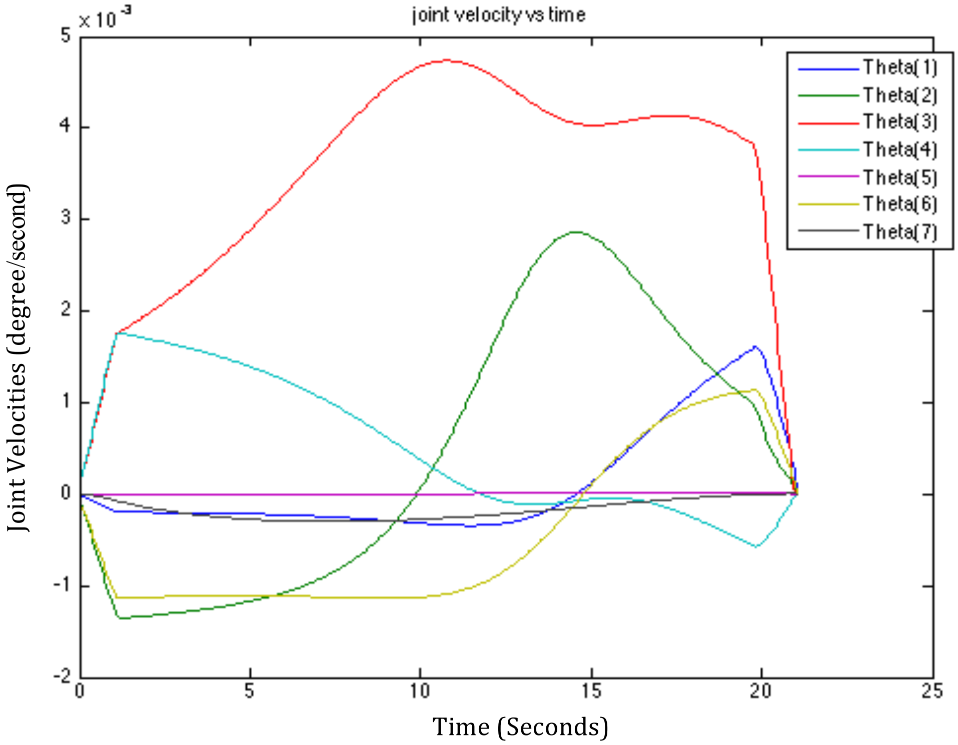


Figure 4.14: The Joint Velocities of Baxter's Right Arm in Test 5.

4.4.2 Improvement of the New Conditions on Joint Limit Avoidance

With forecasting the next joint motion, the joint can more easily move forward to the middle of its range from the joint limits than when there is no forecast, if the weighted least-norm solution is applied. However, there is still some room to improve the performance with some joint trend conditions, which are discussed in Chapter 3 section 3.6.2. Extending the forth test in Chapter 4 section 4.4.1, the new conditions of joint limit avoidance and the original conditions of joint limit avoidance will be implemented with the next joint motion forecasting method. The results of both conditions will be discussed.

With the original conditions, the resulting transformation matrix of the end-effector is:

$${}_{7}^{0}T = \begin{bmatrix} 0.9970 & -0.0461 & 0.0628 & 29.9867 \\ -0.0445 & 0.3247 & 0.9448 & -30.0015 \\ -0.0640 & -0.9447 & 0.3216 & 50.0003 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

The position of this transformation matrix is close to the target (equation 4.2). But the rotation part has some elements that show obvious difference from the target. Figure 4.15 and figure 4.16 show joint angles and joint velocities with the original conditions in this test, respectively. In contrast, the resulting transformation matrix of the end-effector with the new conditions is:

$${}_{7}^{0}T = \begin{bmatrix} 0.9359 & 0.3509 & 0.0295 & 29.9993 \\ -0.0846 & 0.1429 & 0.9861 & -30.0011 \\ 0.3418 & -0.9254 & 0.1634 & 49.9988 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

which has smaller differences in the rotation part, and the position part is also close to the target. Figure 4.17 and figure 4.18 show the joint angles and joint velocities with the new conditions in this test, respectively.

However, even though the results of the new conditions are better, the joint rates with these conditions may be high for the physical platform, as shown in figure 4.18. In Chapter 3 section 3.8, the suggestions of safety conditions [29] have been introduced. Hence, by adding the suggested safety conditions, a better result than equation 4.9 can be obtained as:

$$
{}_{7}^{0}T = \begin{bmatrix} 0.9361 & 0.3507 & 0.0267 & 30.0001 \\ -0.0819 & 0.1436 & 0.9861 & -30.0001 \\ 0.3420 & -0.9254 & 0.1632 & 50.0000 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.10}
$$

The new conditions of joint limit avoidance and the safety conditions are suggested to be implemented on the next joint motion forecasting method. Figure 4.19 and figure 4.20 show the joint angles and joint velocities with the new joint limit avoidance conditions and the safety conditions in this implementation, respectively.

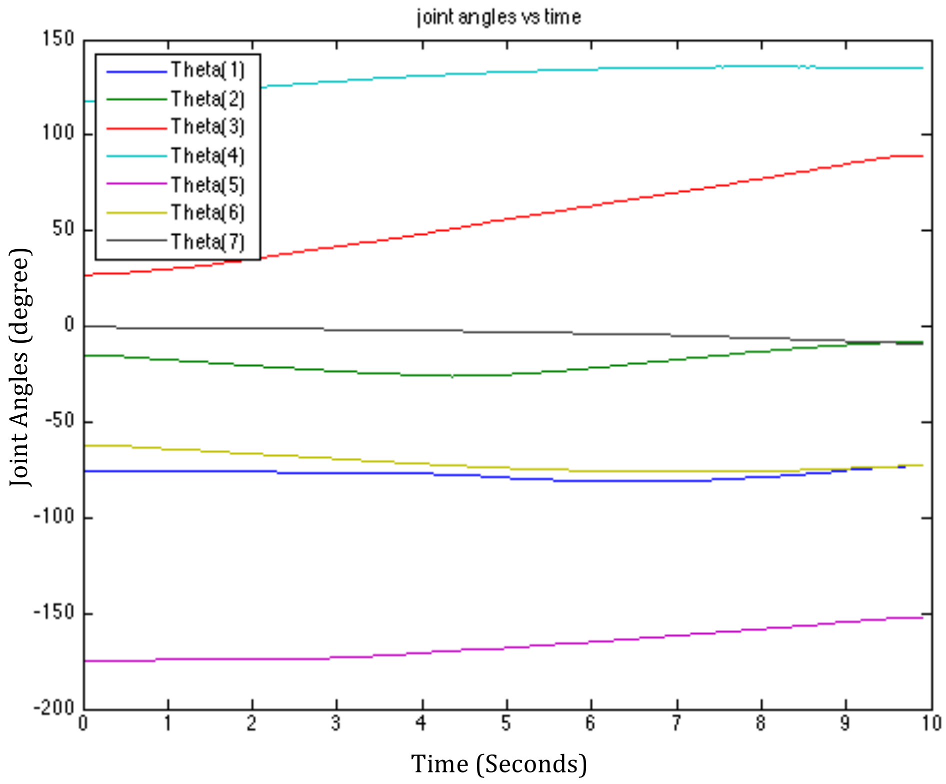


Figure 4.15: Joint Angles with the Original Joint Limit Avoidance Conditions.

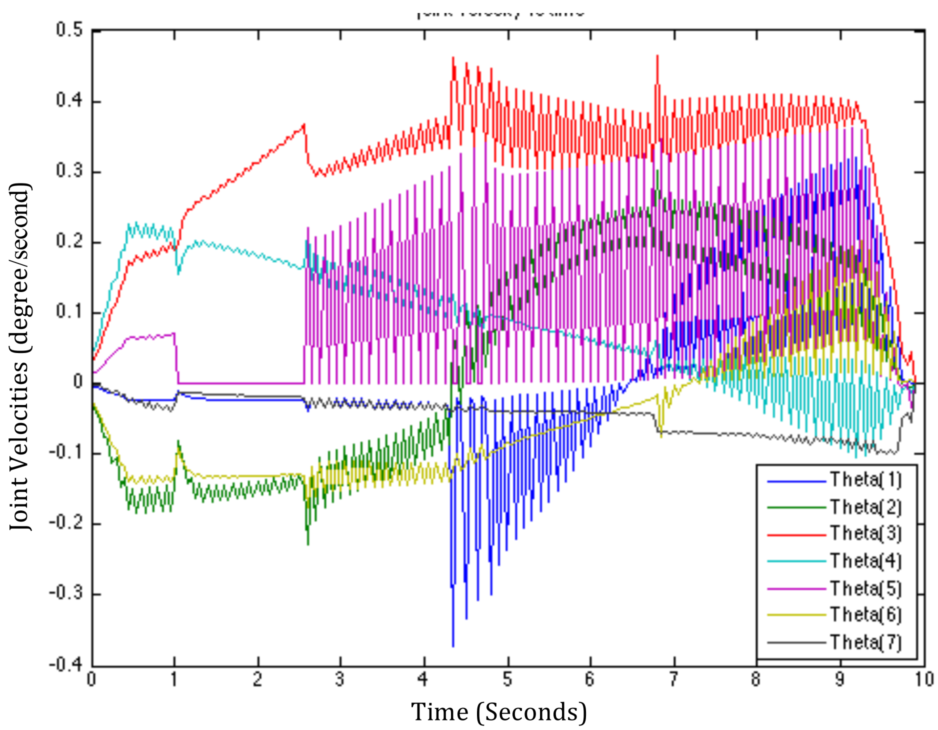


Figure 4.16: Joint Velocities with the Original Joint Limit Avoidance Conditions.

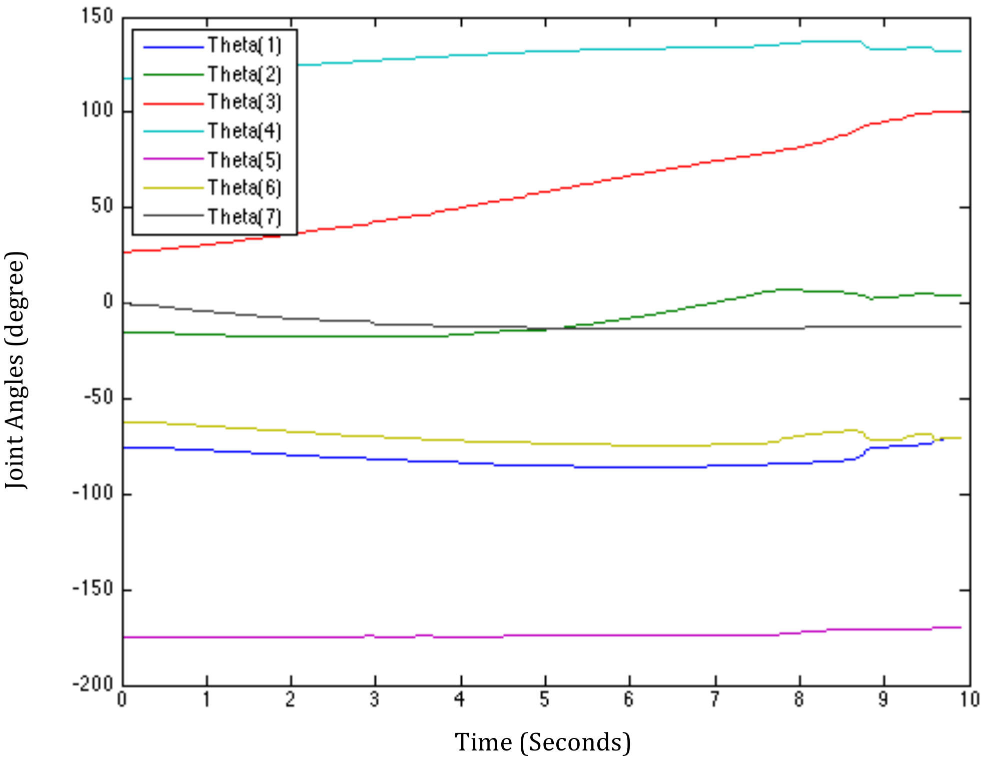


Figure 4.17: Joint Angles with the New Joint Limit Avoidance Conditions.

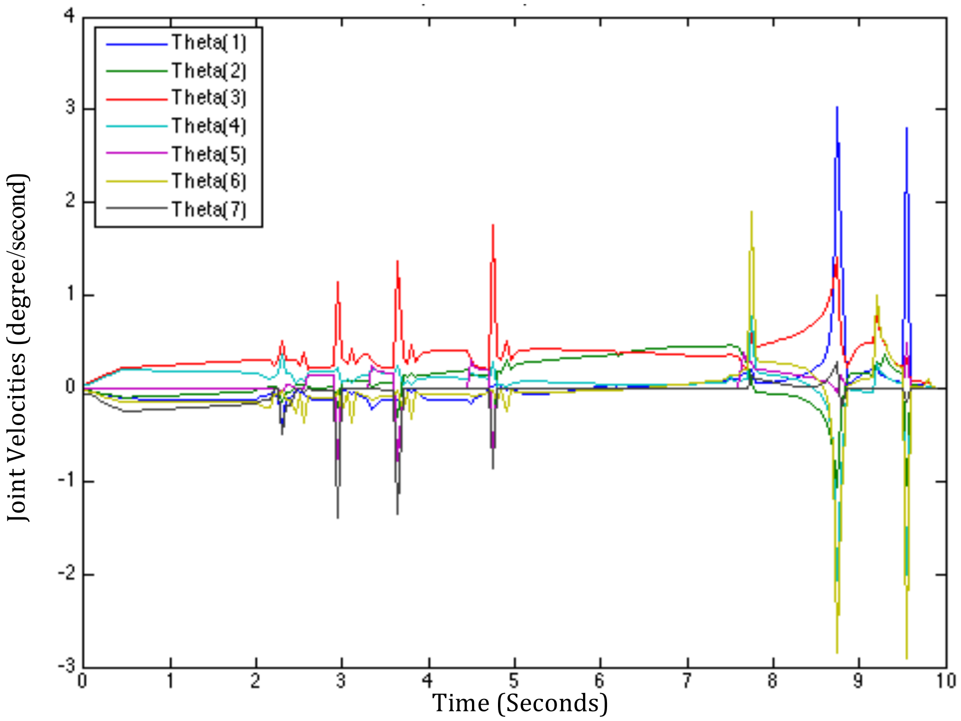


Figure 4.18: Joint Velocities with the New Joint Limit Avoidance Conditions.

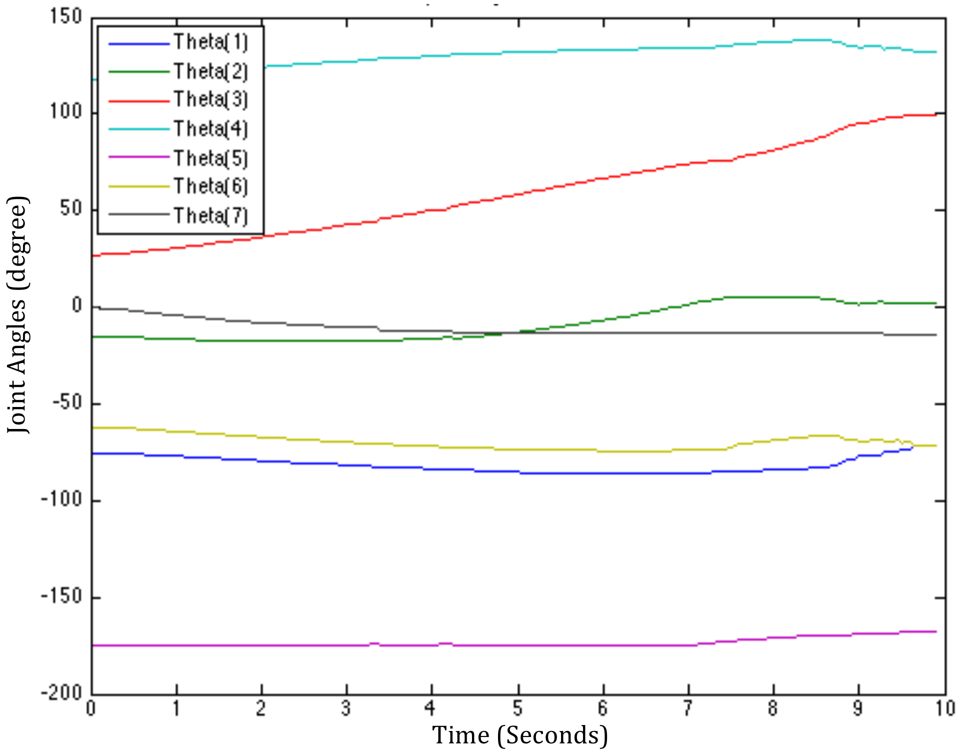


Figure 4.19: Joint Angles with the New Joint Limit Avoidance Conditions and Safety Conditions.

