6-13-2011

Robot-Assisted Manipulation: Utilizing Communication of User Intent Through Tactile Sensing and Spatial Hand Tracking

Matt Ambauen
Santa Clara University

Santa Clara University

Department of Mechanical Engineering

Date: June 13th, 2011

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISIOR BY

Matt Ambauen

ENTITLED

Robot-Assisted Manipulation

Utilizing Communication of User Intent Through Tactile Sensing and Spatial Hand Tracking

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

Christopher A. Kitts
Thesis Advisor

Timothy Hight
Chairman of Department

Robot-Assisted Manipulation

Utilizing Communication of User Intent Through Tactile Sensing and Spatial Hand Tracking

By

Matt Ambauen

GRADUATE MASTER THESIS

Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Mechanical Engineering
in the School of Engineering at
Santa Clara University, 2011

Santa Clara, California

Abstract

Today, we have integrated robots into our society, as seen by the increasing number of robots that emerge from factories and labs to assist humans in their daily activities. Consequently, more frequent interaction between robots and humans has augmented the importance of developing technology that ensures effective, efficient, and safe robot-human collaboration. This research focuses on the robot-assisted manipulation of objects, and successfully demonstrated a robot could manipulate a common load with a human user in an assistive role. Industrial assembly, where a human operator may desire to manipulate a large or heavy object, exemplifies this cooperation. A robotic collaborator could help the user manage the load and maneuver it to the desired location through a precise and safe series of actions. To do this efficiently, the collaborators must communicate objectives and intent via non-explicit cues, such as if they push or pull the object in a desired direction.

In this case, the research team utilizes a novel pair of force sensing gloves, a SCARA configuration industrial robotic manipulator, and an external motion capture system to track positions of the robot and the user and to achieve non-explicit communication. A specified protocol of inputs based on the user's grasping force, which feed into an Inverse Jacobian velocity controller, defines the motion of the robot. This frees the user to drive the object through the workspace without any need for a predefined path or movement. The vision system allows the controller to track the position of the robot, as well as the location of the user, and can therefore ensure that the robot and/or object never collide with the operator. To ensure the accuracy of the controller, the team tested the communication of intent and the resulting protocol in each of the possible motions and will show it resulted in the desired motion.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Robotic systems have been of interest to both scientists and researchers for decades, but were first theorized in the literary world. Science fiction writers have written about the various and sometimes exceptional uses of robots in many, many of their works. Today they are utilized to improve efficiency, repeatability, and production speed in many industries. They are capable of doing the tedious and monotonous jobs in industry, often offering higher precision and better repeatability than what human workers could regularly reproduce. They can operate in hazardous environments and under condition impossible for humans to endure, such as radioactive environments or in space. They offer inhuman speed and accuracy in manipulation tasks, which has allowed the growth of the electronics and automotive industry. More modern uses of robots include robotic surgery, bomb disposal, search and rescue, surveillance, and prosthesis. Everyday, robots affect human life in a positive way, by improving or preserving health and wellness of their operators or patients. They have become integral to our modern world.

From assembling cars and computer chips to operating on cancer patients, humans have come to increasingly rely on robots to support and execute daily processes. Because of this, we need to overcome the technical problems that define a robot's ability to manipulate objects and to interact with their human counterparts. Robots traditionally receive preplanned trajectories and path functions that define their motion through the workspace. Any interaction with its environment is all predetermined and preprogrammed, leaving little room for natural interaction with a human user. In comparison, humans who try to manipulate objects in groups give and receive a myriad

of nonverbal cues and motions in order to accurately move the object. Ultimately, our vision for robot-human interaction is to reflect effective human-human interactions, attempting to match the communication and collaborative action present between two human operators. Thus, in the interest of safety and accurate manipulation and mobility, robots that leave their work-cells and enter the market must functionally accept real-world explicit and non-explicit commands. Current robotic motion is typically a series of planned trajectories, but consumers now demand robots that can handle more organic and spontaneous actions. Several other researchers have recently done work in this area and characterize assistive manipulation in several ways. Robot-assisted manipulation has a paradigm similar to the theories surrounding human-robot interaction, for they both assert that the robot partner perceives the intent and goals of the human counterpart in order to achieve common goals. With this technology an operator on an assembly line can take hold of a car chassis and successfully manipulate it onto the wheelbase in an organic fashion, while the system simultaneously ensures that both the object and the robot avoid contact with the operator. In an elder person's home, a mobile assistance robot can help the human operator remove dinner from the oven. While alone, an amputee who received a robotic prosthetic hand can lift a heavy box down from a shelf. Society has yet to realize the potential of these systems because of technological limitations, namely the need for high resolution visual sensing of the position of the robot, the user and the object being manipulated. If mechanical engineers develop these functions, they would make robots more useful and safer for a greater demographic of human operators.