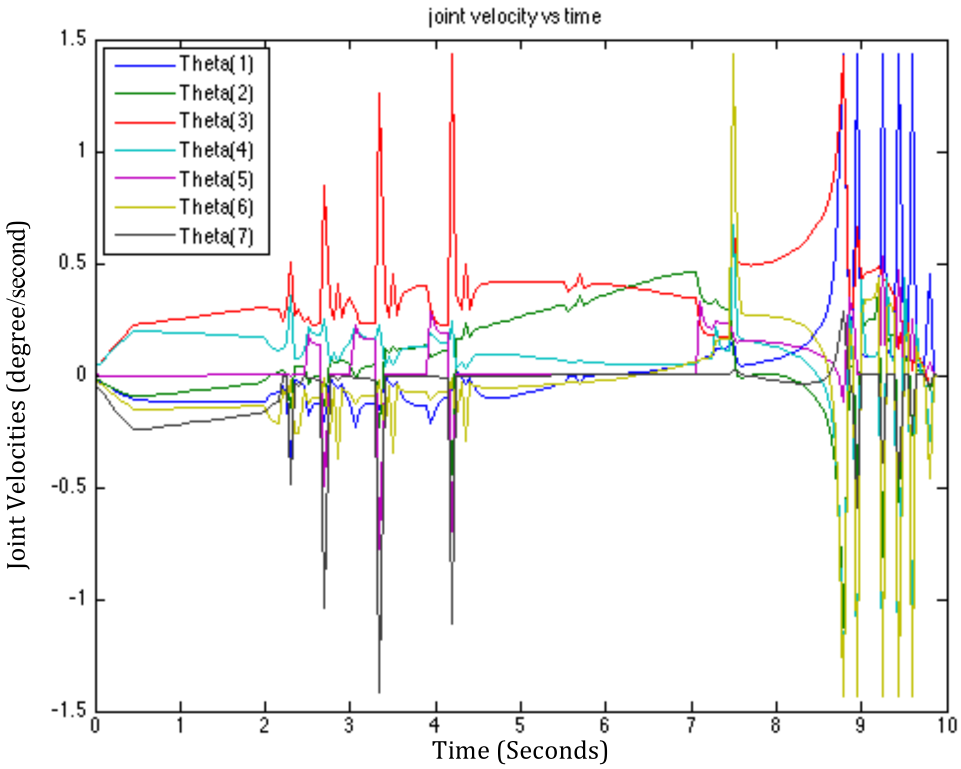


Figure 4.20: Joint Velocities with the New Joint Limit Avoidance Conditions and Safety Conditions.

4.5 Summary

In this chapter, the simulation program and flowchart are introduced. The most important part of the process is the forecasting of the joint angle, which offers the program to recognize the joint motion trend, and better perform joint limit avoidance. Moreover, the manipulability measure and scale factor are found to optimize the singularity for Baxter's arms. All the conditions and processes are tested in simulation before they are applied on the platform. Then, the tests of the forecasting joint angle process and new joint limit avoidance conditions are also discussed.

CHAPTER 5: IMPLEMENTATION OF TELEOPERATION ON THE PHYSICAL PLATFORM

5.1 Introduction

After the simulation test and confirmation, the flowchart from Ch. 4.2 can be applied on the physical platform, Baxter. On Baxter, the flowchart will be tested in motion planning. The results will be compared to the simulation with the same task. Then, teleoperation system can be developed based on the motion planning flowchart on Baxter. Hence, the whole teleoperation system will be discussed and tested in this chapter. Moreover, an obstacles collision force feedback system will be developed and implemented for the teleoperation system. For a better user experience, the user interface and collision feedback will also be introduced.

5.2 Teleoperation Program

There are four main parts to establish the implemented teleoperation system. The first part is the velocity input. In the simulation, the flowchart was only about motion planning, which will be the second part of the implementation. The first and second parts are built in C++ because the motion planning program and API of the haptic device for velocity input are also built on C++. The third part is the communication between the master side and slave side. For building the new teleoperation flowchart, the protocol used in the input and communication signal should be confirmed. The last part of the teleoperation system is the platform, Baxter. Moreover, the collision force feedback is also included in the real-time teleoperation system.

5.2.1 Velocity Input

In inverse kinematics, current joint angles and velocities of the end-effector are necessary inputs to compute joint rates. The current joint angles come from the current pose of the arms. The teleoperation control devices have to input velocity commands. Then the velocities will be sent to the motion planning. Here, because it is the real time input, there is no waypoint between every single target. The motion planning program will compute the transformation matrix from the current position to the next target.

5.2.2 Communication

The communication between Baxter and the user controls is done via TCP/IP socket across the network. In this research, Baxter part is the server point, and the motion planning part is the client point. The client point is built in C++, and Python builds the server point in Robot Operating System (ROS). The main signal between the server and client carries the current joint angles, joint angle commands, griper open and close commands, and force feedback vectors.

5.2.3 Physical Platform

The physical platform for the research is the Baxter robot, which is made by Rethink robotics. Baxter has dual 7-degree-of-freedom arms. The typical accuracy of its joints is +/−0.10 degrees, and when the joints approach the joint limit, the accuracy is worst, which is +/−0.25 degrees [30]. The ROS interface for Baxter allows writing the program with C++ and python.

5.2.4 Collision Feedback

For a better teleoperation experience, the force feedback related to obstacle collision is considered. In this research, the user can only control the end-effectors of Baxter's arms, so the force feedback is for collisions between the end-effectors and the plane, or obstacles on the plane. By using a Kinect camera with the Object Pose Estimation system [31], when objects are on a plane, the information about the plane and each object can be obtained, such as position, orientation, and size, as shown in figure 5.1. That information will consider the shapes of objects as rectangular. On the other hand, the position and orientation of the end-effector can be computed by forward kinematics. The shape of the end-effector is simplified as a sphere, and the center of the sphere is the position of the end-effector. This way, the collision between the end-effector and an obstacle can be seen as the collision from a 3D sphere to a 3D rectangle.

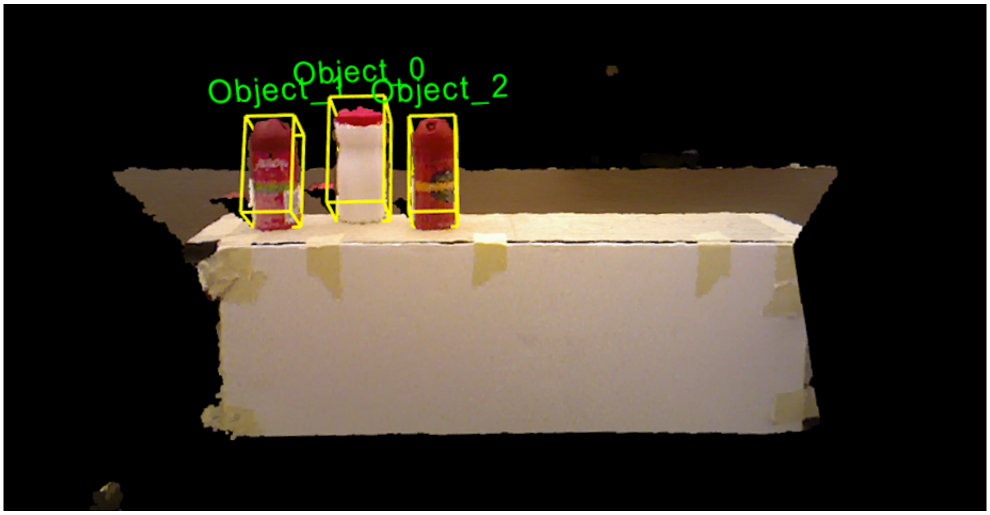


Figure 5.1: Scene in Object Pose Estimation System.

Because the collision can be simplified to the collision of a sphere and a rectangle, the distance between the sphere and the rectangle will be considered. If the shortest distance, D, from the center of the sphere to the surface of the rectangle is equal to or less than the radius, r, of the sphere, the collision is recognized. If the collision happens when the center of the sphere is above of any plane of the rectangle, the force feedback, F, will be perpendicular to the surface of the rectangle. Depending on the orientation of the arms, the force vectors will be applied on the haptic controller. For instance, in figure 5.2, a sphere is closing to the xy-plane of a rectangular object. The position of the object in three dimensional Cartesian coordinate is (P_x, P_y, P_z). The position of the corners on the xy-plane can be compute as:

$$P_{x,\pm corner} = P_x \pm P_{x,size} \tag{5.1}$$

$$P_{y,\pm corner} = P_y \pm P_{y,size} \tag{5.2}$$

$$P_{z,surface} = P_z + P_{z,size} \tag{5.3}$$

Here, $P_{x,size}$ is the distance from yz-plane to P_x along x-axis, $P_{y,size}$is the distance from xz-plane to P_y along y-axis, and $P_{z,size}$ is the distance from the xy-plane to P_z along z-axis. The positions of the four corners are ($P_{x,+corner}, P_{y,+corner}, P_{z,surface}$), ($P_{x,+corner}, P_{y,-corner}, P_{z,surface}$), ($P_{x,-corner}, P_{y,-corner}, P_{z,surface}$), and $(P_{x,-corner}, P_{y,+corner}, P_{z,surface})$. If the end-effector of the arm is above of the xy-plane, the shortest distance between the xy-plane and the arm should be calculated.

$$\begin{gathered} if\,[\,\left(P_{x,-corner} \le P_{arm,x} \le P_{x,+corner}\right)\, and\, \left(P_{y,-corner} \le P_{arm,y} \le P_{y,+corner}\right)] \\ D_{xy.plane} = P_{arm,z} - P_{z,surface} \end{gathered} \tag{5.4}$$

Here, $P_{arm,x}$, $P_{arm,y}$, and $P_{arm,z}$ are the positions of the end-effector of the arm on x-axis, y-

axis, and z-axis, respectively. $D_{xy.plane}$ is the distance between the xy-plane and the arm along z-axis. If $D_{xy.plane}$ is equal to or smaller than r, the force feedback will be given. And the force feedback will be perpendicular to the xy-plane.

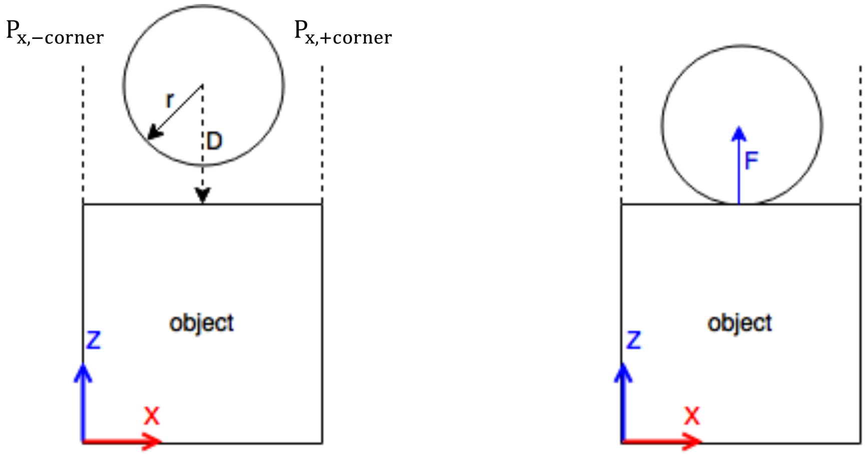


Figure 5.2: The Distance Between the Center of Sphere and the Surface of Rectangle.

If the collision happens when the center of the sphere is not placed above any plane of the rectangle, the collision can be considered happening at the sides or corners of the rectangle. The force feedback vector is applied via the ratio of the relationship between the shortest distance between every plane, side, or corner of the rectangle and the center of the sphere. For example, in figure 5.3, the center of the sphere is not above the plane of the rectangle, but near a side of the rectangle, which is along the y-axis. In this example, the side is called y-corner. When the shortest distance, D, equals r, the vector of the force feedback is applied as the vector between the center and the corner:

$$P_{z,\pm corner} = P_z \pm P_{z,size} \tag{5.5}$$

$$\begin{gathered} if\ [\ (P_{x,+corner} \leq P_{arm,x})\ and\ (P_{y,-corner} \leq P_{arm,y} \leq P_{y,+corner})\ and \\ (P_{z,+corner} \leq P_{arm,z})\ ] \end{gathered} \tag{5.6}$$

$$D_{y.corner} = \sqrt{\left(P_x - P_{x,+corner}\right)^2 + \left(P_z - P_{z,+corner}\right)^2} \tag{5.6}$$

Here, $P_{z,+corner}$ is a position on the side of the rectangle on the z-axis, and $D_{y.corner}$ is the shortest distance between the position of the end-effector of the arm and the y-corner. Once the collision happens, the haptic device will receive force feedback commands to give correct force feedback. The force vectors are implemented as the ratio of the distances on the x-axis and the z-axis.

$$\begin{gathered} if\ \left(D_{y.corner} \le r\right) \\ F_x = F \times \frac{\left|P_x - P_{x,+corner}\right|}{D_{y.corner}} \\ F_z = F \times \frac{\left|P_z - P_{z,+corner}\right|}{D_{y.corner}} \end{gathered} \tag{5.7}$$

Here, F_x and F_z are the force vectors along the x-axis and the z-axis, respectively, and F is a total force feedback.

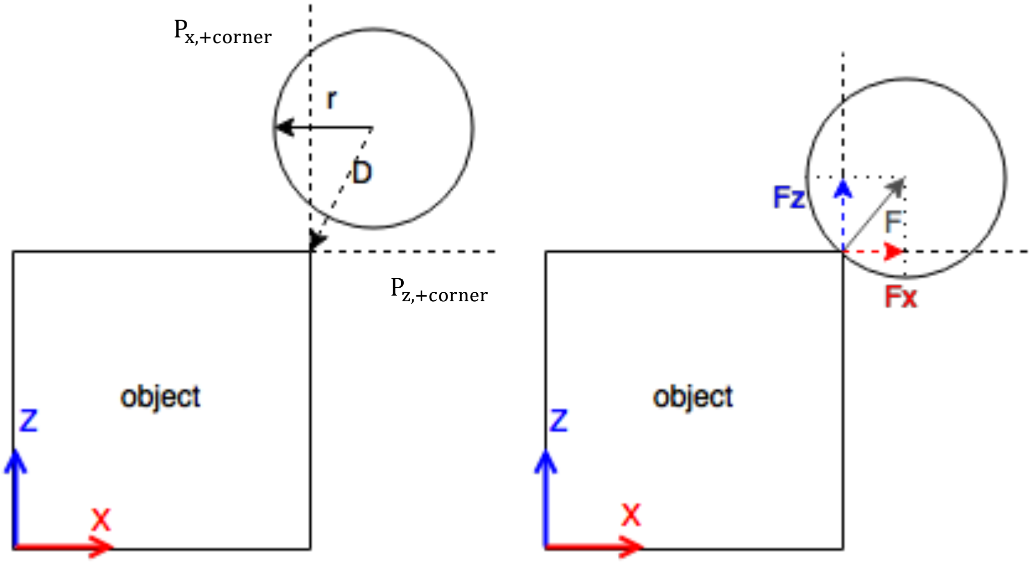


Figure 5.3: The Distance Between the Center of Sphere and the Corner of Rectangle.

From the above descriptions, it is clear that the result of collision recognition does not influence the motion planning, so the collision recognition system is considered an additional system for the program. Even if there is no collision recognition system, the teleoperation program can still run well.

5.2.5 Process of Real-Time Teleoperation Program

Every time, when the user provides the velocities of the end-effector using the haptic device, the velocities are input into the motion planning program. Because the robot stays at the ready pose before the teleoperation, the current pose is known in the motion planning program. After computing, the motion planning program will output the joint commands. The command is a message, which is sent from the client point to server point via TCP/IP. Then, the joint command is sent to the robot. After receiving the command, the robot starts to move to the target, and the robot will send its current angle to the server point and the forward kinematics program to calculate the positions and orientations of the end-effectors. By using the information of the end-effectors and the information about the obstacles from Kinect, the collision recognition can be confirmed. The collision recognition program computes the vectors of force feedback, and sends it to the server point. Combining the current joint angles and force vectors, the server point sends that information back to the client point. Then, the information is separated into its' current joint angle part and force vectors part, which are then sent to the motion planning program to compute the joint angles for the next motion and to the controller to offer the direction of force feedback. These are done in one loop for the teleoperation. This loop keeps repeating when the user is giving the velocities of end-effector using the haptic controller.

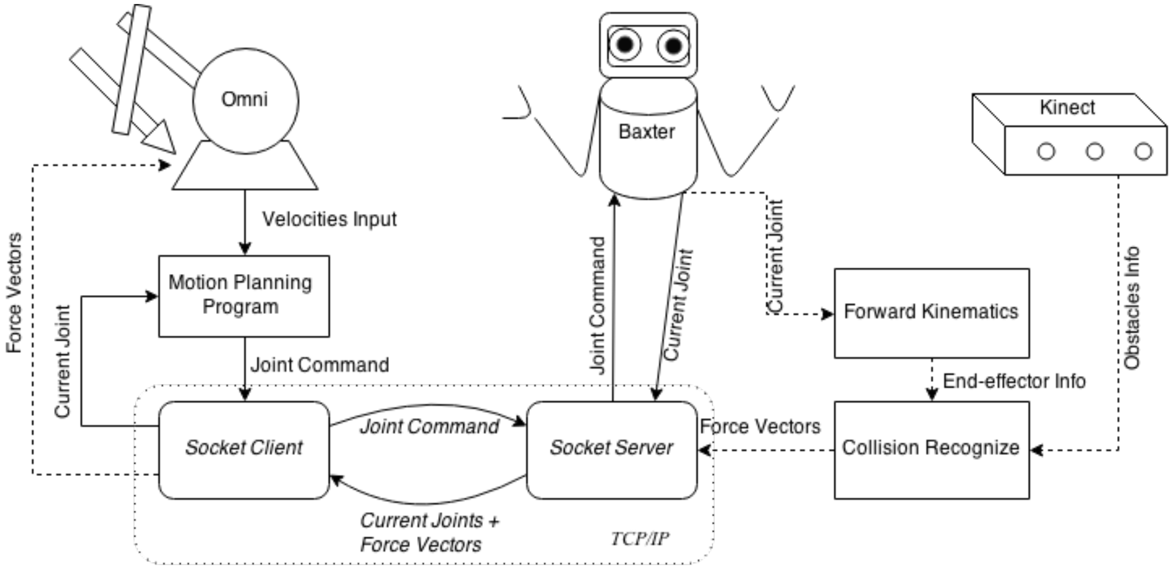


Figure 5.4: Flowchart of Teleoperation System.

5.3 Platform Testing Result Using the Same Simulation Algorithm

The motion planning algorithm should be tested on the real platform before implementing teleoperation. Hence, based on Chapter 5.2, the input resource is switched to keyboard. The user can enter the desired position and orientation. Moreover, the waypoints are required because it is not real-time control. The end-effector can only receive and act on the next command after it finishes its' current processing. In the test, the initial positions and orientations of the right arm in both simulation and platform are:

$$ {}_{7}^{0}T_i = \begin{bmatrix} 0.0000 & 0.0001 & 1.0000 & 32.8072 \\ 0.0000 & 1.0000 & 0.0001 & -64.9976 \\ 1.0000 & 0.0000 & 0.0000 & 9.9978 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.8} $$

and the target are both:

$$ T_g = \begin{bmatrix} 0.2549 & -0.1622 & 0.9533 & 50.0000 \\ -0.0449 & 0.9828 & 0.1792 & -50.0000 \\ -0.9659 & -0.0885 & 0.2432 & -10.0000 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.9} $$

The motion planning results of the simulation is same as the target, which is:

$$
{}^{0}_{7}T_{simulation} = \begin{bmatrix} 0.2549 & -0.1622 & 0.9533 & 50.0000 \\ -0.0449 & 0.9828 & 0.1792 & -50.0000 \\ -0.9659 & -0.0885 & 0.2432 & -10.0000 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.10}
$$

And the motion planning results of Baxter is:

$$
{}^{0}_{7}T_{Baxter} = \begin{bmatrix} 0.2297 & -0.1576 & 0.9604 & 49.9588 \\ -0.0441 & 0.9841 & 0.1720 & -49.8693 \\ -0.9723 & -0.0819 & 0.2191 & -10.4051 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.11}
$$

which is close to the target. Compared to the figures of the command of joint angles in figure 5.5 and 5.6, the slope, and the beginning and end positions are similar. But the joint rates of simulation and physical platform are different because the physical platform cannot reach the required joint rates as simulation does. If initially the joint rates are slower than the required command, the distance from the current position to the next waypoint will be farther than expected. Then, the joint rates will be raised to achieve the next waypoint. However, raising the joint rates may cause some joints to move beyond the required angles. If that happens, then the next step will be slower. In figure 5.7 and figure 5.8, the joint rates of the simulation are smoother than the joint rates of the physical platform. For this reason, in the physical platform, the joint angle commands and the results of joint motion are different. The joint angles on the platform can be found in figure 5.9. Figure 5.10 shows the difference between the commanded values and the resulted values of the angles in the platform. The differences are within 0.6°, which is very small. To sum it up, even though the results of the transformation matrix of the simulation and Baxter have a narrow gap; the results of Baxter are still close to the target. Moreover, based on similar results of the joint angles from the motion planning program in simulation and from the platform, the motion planning on the physical platform is acceptable, which are shown in figure 5.5 and figure 5.9, respectively.

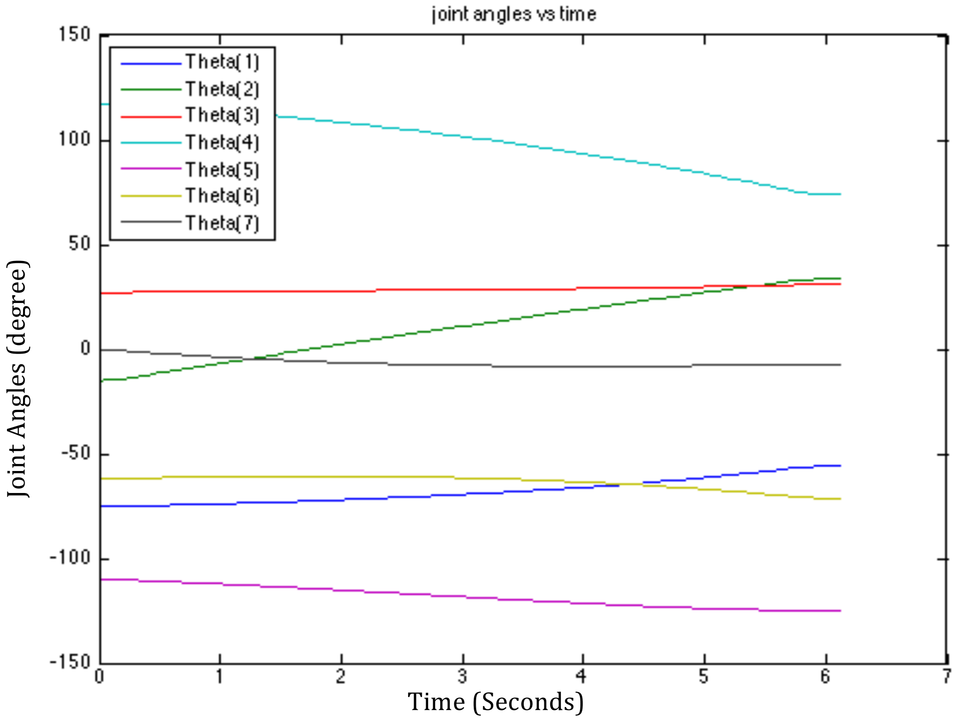


Figure 5.5: Joint Angles in Simulation.

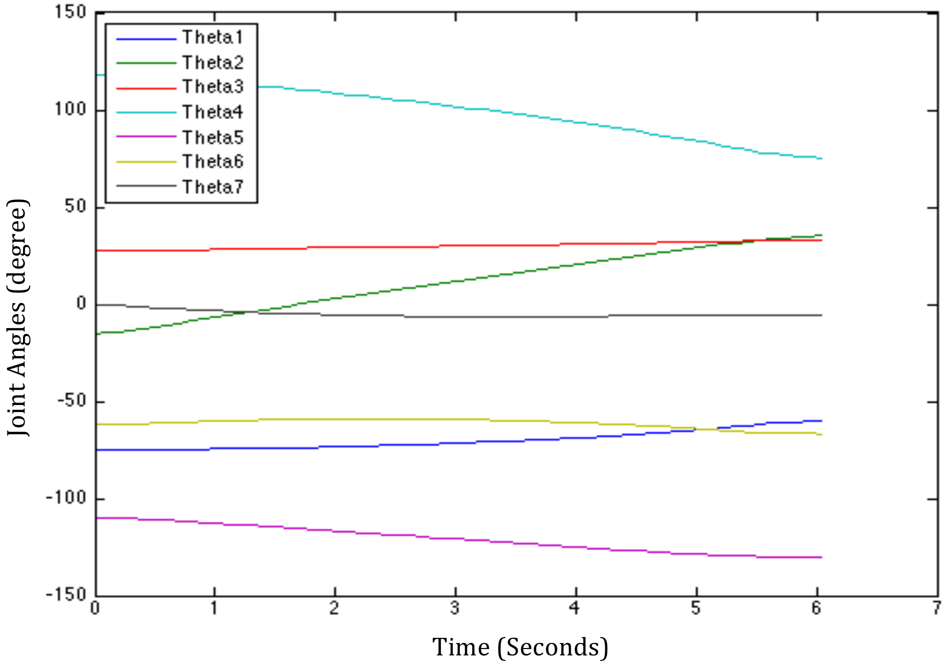


Figure 5.6: Joint Angle Commands on Platform.

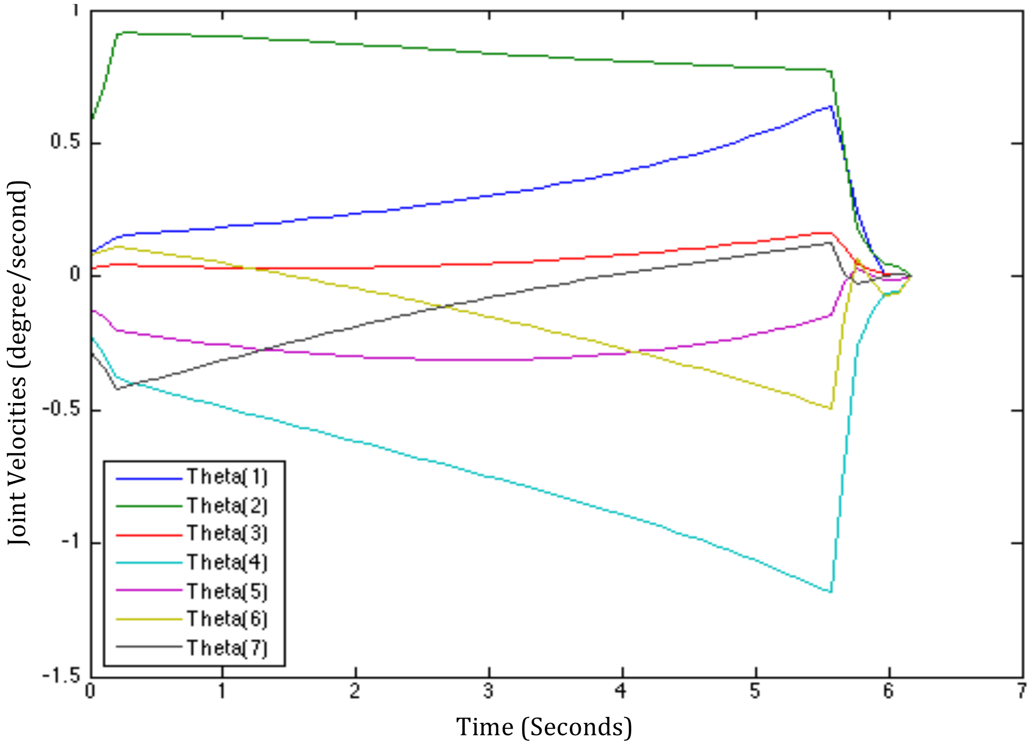


Figure 5.7: Result of Joint Velocities in Simulation

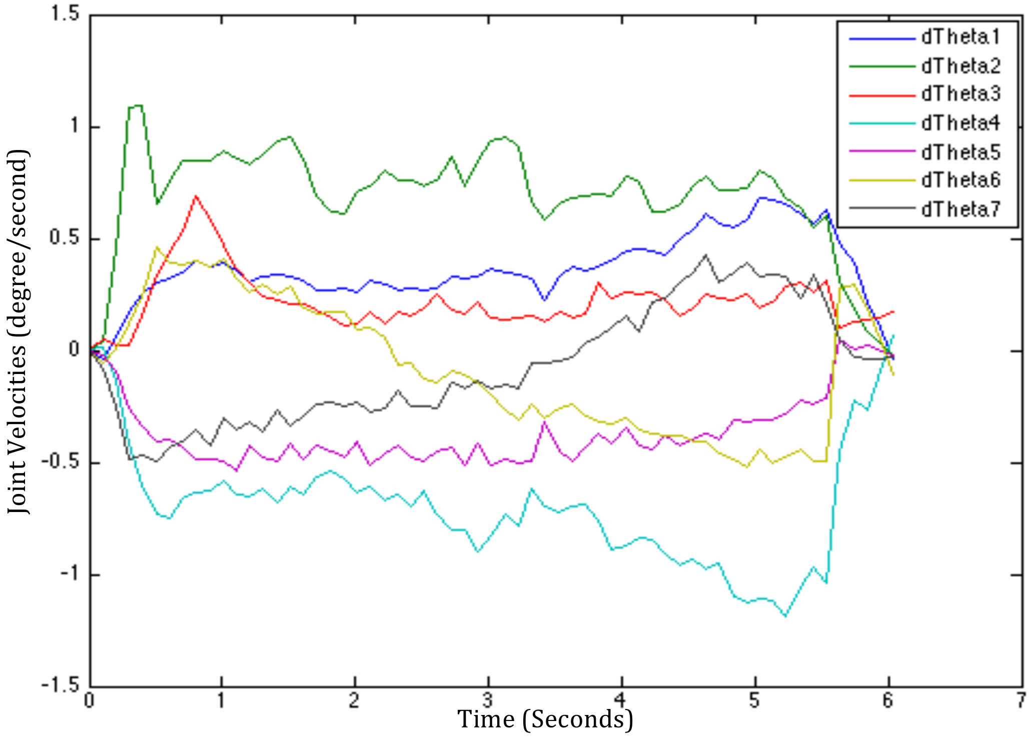


Figure 5.8: Result of Joint Velocities on Platform

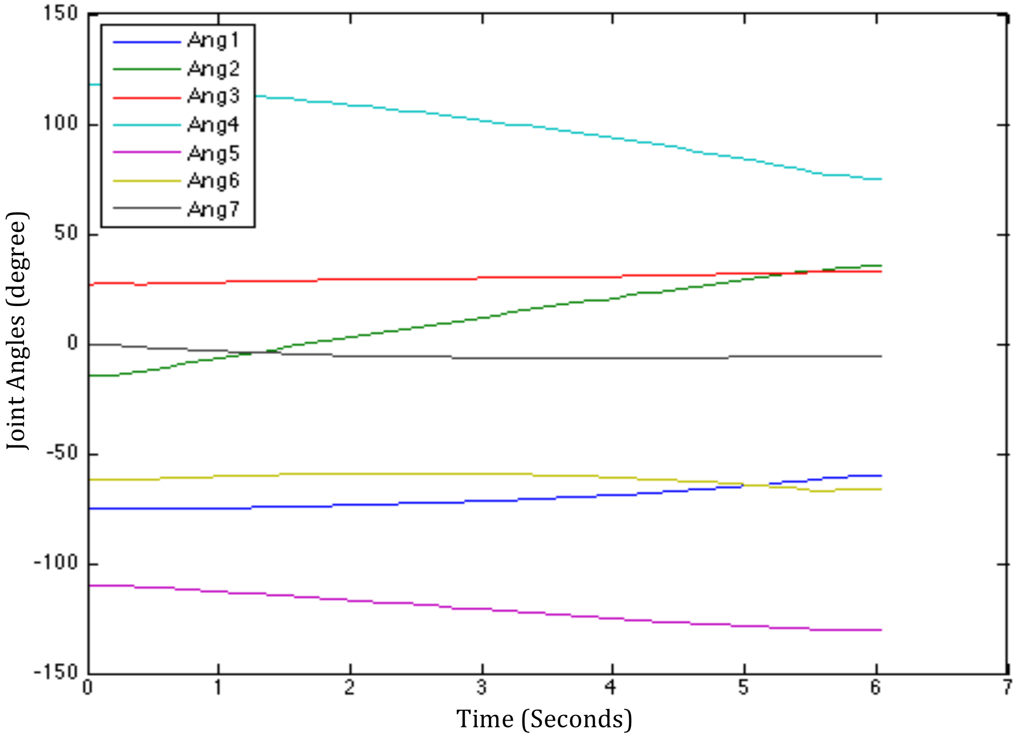


Figure 5.9: Joint Angles on Platform.

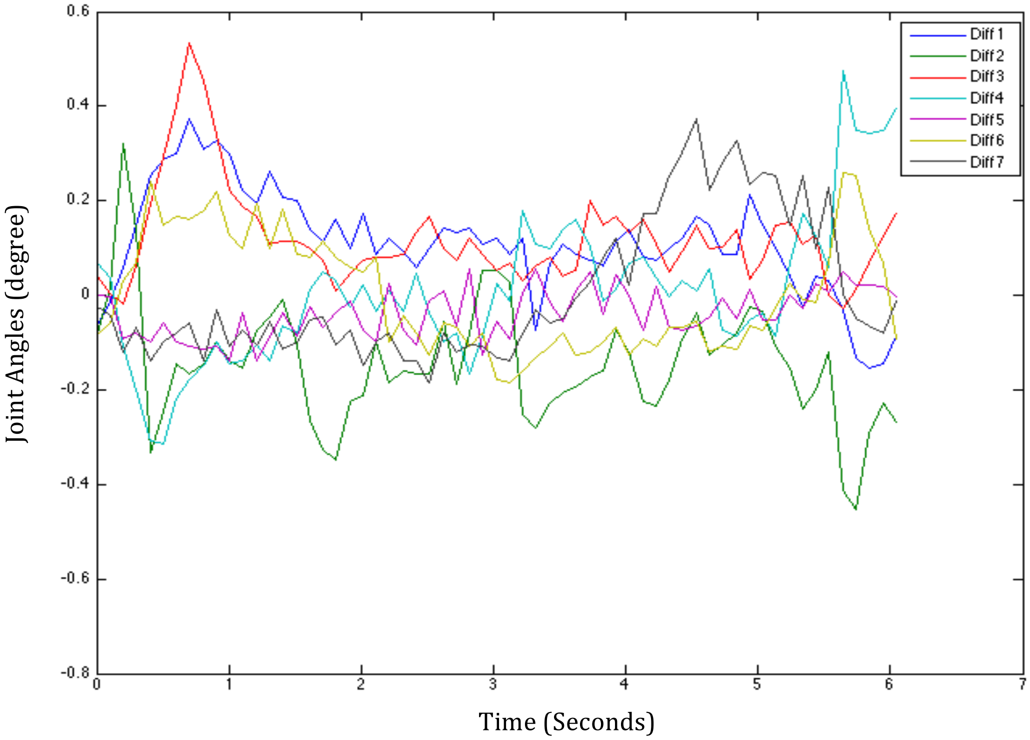


Figure 5.10: Differences Between Joint Commands and Real Joint Angles.

5.4 User Interface

The user interface is important for the user to remotely check the arms. To remote control the arms, the interface should allow the user to input velocities of the end-effectors. The Phantom Omni is a good device to get velocities from, in figure 5.11. There are six joints on the Omni. It can present the position and orientation of its end point in Cartesian coordinates. Moreover, it is also a haptic device, so the user can feel the collision force feedback through it. There are also two buttons on it. The first button is used to launch the arm. Every time, when the user pushes the first button, that position and orientation is set as the initial point, zero. If the user is still pushing the button while moving the Omni, the difference between the current and initial points of the end point's position and orientation will be used as the velocity input. The second button is set as a controller for closing or opening the gripper. The user only needs to click once to close or open the gripper.

Figure 5.12 shows the monitor screen as part of the user interface. We installed three webcams that we attached to Baxter instead of the three original cameras, that is because the video frame rates of the original cameras are not acceptable with high resolution. As shown in figure 5.13, there are three webcams, two attached on both end-effectors and one on Baxter's head, whose direction can be controlled by the keyboard. Hence, through the webcams, the user can see the video of the robots' environment on their monitor. On the monitor screen, the real time model of Baxter is also displayed. From this model, the user can check the pose of Baxter, and control it with high efficiency. Because the webcams can only offer 2D video, and collision force feedback happens when the end-effector is very close to the obstacles, the IR sensor is offered for the user to realize the distance from the end-effector to any obstacles. The data of the IR sensor is shown on the user's screen as bars. As the obstacles get closer to the end-effector, the bar raises. In contrast, the bar goes lower when the end-effector is farther from the obstacle.

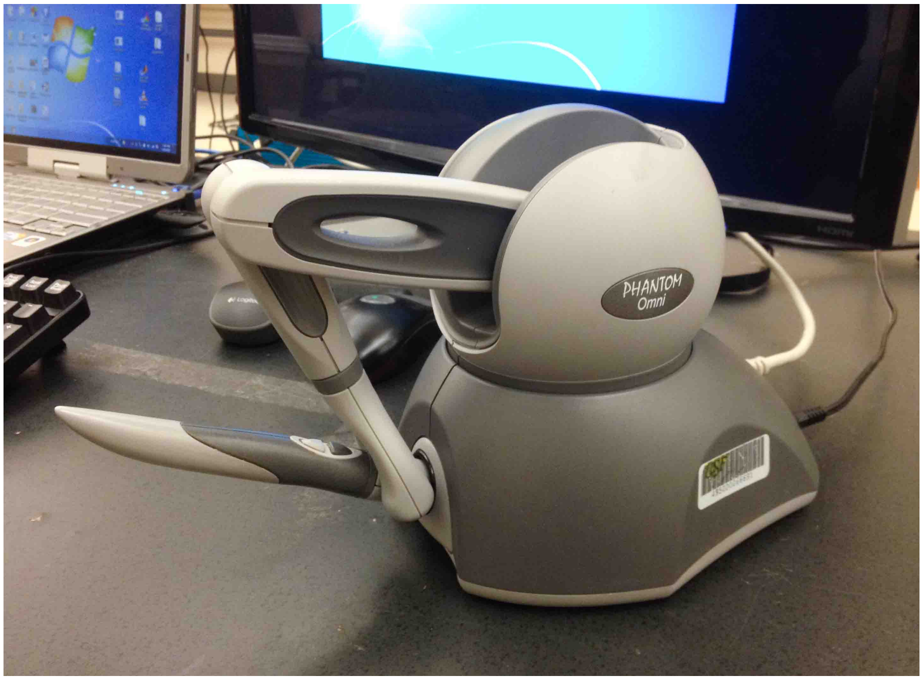


Figure 5.11: Image of Phantom Omni

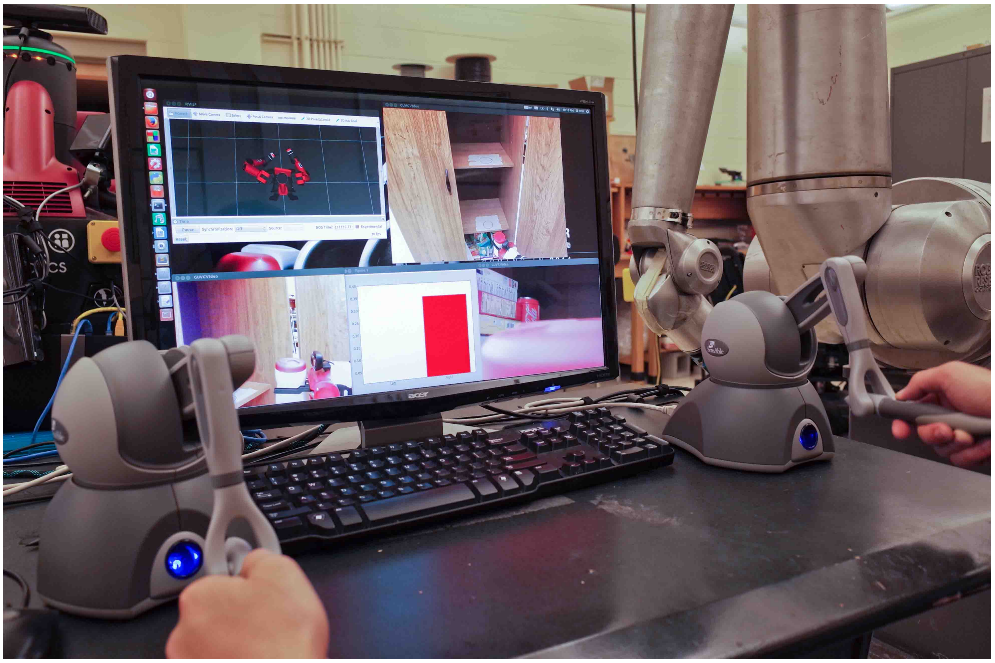


Figure 5.12: Two Omni's and the User Interface on the Monitor.

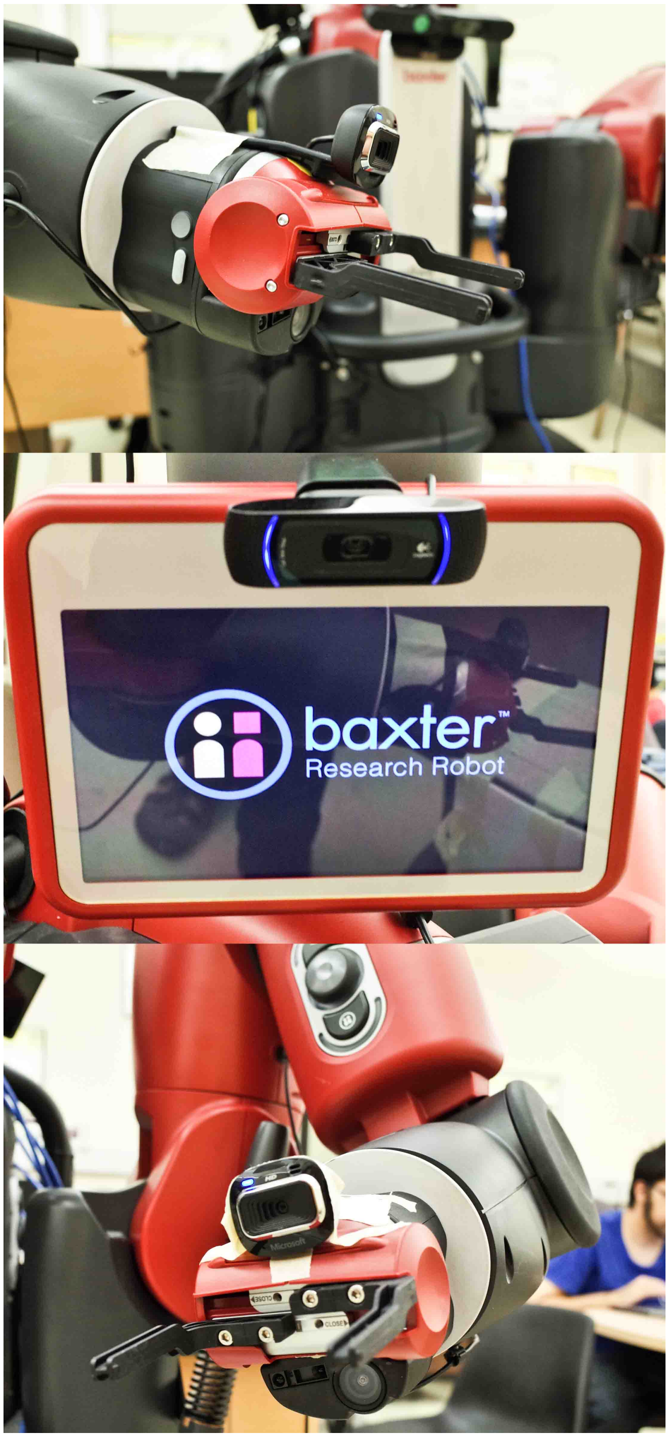


Figure 5.13: Webcams on Baxter's Right Arm (Top), Head (Middle), and Left Arm (Bottom).

5.5 Result of Teleoperation on the Physical Platform

The remote control algorithm was tested after building the whole teleoperation system. At the beginning of the test, as time went by in teleoperation, Baxter's arms started to vibrate because the update frequency of Omni was higher than that of Baxter. If the frequency of Omni is higher, new commands come to Baxter before Baxter completes its previous commands. When Baxter cannot complete the previous commands, the gap between the new commands and pervious commands will be increased. As time goes by, the increased gap needs higher velocities to achieve. Hence, joint rates become higher. Once the joint velocities go too fast, accuracy of Baxter's arms motion are reduced because the joint may move more than the commanded value. Those relative problems cause the vibration. Once the frequency of Omni was reduced, the position and orientation of the end-effectors of Baxter's arms could follow the joint commands from the Omni with good accuracy. Furthermore, with the collision feedback system, the directions of force feedback were correct, and the force feedback was clear. The results of teleoperation using Baxter was acceptable.

5.6 Summary

Many details had to be discussed in order to build the teleoperation system after the simulation was done. In this chapter, the control system is extended from the motion planning system from simulation. The motion planning system was then tested on the platform. After that, the teleoperation algorithm was discussed and built. Moreover, the additional collision force feedback system was created with the OPE system. The system can offer the position and size of objects detected on a plane. The information allows the control system to recognize when a collision is occurring. The user interface is also an important piece in the teleoperation system. The interface includes the haptic controllers, the information of the pose of Baxter, distance from the IR sensors, and video from the robot's environment. The result of the teleoperation system was as expected. Baxter's arms could be remote controlled well, and the collision force feedback was correct and clear.

CHAPTER 6: HUMAN SUBJECTS TESTING

6.1 Introduction

In this chapter, the teleoperation system will be tested on human subjects to understand the usability of the control system. There are three teleoperation tasks designed for human subjects. The subjects have been asked to finish the tasks while being timed. The details and results of the tasks will be discussed in this chapter.

6.2 Design of Human Subjects Testing

After discussing the control method and building the real-time teleoperation algorithm, the teleoperation program with the control method is tested in three real applications. The experiments are designed for Baxter as an assistance to finish common works in daily tasks. The experiments are also focused on the flexible cooperative applications of dual manipulators. The users can decide to use single or both arms to complete the tasks. The difficulty of the applications increased in series from the first to the last test.

The first test is called “grip-a-bottle”. There are two tasks in the first experiment. The First task, which is an easy test, is to grasp a bottle from the initial position on a table with one robot arm, and pass it to the other arm. Then by controlling the second arm, the user will place the bottle into the desired destination area on the table. The scene of the first task is shown in figure 6.1. The blue circle in the figure is the desired destination area. This task is done to test the efficiency for passing items.

A second task is extended from the first task. Based on the first task, two red bottles are added as obstacles. The red bottles, nearby the object, are placed to increase the difficulty, as shown in figure 6.2. The time to finish both of the tests was recorded, and the number of collisions happened was also noted to compare the efficiency with and without obstacles. Figure 6.3 shows the moment when the bottle is being passed.

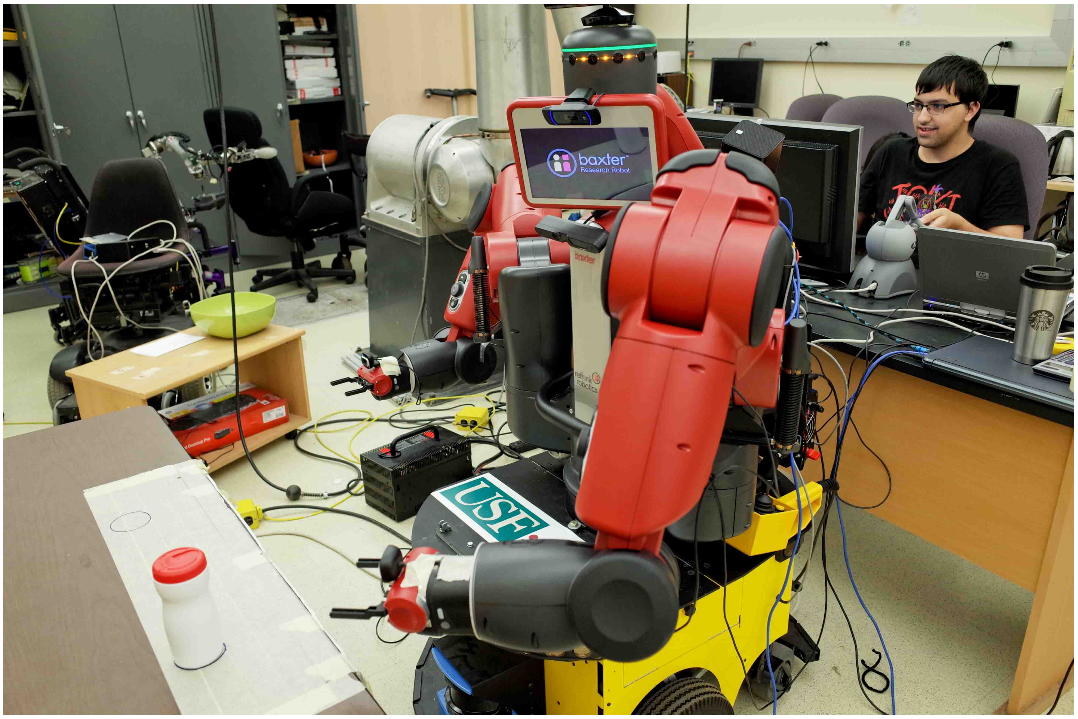


Figure 6.1: First Test without Obstacles.

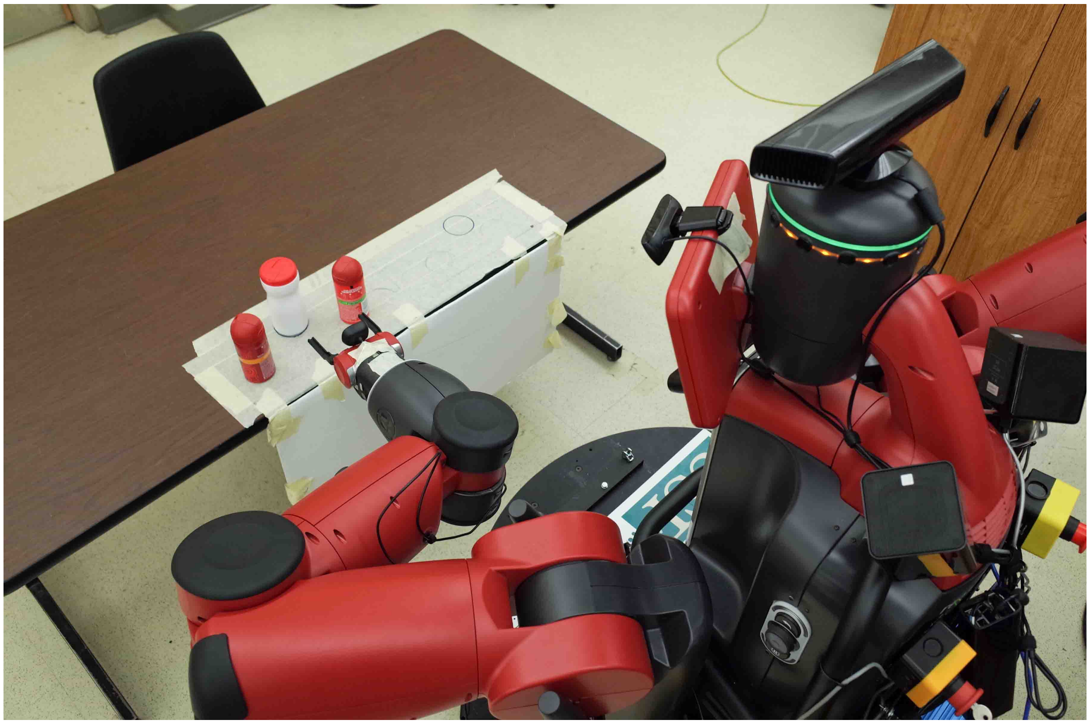


Figure 6.2: First Test with Obstacles.

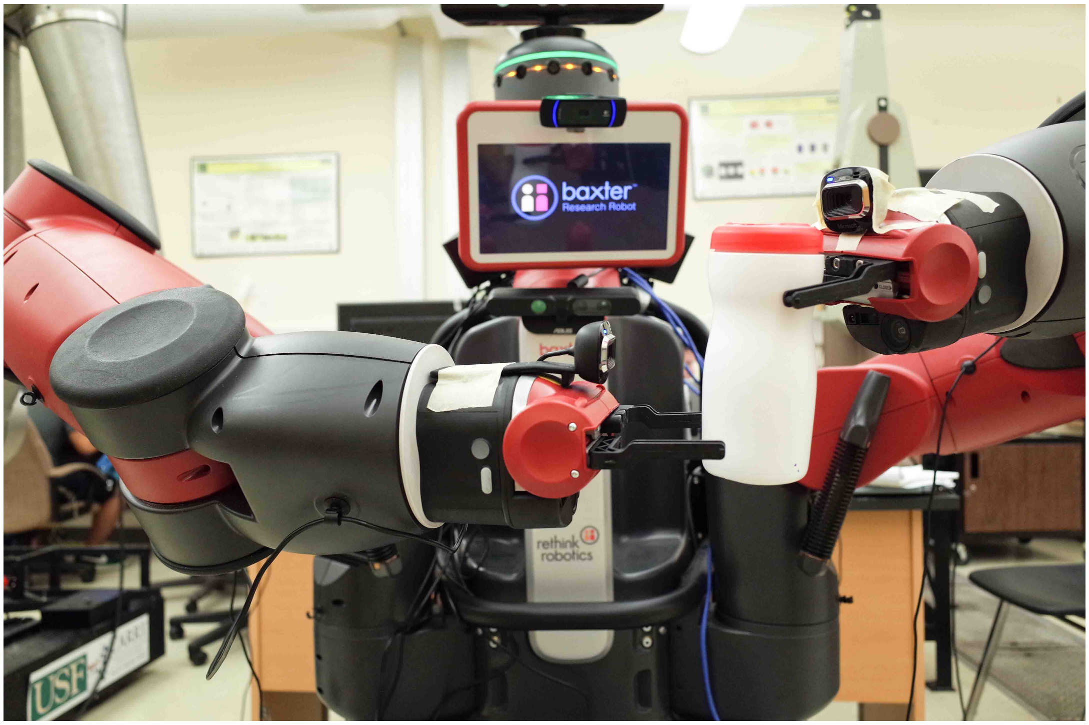


Figure 6.3: Baxter Is Passing a White Bottle to the Other Arm in the First Test.

The second experiment is called “open-a-cabinet”. In this test, the users have to remotely control the robot to open a door of a cabinet. Because the shapes of the grippers are hard to grasp the handle of the door tightly, a small hook was attached to the gripper of the right arm to hold the door handle tightly. The hook and the door of the cabinet are shown in figure 6.4. After the door is opened, the user controls any one of the manipulators to grasp the bottle inside, and place it in the desired area on the higher shelf of the cabinet. Figures 6.5 and figure 6.6 show the opening of the door and the structure of the cabinet while the experiment is in progress. This test is harder than the first experiment. The point for this application is to test the ability of the user to remotely control Baxter with more complex trajectories. The numbers of collisions and the time to complete the task were noted.

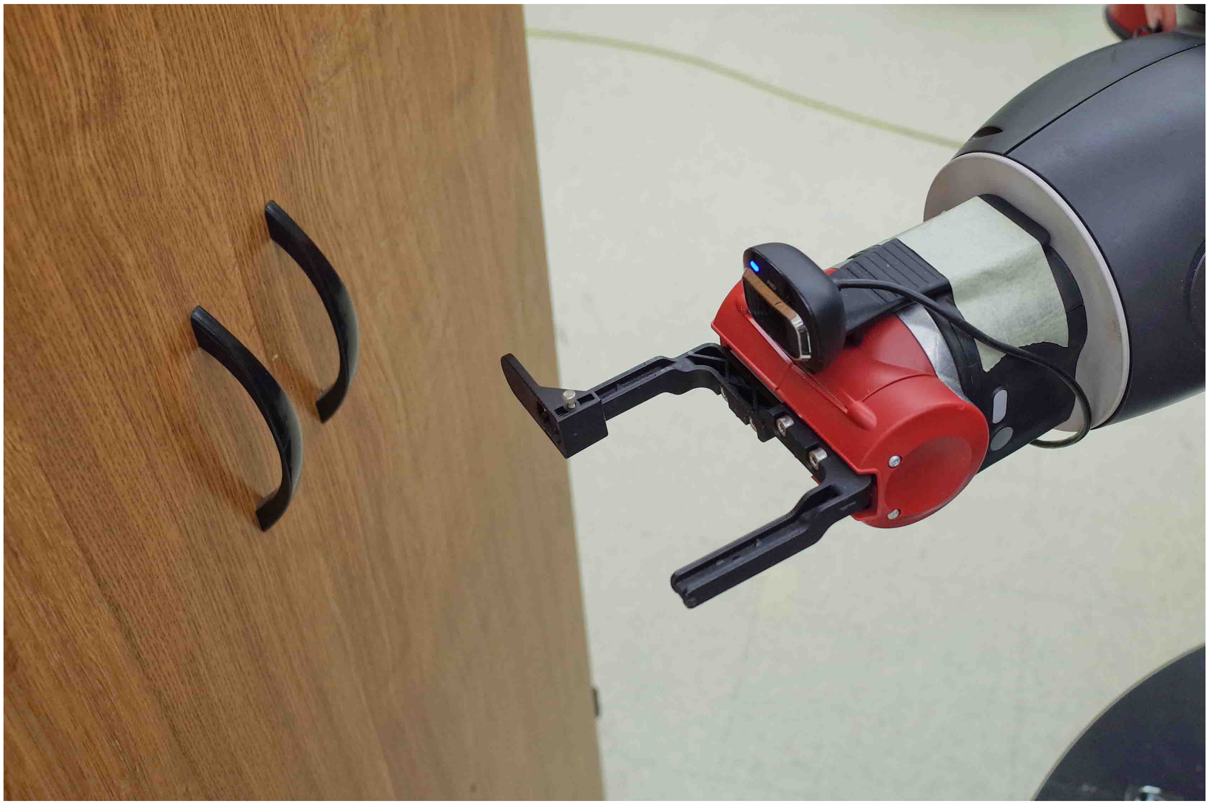


Figure 6.4: A Hook on Gripper

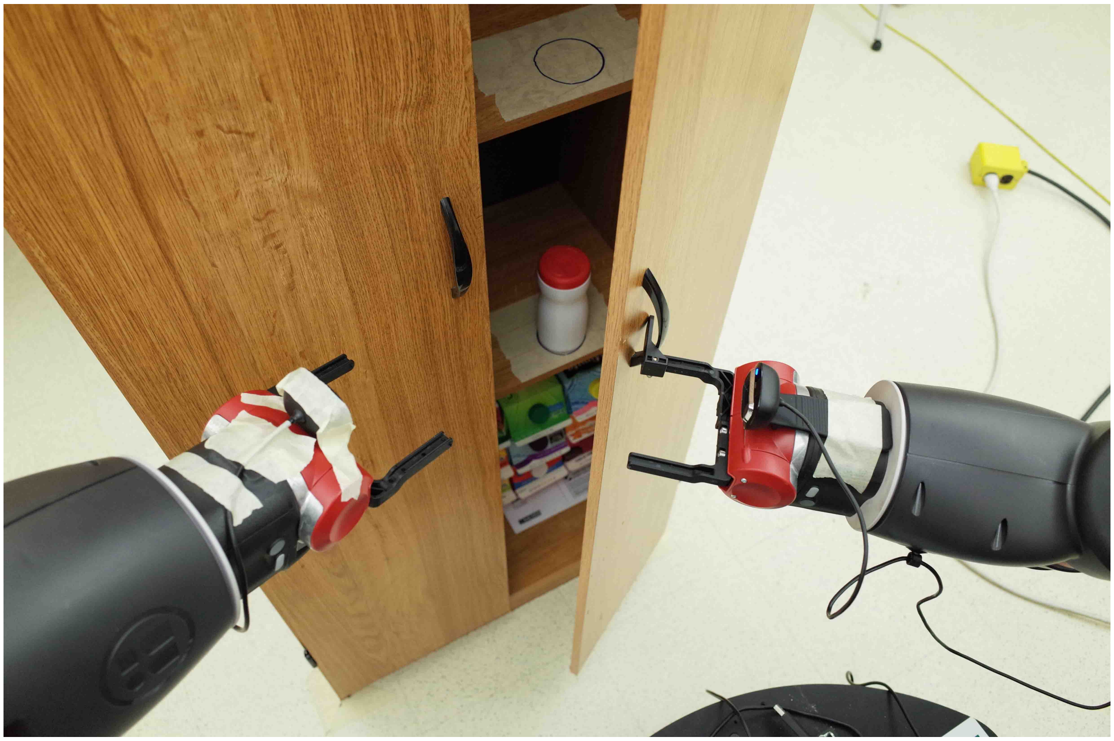


Figure 6.5: Baxter Is Opening the Door of a Cabinet.

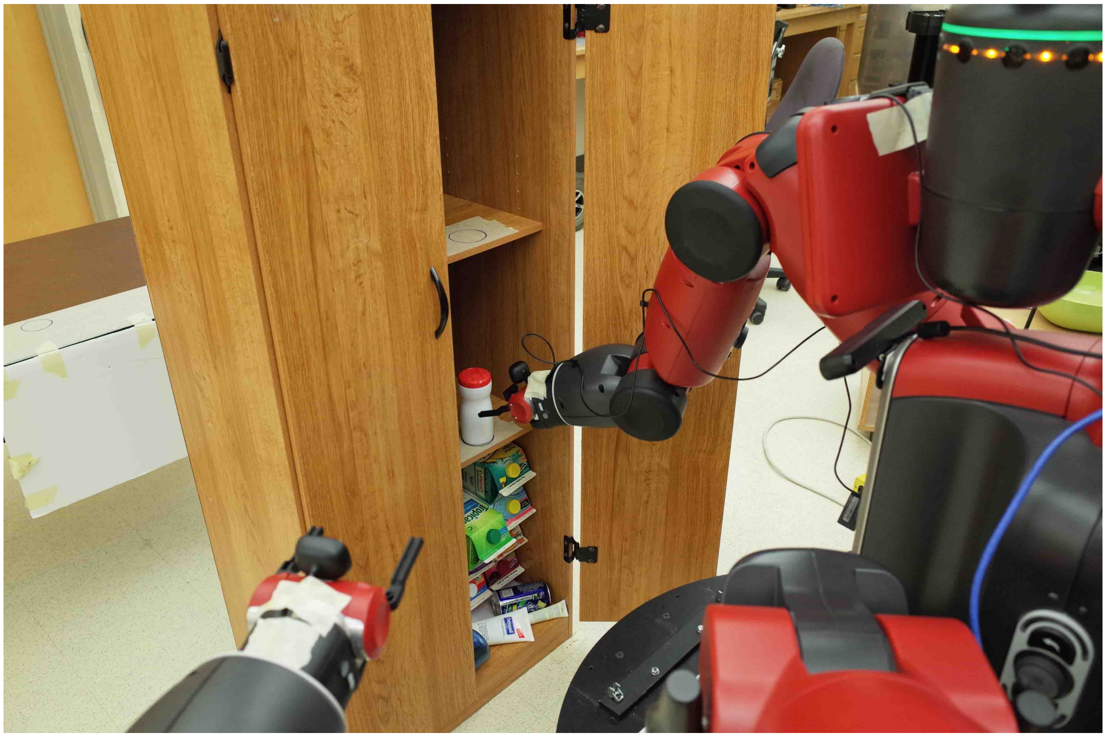


Figure 6.6: Structure of the Cabinet in the Second Test.

The last test is called "place-a-basket-on-bookshelf", which is the most complicated to achieve of all the experiments. At the beginning of the test, there is a bottle on top of a bookshelf, and a basket is placed in front of the bookshelf. In this application, the users are asked to control the arms to grasp the bottle and then place it into the basket. Then, they use both Baxter's arms to raise the basket with the bottle inside, and place them on the top layer of the bookshelf, as shown in figure 6.7. Figure 6.8 shows the environment of this test. In the test, if either the bottle or the basket falls, then the user fails the task, and the task has to be restarted. The amounts of failures are noted in order to compute the percentage of success. Moreover, the number of collisions and the time to complete the test are noted. Figure 6.9 shows the raised basket with the bottle inside.

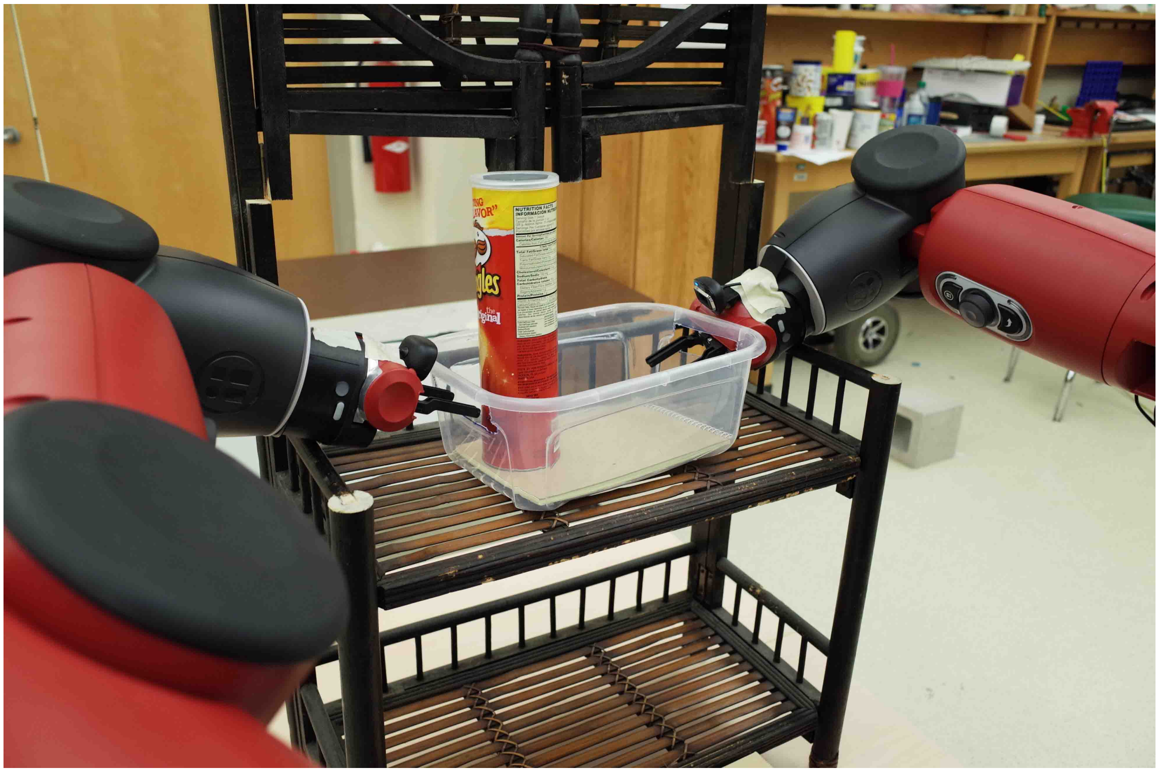


Figure 6.7: Raising Basket with Bottle, and Place on Bookshelf.

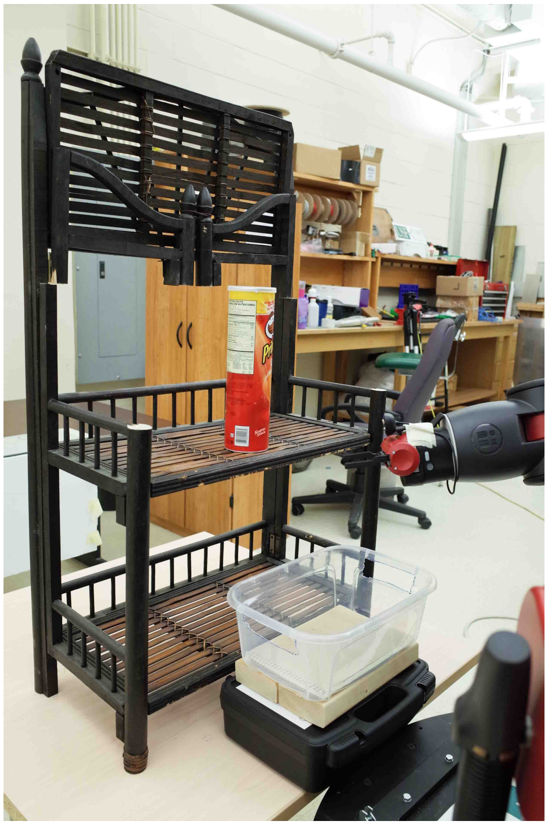


Figure 6.8: Environment of Third Test.

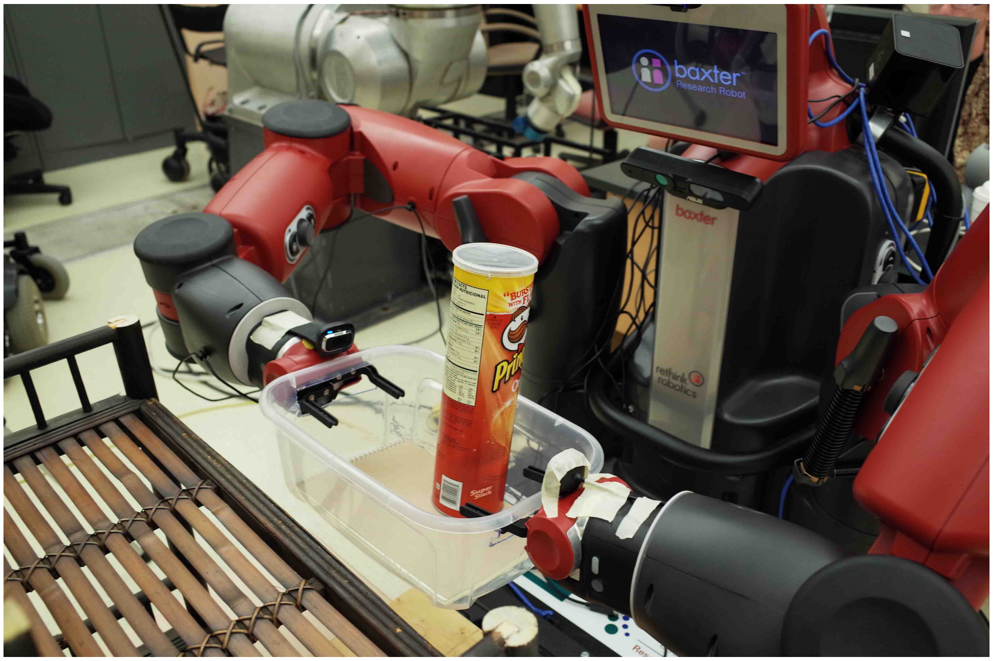


Figure 6.9: Raising a Basket with a Bottle Inside.

6.3 Result of Human Subjects Testing

There were 22 healthy subjects for these experiments. The age range is between 19 and 29 years old. They were asked to complete each test independently. 14 of them are male and eight of them are female. Eight of the subjects had more than four hours experience in this teleoperation system. The rest of them who do not have experience had 40 minutes of training before the tests. The number of collisions and the time taken to complete each test were noted. And after the experiments, the users were asked to give scores (from one to ten, where one is the worst and ten is the best) in a survey and give some comments about the user experience and control program.

In the "grip-a-bottle" test without obstacles, the users usually had difficulty in passing the bottle, and in the test with obstacles, the users also took longer time to avoid the obstacles and grasp the bottle, basically. For the second test, "open-a-cabinet", the users had a challenge in opening the cabinet door. But they were still able to finish the test. In the third test, "place-a-basket-on-bookshelf", the users struggled to grip the basket and raise it up to the bookshelf with both manipulators at the same time.

The average time to complete the "grip-a-bottle" test without obstacles was 331.33 seconds, the "grip-a-bottle" test with obstacles was 372.83 seconds, the "open-a-cabinet" test was 460.65 seconds, and the "place-a-basket-on-bookshelf" test was 503.17 seconds, as shown in figure 6.10. The error bars of histogram in figure 6.10 and the following figures are standard errors of sample mean. However, the average time between test 1 without obstacles and test 1 with obstacles, and test 2 and test 3 are close. By using t-test to analyze the differences among those groups, there are no statistically significant difference between test 1 without obstacles and test 1 with obstacles ($p = 0.2925$), and test 2 and test

3 ($p = 0.2521$), but there are significant differences between both tasks of test 1 and test 2, or both tasks of test 1 and test 3. The results of the t-test indicate that the times to complete test 1 without obstacles and test 1 with obstacles are equal for the users, and the times to finish test 2 and test 3 are the same.

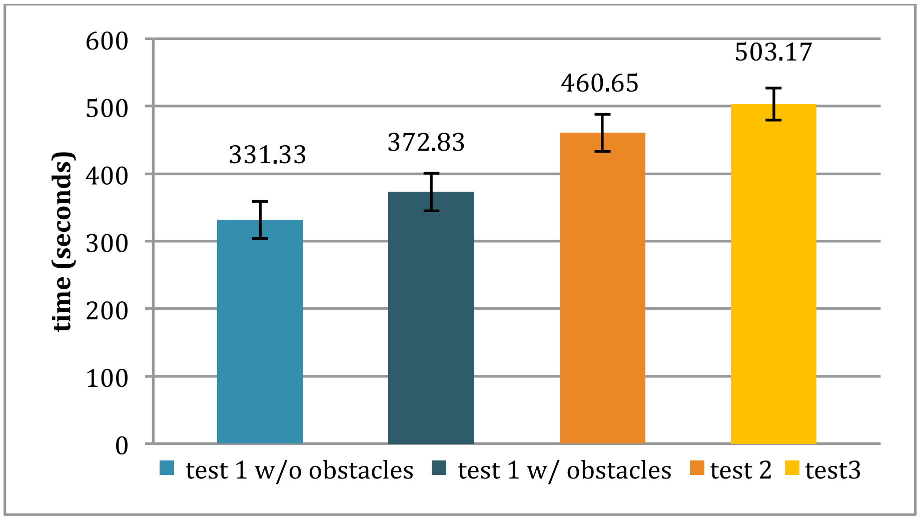


Figure 6.10: Average Time to Finish Different Tests

In the survey, the average scores of the difficulty from the first task to the last task were 3.82, 5.05, 6.64, and 7.5, respectively, as shown in figure 6.11. Because of the increase in difficulty of the test from easy to hard, the average scores of the difficulty from the first task to the last task were increasing. The difference between the average time to complete each test in the t-test, and the average scores of the difficulty between test 1 without obstacles and test 1 with obstacles, is statistically significant difference ($p = 0.0348$). That means the existing obstacles influence users' feelings about test 1. But there is no significant difference between the difficulty of test 2 and test 3, and $p - value$ is 0.1674 in the t-test. The results of the t-test show that even the completion times of the two tasks of test 1 are equal, the users still consider their difficulties to be different. The results of the completion times and difficulties of test 2 and test 3 are identical in the t-test.

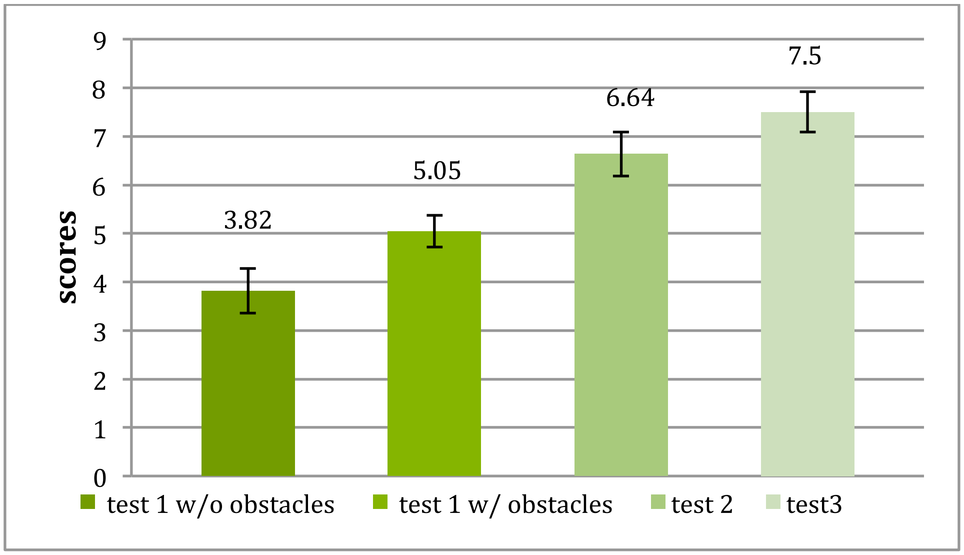


Figure 6.11: Average Scores for Difficulty in Different Tests.

Figure 6.12 shows the comparative time with experienced and non-experienced users. With analysis of variance (ANOVA) [33], the average time to complete each test is the response vector, and the experience is the grouping variable (factor). The $p - vaules$ of ANOVA is 0.008, 0.0037, and 0.0278 in test 1 without obstacles, tset1 with obstacles, and test 3 respectively, which means that the mean responses for the average time to complete the tests of the experience are significantly different. The $p - vaule$ in test 2 is 0.1358, which indicates that the mean response for test 2 of the factor, experience, is not significantly different. Using the ANOVA, it is clear that the users who had existing experience took a shorter average time to complete both test1 and test 3 than the users who did not have experience. But the average time to complete test 2 for both with the experienced users and the non-experienced users is similar because the users had difficulties finding the pivot of the door of the cabinet. The cameras cannot provide enough information of the door of the cabinet here.

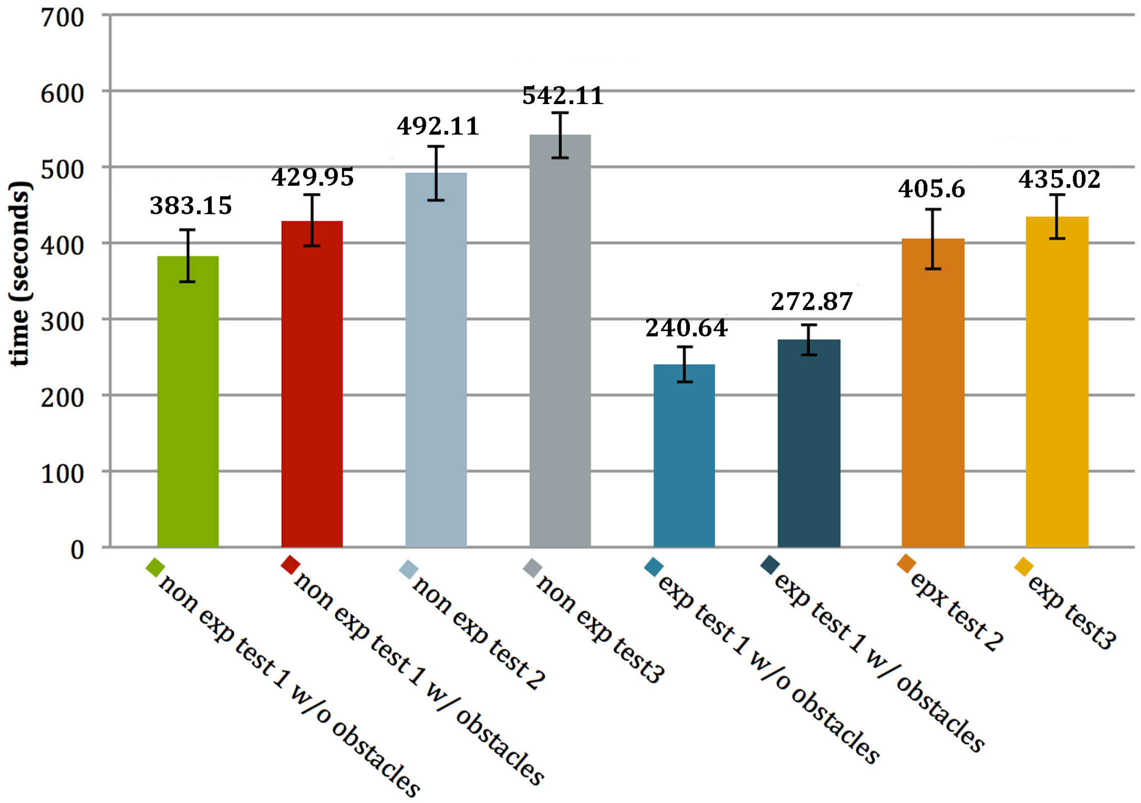


Figure 6.12: Comparative Time to Experienced (Right Part) and Non-Experienced (Left Part) Users.

Similar to the average time to complete the tests, there are no statistically significant differences for the average number of collisions between test 1 without obstacles and test 1 with obstacles ($p = 0.3574$), and test 2 and test 3 ($p = 0.6048$) in the t-test. Figure 6.13 shows the histogram of the average number of collisions in each task. Figure 6.14 shows the average number of collisions for experienced and non-experienced users in different tests. Compared to the collision results for the number of experienced and non-experienced users, there are no statistically significant differences in test 1 without obstacles, test 1 with obstacles, and test 2 with ANOVA. The $p - vaule$ of the ANOVA are 0.2102, 0.4255, and 0.2825, respectively. Those values suggest that experience does not help the users to reduce the collision numbers in both of test 1 and test 2. But the $p - vaule$ of ANOVA in test 3 is 0.0095, which means that experience helps the users to reduce the collision numbers when they are doing test 3.

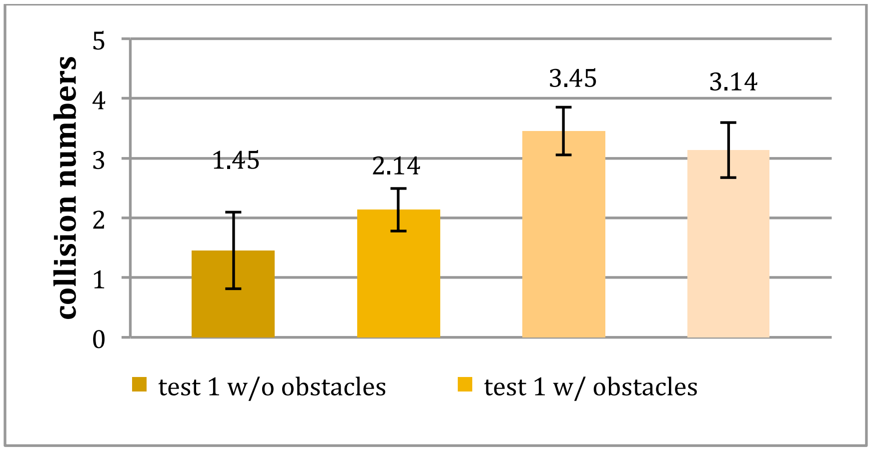


Figure 6.13: Average Number of Collisions in Different Tests.

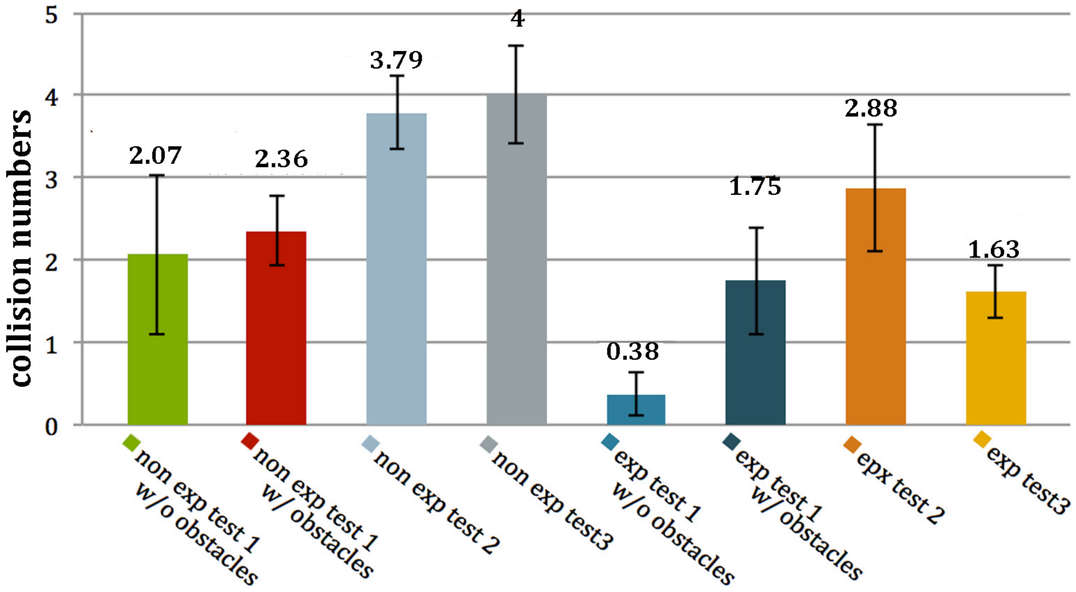


Figure 6.14: Comparison of the Average Number of Collisions of Experienced and Non-experienced Users.

In the "place-a-basket-on-bookshelf" test, the amounts of failures were noted. The results of the last test are shown in figure 6.15. On average, every user fails 1.14 times before succeeding. The experienced and non-experienced users spent 0.5 and 1.5 failures before they completed the last test on average, respectively. Compared to the failure number of experienced and non-experienced users in ANOVA, there is no statistically significant difference ($p = 0.0966$).

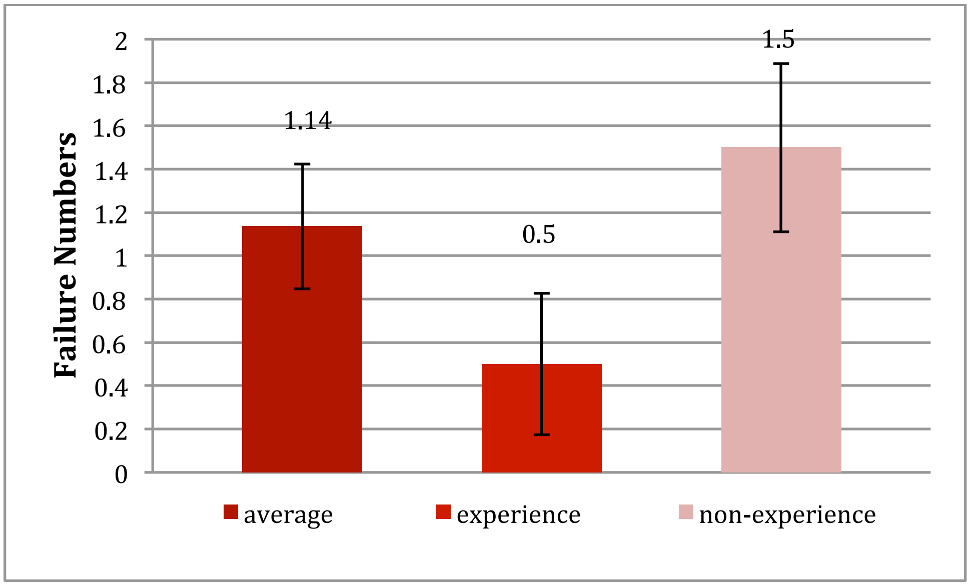


Figure 6.15: Average of Failure Numbers.

In the survey, the questions of ease of use and ease to learn got 6.86 and 7.68 points, respectively. The users gave the interface design 7.54 points in average. And the amount of users' frustration got 4.64 points in average. For the interface, the users thought the force feedback and IR sensor did not give much help in the test. Each of them got 6.23 and 6.95 points in average, respectively. But the cameras at the end-effectors and the cameras on the head did help much in teleoperation. Their average points were 9.27 and 9.14, respectively.

Over all, most parts of the system received positive responses. The results of the surveys are shown in table 6.1.

Table 6.1: Result of the Survey of Human Subjects Testing (From 1 to 10, Lightest to Strongest)

	Easy to use	Easy to learn	Interface	Frustration
Average Points	6.86	7.68	7.55	4.64
	Force feedback	IR sensor	Arm camera	Head camera
Average Points	6.22	6.95	9.27	9.14
	Difficulty of test I w/ obstacles	Difficulty of test I w/o obstacles	Difficulty of test II	Difficulty of test III
Average Points	6.86	5.05	6.64	7.5

6.4 Summary

The human subjects testing of the teleoperation algorithm was tested in three daily life applications. The time to complete the tests and number of collisions were noted. The completion time and collision number was analyzed through the t-test and ANOVA. Through the record, it was clear that users with more experience of teleoperation provided shorter time to complete test 1 and test 2, and better results of collision in test 3 on Baxter. The results of the survey were discussed. Through completion of the testing, the teleoperation algorithm was satisfactory to complete the daily common tasks.

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

7.1 Overview

An optimized control method for two 7-degree-of-freedom arms was discussed in this thesis. Baxter robot is the platform used for the hardware implementation of this work. The simulation program of the arms was built in Matlab. With the simulation program, the motion planning process was tested and checked. The motion planning program was utilized to develop a real-time teleoperation system for Baxter robot. The user interface was also built for this teleoperation system. In the teleoperation system, collision force feedback was added to the control program. The teleoperation system can work well with or without the collision force feedback program. Finally, the simulation system was tested in three different applications.

7.2 General Discussion and Recommendations

The optimization of the redundant arm was built with SR-inverse and weighted least-norm solution in this thesis. Forecasting the next joint motion was suggested to give better results of the weighted least-norm solution when avoiding joint limits. With the new conditions, the joint can move freely and smoothly within the joint range, and easily move back from the heavy weighted regions, and be stopped when the joint is going to reach its limits.

In the simulation of this research, there was only one arm considered for implementation on Matlab. To present a better performance, both arms together can be included in the simulation.

The results of ANOVA in test 2 show the experienced users and the non-experienced users have the same results of the time to complete the test because the cameras could not provide enough information of the door of the cabinet there, and they had difficulties in finding the pivot of the door of the cabinet. Hence, cameras with wider shooting angle are suggested, and the teleoperation system combines with force control is recommended.

In the survey, the score of collision force feedback was lower than the scores for the other questions. When the collision force feedback was applied in the teleoperation system, some users felt that the force feedback was not useful because the shapes of the grippers and objects were simplified to those of a sphere and a rectangle. Actual models of shapes of the end-effector and objects are recommended to give better feeling of force feedback.

Additionally, there were some difficulties when the users were trying to remotely control the robot to raise the basket because the speed at which the arms were raising were not always the same. It may have confused the users when they were thinking that the arms would move together perfectly. Actually, the users did not know what the exact direction and velocities were given to the program when they were controlling the robot, because when they were trying to give a specific motion, their hands were move towards other directions. For example, if they were trying to move forward, their hand might also turn a little bit left or right. This situation caused both arms to not move synchronously. For tasks that used a single arm, the users were able to correct the moving direction easily by changing their poses. For the tasks that required both arms, there should have been some mode to assist the users to complete the task, such as a puppet control mode.

7.3 Future Work

The results of this thesis show that the control algorithm can be applied for teleoperation. The control method is based on positions, angles, and velocities. Since there are torque sensors on each of Baxter's joints, the force control can also be applied for more possibilities of various teleoperation applications. If the control algorithm is based on force control, we can consider adding a PD controller to the control algorithm to improve precision.

Also, the experiments show the significance of dual arm manipulation. In the future, there should be more discussions and developments of dual arm coordinated teleoperation. Both arms are on the same torso, so the control should use the information from both arms to assist the user to control them. If there are some modes of dual arm manipulation for some tasks, it will be much easier to control the robot, such as one arm copies the pose of the other arm, or mirrored poses. Moreover, if the force control is applied to the dual arm control algorithm, the robot can have more possibilities and it will be easier to implement more complex tasks.

REFERENCES

[1] T.B. Sheridan,” Teleoperation, Telerobotics and Telepresence: A progress Report,” *Control Engineering Practice*, pp. 205-214, 1995.

[2] T. B. Sheridan, “Telerobotics*,” Automatica, vol. 25, no. 4, pp. 487–507, 1989.

[3] G. Niemeyer and J.-J. E. Slotine, “Stable adaptive teleoperation,” *IEEE J. Oceanic Eng.*, vol. 16, pp. 152-162, 1991.

[4] S. Goto, T. Naka, Y. Matsuda and N. Egashira, “Teleoperation System of Robot Arms Combined with Remote Control and Visual Servo Control,” *Proceedings of SICE Annual Conference 2010*, FB05.01, pp. 1975-1981, 2010.

[5] C.-L. Lai and P.-L. Hsu, “Design the remote control system with the time-delay estimator and the adaptive smith predictor,” *IEEE Trans. Ind. Informatics.*, vol. 57, no. 1, pp. 73–80, 2010.

[6] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, “Dual arm manipulation - a survey,” *Robotics and Autonomous Systems*, 2012.

[7] J. Kruger, G. Schreck and D. Surdilovic, “Dual arm robot for flexible and cooperative assembly,” *CIRP Annals – Manufacturing Technology*, vol.60, pp. 5-8, 2011.

[8] P. Hebert, N. Hudson, J. Ma, and J. Burdick, “Dual arm estimation for coordinated bimanual manipulation,” in *ICRA*, 2013.

[9] T. Wimbock and C. Ott, “Dual-arm manipulation,” in *Towards Service Robots for Everyday Environments*. Springer Berlin / Heidelberg, 2012.

[10] R.L.A. Shauri, and K. Nonami, “Assembly manipulation of small objects by dual-arm manipulator,” *Assembly Automation*, vol. 31, pp. 263-274, 2011.

[11] J. Smisek, M. Jancosek, and T. Pajdla, “3D with Kinect,” in *ICCV Workshop on Consumer Depth Cameras for Computer Vision*, 2011.

[12] P. Rakprayoon, M. Ruchanurucks and A. Coundoul, "Kinect-based Obstacle Detection for Manipulator," in *IEEE/SICE International Symposium on System Integration*, pp.68-73, 2011.

[13] F. Flacco, T. Kroger, A. De Luca, and O. Khatib, "A depth space approach to human-robot collision avoidance," in *Proc. IEEE Int. Conf. on Robotics and Automation*, St. Paul, MN, May 2012.

[14] A. Leeper, K. Hsiao, M. Ciocarlie, I. Sucan, and K. Salisbury, "Arm teleoperation in clutter using virtual constraints from real sensor data," in *RSS Workshop on Robots in Clutter: Preparing Robots for the Real World*, 2013.

[15] J. Park and O. Khatib, "A haptic teleoperation approach based on contact force control," *Int. J. Robot. Res.*, vol. 25, no. 5/6, pp. 575–591, May/Jun. 2006.

[16] M. A. Srinivasan and C. Basdogan, "Haptics in virtual environments: Taxonomy, research status, and challenges," *Comput. Graphics*, vol. 21, no. 4, pp. 393–404, 1997.

[17] B. Hannaford, "A design framework for teleoperators with kinesthetic feedback," in *IEEE Trans. Robotics Automat.*, vol. 5, no. 4, pp. 426-434, 1989.

[18] P. Nadrag, L. Temzi, et al., "Remote control of an assistive robot using force feedback," in *Proc. 15th IEEE International Conference of Advanced Robotics*, Tallinn, Estonia, Jun. 20-23, 2011.

[19] A. Leeper, S. Chan, K. Hsiao, M. Ciocarlie, and K. Salisbury, "Constraint-based haptic rendering of point data for teleoperated robot grasping," in Haptics Symposium (HAPTICS), 2012 IEEE, march 2012, pp. 377 –383.

[20] J. J. Craig, Introduction to Robotics: Mechanics and control, 3rd ed. NJ: Pearson Education, Inc., 1989.

[21] C. A. Klein and C. H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. 13, pp. 245–250, 1983.

[22] S. Lee, "Dual redundant arm configuration optimization with taskoriented dual arm manipulability," *IEEE Trans. Robot. Automat.*, vol. 5, pp. 78–97, Feb. 1989.

[23] F. Huber, K. Kondak, K. Krieger, D. Sommer, M. Schwarzbach, M. Laiacker, I. Kossyk, S. Parusel, S. Haddadin, and A. Albu-Schaffer, "First analysis and experiments in aerial manipulation using fully actuated redundant robot arm," in IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 2013.

[24] T. F. Chan and R. V. Dubey, “A Weighted Least-Norm Solution Based Scheme for Avoiding Joint Limits for Redundant Joint Manipulators,” *IEEE Trans. on Robotics and Automation*, vol. 11, no. 2, pp. 286–292, 1995.

[25] T. Yoshikawa, “Manipulability of robotic mechanisms,” Int. J. Robot. Res., vol. 4, no. 2, pp. 3–9, 1985.

[26] J. Luh, M. Walker, and R. Paul, “Resolved-acceleration control of mechanical manipulators,” IEEE Trans. Autom. Control, vol. AC-25, no. 3, pp. 468–474, Jun. 1980.

[27] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst*., Man, Cybern., vol. SMC-7, pp. 868-871, Dec. 1977.

[28] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Reading, MA, USA: Addison-Wesley, 1991

[29] R. Alqasemi, “Maximizing Manipulation Capabilities of Persons with Disabilities Using a Smart 9-Degree-of-Freedom Wheelchair-Mounted Robotic Arm System,” *PhD Thesis*, University of South Florida, Tampa, FL, USA, 2007.

[30] Hardware Specifications (n.d). Rethink Robotics. [Online]. Available: http://sdk.rethinkrobotics.com/wiki/Hardware_Specifications. Accessed Dec. 2, 2014.

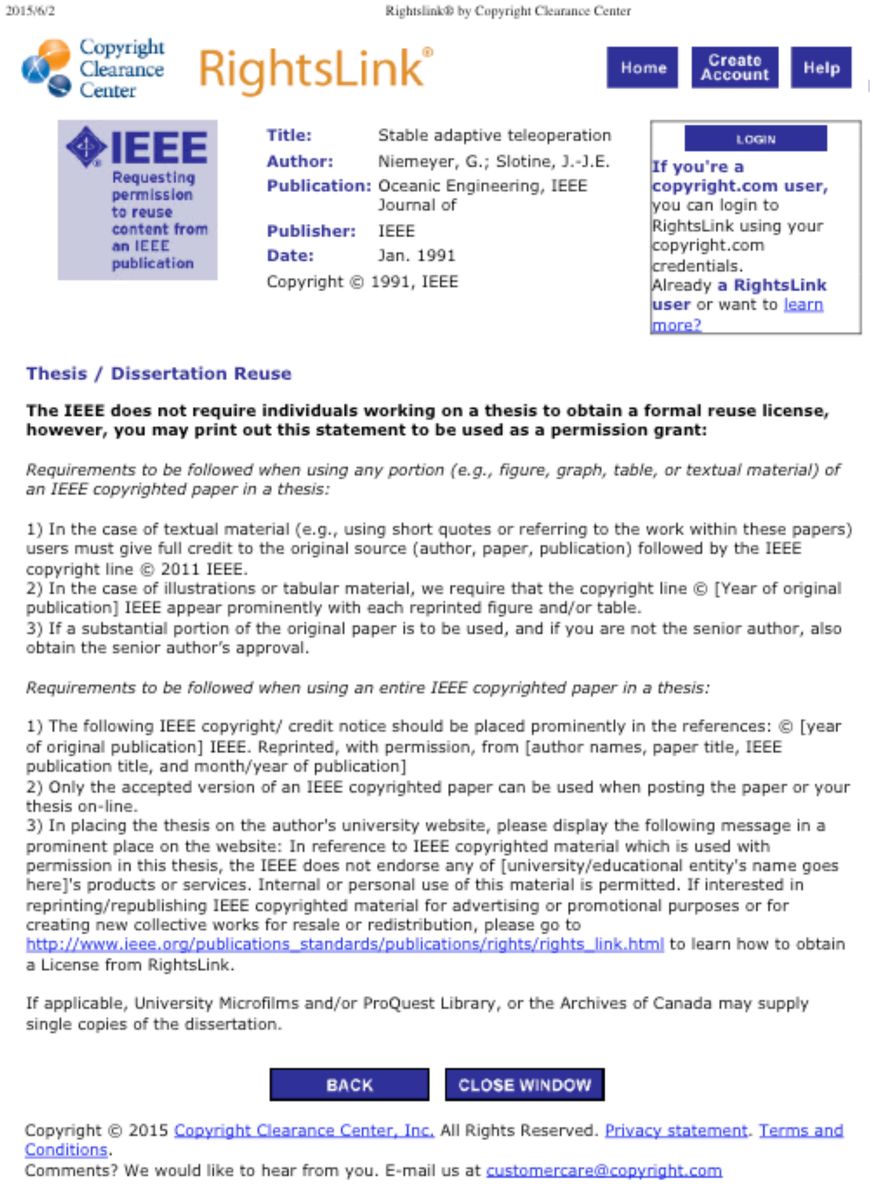
[31] K. Duncan, “Scene-Dependent Human Intention Recognition for an Assistive Robotic System, ” *PhD Thesis*, University of South Florida, Tampa, FL, USA, 2014.

[32] A. Leeper, K. Hsiao, M. Ciocarlie, I. Sucan, and K. Salisbury, “Methods for collision-free arm teleoperation in clutter using constraints from 3d sensor data,” in *IEEE Intl. Conf. on Humanoid Robots*, Atlanta, GA, 10/2013 2013.

[33] G. Box, "Some theorems on quadratic forms applied in the study of analysis of variance problems, I. Effect of inequality of variance in the one-way classification." *The annals of mathematical statistics*, 25, no. 2, pp. 290-302, 1954

APPENDIX A: COPYRIGHT PERMISSIONS

Below is permission for the use of materials in Chapter 2 and Chapter 3.


Copyright Clearance Center
RightsLink®
Home
Create Account
Help
IEEE
Requesting permission to reuse content from an IEEE publication
LOGIN
BACK
CLOSE WINDOW

2015/6/2 Rightslink® by Copyright Clearance Center

Title:	Stable adaptive teleoperation
Author:	Niemeyer, G.; Slotine, J.-J.E.
Publication:	Oceanic Engineering, IEEE Journal of
Publisher:	IEEE
Date:	Jan. 1991

Copyright © 1991, IEEE

If you're a copyright.com user, you can login to RightsLink using your copyright.com credentials. Already **a RightsLink user** or want to learn more?

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

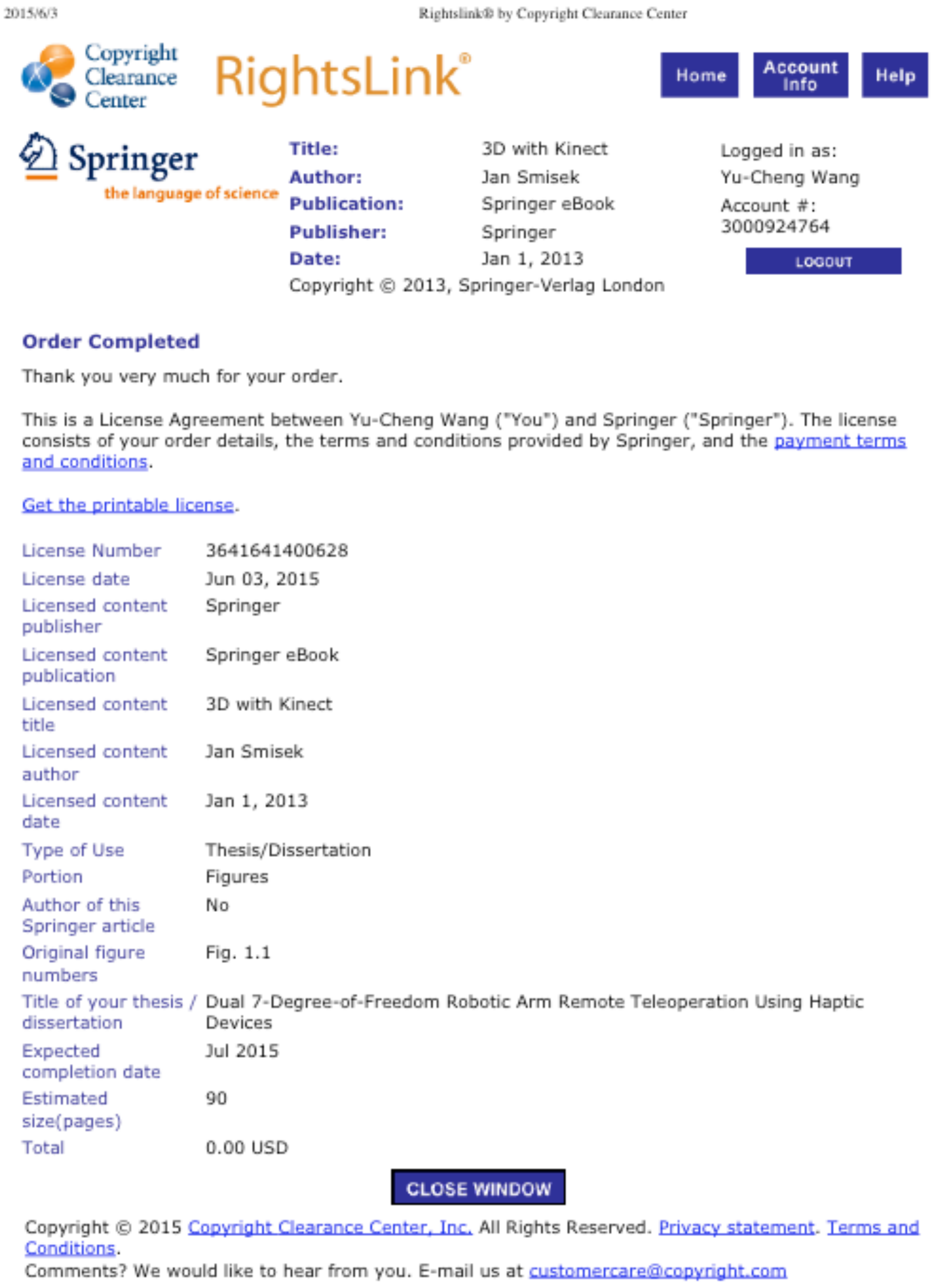
Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Comments? We would like to hear from you. E-mail us at customercare@copyright.com

Below is permission for the use of Figure 2.3.



2015/6/3 Rightslink® by Copyright Clearance Center

Title:	3D with Kinect
Author:	Jan Smisek
Publication:	Springer eBook
Publisher:	Springer
Date:	Jan 1, 2013

Copyright © 2013, Springer-Verlag London

Logged in as:
Yu-Cheng Wang
Account #:
3000924764

Order Completed

Thank you very much for your order.

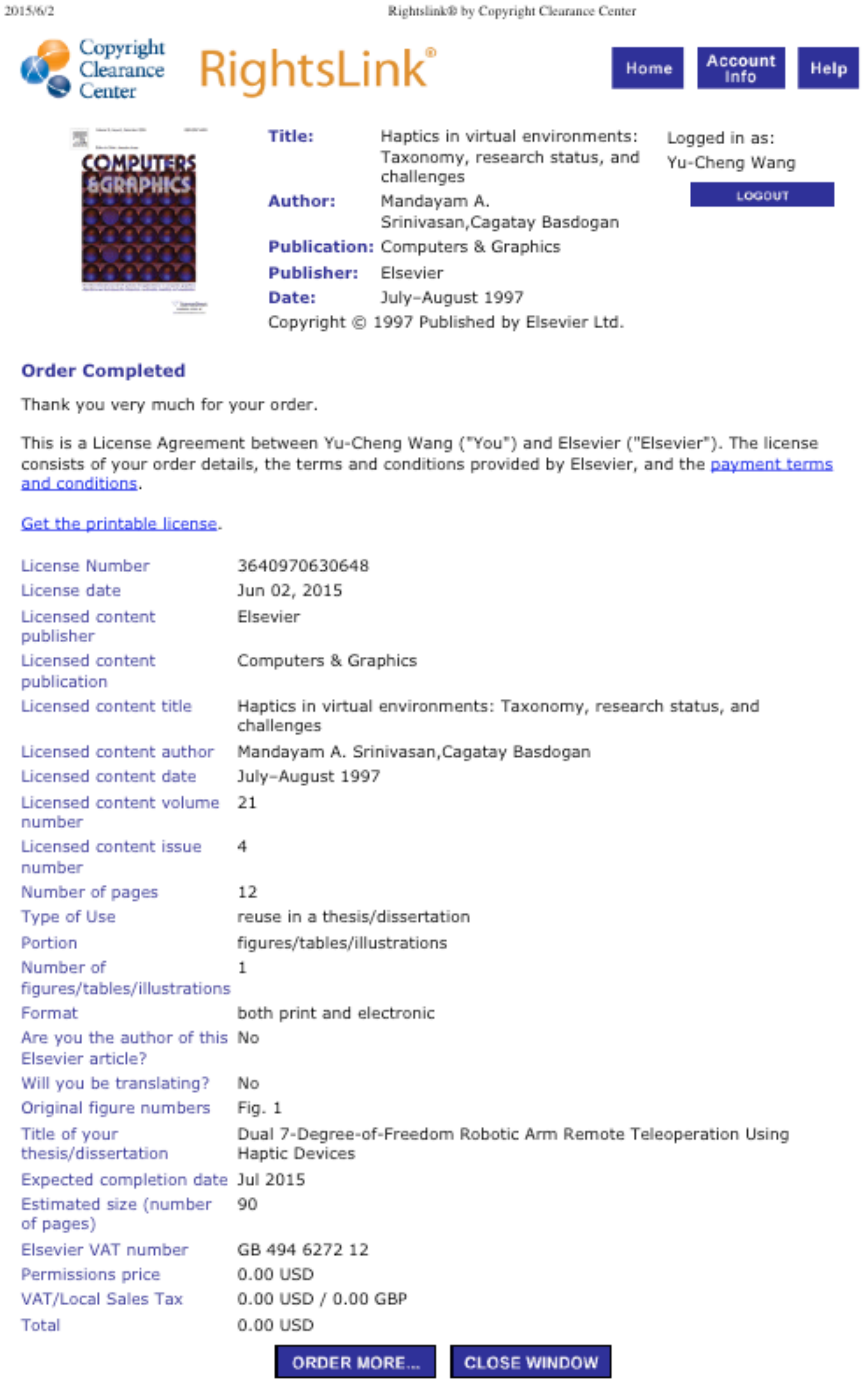
This is a License Agreement between Yu-Cheng Wang ("You") and Springer ("Springer"). The license consists of your order details, the terms and conditions provided by Springer, and the payment terms and conditions.

Get the printable license.

License Number	3641641400628
License date	Jun 03, 2015
Licensed content publisher	Springer
Licensed content publication	Springer eBook
Licensed content title	3D with Kinect
Licensed content author	Jan Smisek
Licensed content date	Jan 1, 2013
Type of Use	Thesis/Dissertation
Portion	Figures
Author of this Springer article	No
Original figure numbers	Fig. 1.1
Title of your thesis / dissertation	Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices
Expected completion date	Jul 2015
Estimated size(pages)	90
Total	0.00 USD

Comments? We would like to hear from you. E-mail us at customercare@copyright.com

Below is permission for the use of Figure 2.7



2015/6/2 Rightslink® by Copyright Clearance Center

Title:	Haptics in virtual environments: Taxonomy, research status, and challenges
Author:	Mandayam A. Srinivasan,Cagatay Basdogan
Publication:	Computers & Graphics
Publisher:	Elsevier
Date:	July–August 1997

Copyright © 1997 Published by Elsevier Ltd.

Logged in as:
Yu-Cheng Wang

Order Completed

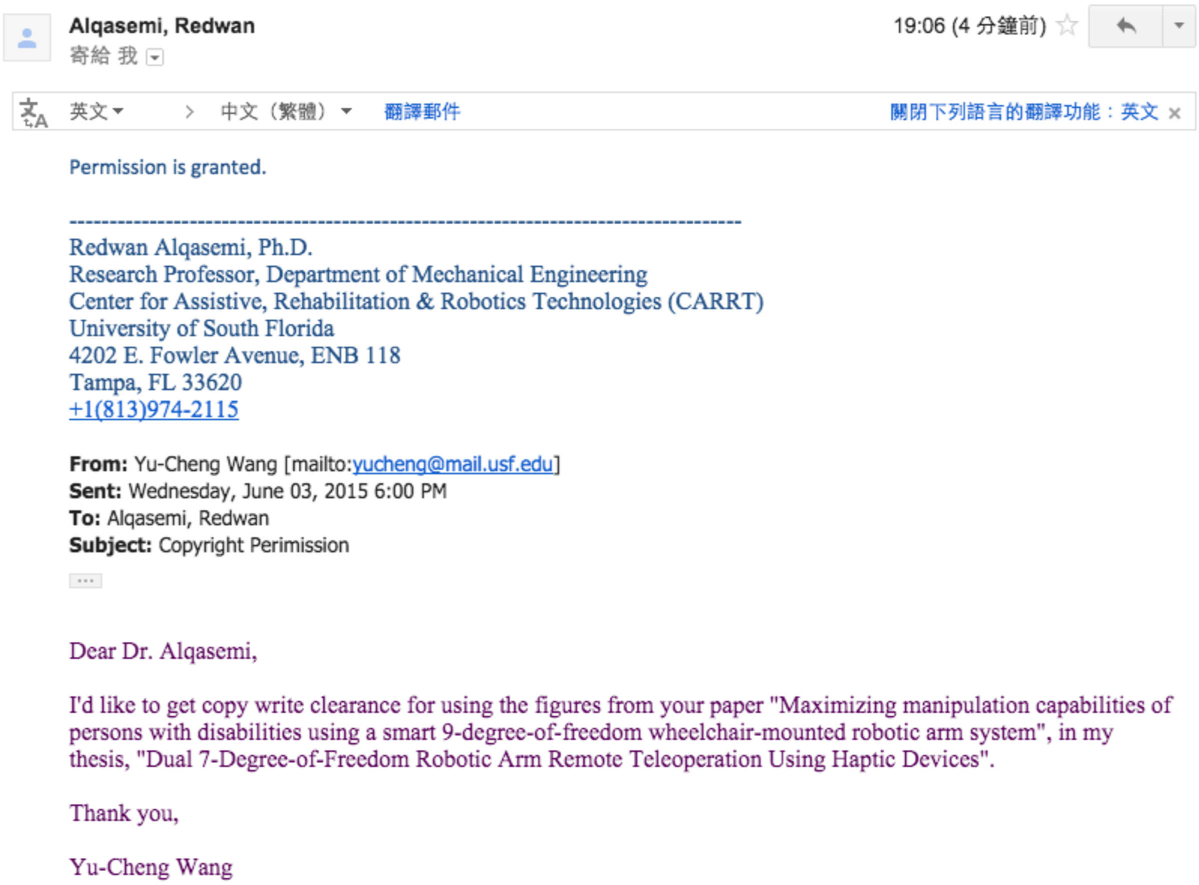
Thank you very much for your order.

This is a License Agreement between Yu-Cheng Wang ("You") and Elsevier ("Elsevier"). The license consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

Get the printable license.

License Number	3640970630648
License date	Jun 02, 2015
Licensed content publisher	Elsevier
Licensed content publication	Computers & Graphics
Licensed content title	Haptics in virtual environments: Taxonomy, research status, and challenges
Licensed content author	Mandayam A. Srinivasan,Cagatay Basdogan
Licensed content date	July–August 1997
Licensed content volume number	21
Licensed content issue number	4
Number of pages	12
Type of Use	reuse in a thesis/dissertation
Portion	figures/tables/illustrations
Number of figures/tables/illustrations	1
Format	both print and electronic
Are you the author of this Elsevier article?	No
Will you be translating?	No
Original figure numbers	Fig. 1
Title of your thesis/dissertation	Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices
Expected completion date	Jul 2015
Estimated size (number of pages)	90
Elsevier VAT number	GB 494 6272 12
Permissions price	0.00 USD
VAT/Local Sales Tax	0.00 USD / 0.00 GBP
Total	0.00 USD

Below is permission for the use of materials in Chapter 3



Alqasemi, Redwan

Permission is granted.

Redwan Alqasemi, Ph.D.
Research Professor, Department of Mechanical Engineering
Center for Assistive, Rehabilitation & Robotics Technologies (CARRT)
University of South Florida
4202 E. Fowler Avenue, ENB 118
Tampa, FL 33620
+1(813)974-2115

From: Yu-Cheng Wang [mailto:yucheng@mail.usf.edu]
Sent: Wednesday, June 03, 2015 6:00 PM
To: Alqasemi, Redwan
Subject: Copyright Perimission

Dear Dr. Alqasemi,

I'd like to get copy write clearance for using the figures from your paper "Maximizing manipulation capabilities of persons with disabilities using a smart 9-degree-of-freedom wheelchair-mounted robotic arm system", in my thesis, "Dual 7-Degree-of-Freedom Robotic Arm Remote Teleoperation Using Haptic Devices".

Thank you,

Yu-Cheng Wang

APPENDIX B: A FORM OF SURVEY OF THIS THESIS

Dual 7 Degree-of-Freedom Robotic Arms Control with Haptic Device over TCP/IP
Experiment Survey

1. Personal Information

Name		Age		Gender		Date	
Email				Weight		Birth Day	

2.Experment Data

Task I. Grip Bottle

	1	2	3	Average
W/o Obstacle				
Collision				
W/ Obstacle				
Collision				

Task II. Cabinet

	1	2	3	Average
Time				
Collision				

Task III. Basket and Bookshelf

	1	2	3	Average	Failure
Time					
Collision					

3. Experience (1 to 10; 1: worst, 10: best)

Score	1	2	3	4	5	6	7	8	9	10
Easy to use	□	□	□	□	□	□	□	□	□	□
Easy to learn	□	□	□	□	□	□	□	□	□	□
Interface experience	□	□	□	□	□	□	□	□	□	□
Frustration level	□	□	□	□	□	□	□	□	□	□
Usefulness of force feedback	□	□	□	□	□	□	□	□	□	□
Usefulness of IR sensor	□	□	□	□	□	□	□	□	□	□
Usefulness of arm cameras	□	□	□	□	□	□	□	□	□	□
Usefulness of head camera	□	□	□	□	□	□	□	□	□	□
Difficulty of task I (no obstacle)	□	□	□	□	□	□	□	□	□	□
Difficulty of task I (obstacle)	□	□	□	□	□	□	□	□	□	□
Difficulty of task II	□	□	□	□	□	□	□	□	□	□
Difficulty of task III	□	□	□	□	□	□	□	□	□	□

4. Please leave your comment or suggestion for this experiment