The goal of this project was to develop a system that could organically and fluidly assist an operator in manipulating an object through the workspace. In order to do so, the team needed to develop a novel user interface system to sense the intent of the user and

communicate that to the robot in an efficient and expedient manner to ensure both the accuracy and stability of the system. The major challenge presented was similar to that stated above, the need for robust external sensing of the position and orientation of the robot, the object being manipulated, and the human operator. Also the need to sense user motion and forces exerted on the object became a challenge as there was not a readily available human interface system intended to gather and communicate that kind of input. In order to overcome these goals, the team designed their own human interface device to gather and communicate grasping forces and orientations as the user interacted with the object. They wrote the sensing protocol to interpret the user intent from the sensor data and dictate the resulting robotic motion. User intent, in the context of this research, is the interpretation of the user's grasping forces and hand positions to determine the desired object motion. The robot is not aware of the overall goal of the user manipulation but it is responsible for interpreting and predicting the desired motion from the sensor network. To ensure accuracy and safety, the team utilized an external vision system to track the position and orientation of the robot, the object being manipulated and the human user the robot is assisting.

1.2 Review of Literature

Engineers have recently published many articles that regard collaborative manipulation; the varied methodology and hardware used by each author differs as widely as their research projects. They have given the issues of human interaction and of communicating human intent much attention, and have suggested a multitude of potential solutions to these problems. The mouse and the keyboard remain classic human interface devices, for they allow a human user to input data and give commands to a computer system. However, their success first required several iterations of similar devices in order

to develop the modern version of these refined and useful tools. In robotics, the joystick is the standard human interface device for mobile robots, and for two main reasons: it can give varying degrees of commands and track user intent across a wide range of commands; and it also allows for intuitive steering and turning of tracked and wheeled robots. Manipulators, on the other hand are usually bench top hardware, which forces users to control them through a teaching pendant or joystick. However, users find these methods cumbersome and unintuitive when it comes to manipulation and human interaction. In response to this problem, researchers have developed glove sensors, passive compliance end-effectors, and exoskeletons (Tarchanidis)(Caldwell)(De Carli). They often equip these devises with haptic feedback, and they design them to track the position and orientation of a user's hand or arm as well as the interaction forces exerted by a user or felt by the robot.

Figure 1: User Interface Devices from (Tarchanidise), (Caldwell) and (DeCarlo)

In order to simplify the human-robot interaction problem, some researchers have made assumptions about the nature of communication between human operators and robots. In Kosuge's seminal 1993 paper, from his research at Nagoya University, he states that the researchers assume that the robot and the human only communicate through the object (Kosuge). Kosuge also assumes that the human manipulates the object about a point fixed in the object. In the scenario laid out in Kosuge's paper the human operator exerts a force on the object in a specific direction to command the object's

motion. Kosuge proposes a passive dynamics approach to the human-robot collaborative manipulation problem rather than an approach that senses the intention of the human operator. Even if Kosuge modified his work for a novel take on the human-robot manipulation problem, he clearly defines the fundamental concept of interaction via the object manipulated.

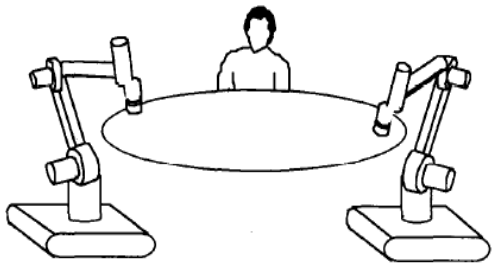


Figure 2: Possible Configuration for Collaborative Mulch-Robot Manipulation Scenario (Kosuge)

Human robot interaction via the object being manipulated implied some type of communication between the user and the robot, specifically the intent of the user. In order to accept commands via user intent, researchers tracked non-explicit cues in human-robot interactions. In research done by De Carli in 2009, force and torque sensors at the wrist of a robot were used to sense these cues. The hardware for this research was a 6 DOF robotic manipulator with a handle attached to the end effector. The base of this handle was equipped with force and torque sensors that registered the forces being exerted on the handle by the user. This allowed the robotic controller to interpret the user communication that specified a change in direction (Carli). A uniform impedance controller, located in the end-effector space of the robot, set out the plan of the path for the robotic motion. The user input to the system dictated the path re-computation. In contrast, current research by the Santa Clara University (SCU) robotics team has the user

intent directly drive the directionality of the robotic motion, which thus ultimately negates the step of path planning. This means that the robot has a relatively instantaneous and very accurate response, to the resolution of the sensors.

Similar to the human-human manipulation task, the human-robot manipulation task also has a leader and a follower. Reed experimented with these interactions in an attempt to better understand how the individual reacts on its own and how it interacts with a partner. The experiment presented in his 2008 paper eliminated the sense of sight, the absence of which required the subjects to manipulate a disc with and without a partner. Reed measured the position and velocity of the disc throughout the task. The experiment aimed to establish how users interact with the disc and each other in follower and leader roles (Reed). In contrast, the SCU RSL experiment studied the human-robot collaboration problem with the robotic arm tasked in a pure follower role. Because the manipulator is tasked as a pure follower, our research is best classified as robot-assisted manipulation.

Researchers find it difficult to determine the role of the agents, either as followers or as leaders, in situations where those agents interact with each other through physical contact via the object manipulated. Evrard posited that "[although] a robot can perform simple collaborative tasks, it certainly is unable to negotiate them with their human partners."(Evrard) On the other hand, these two roles rely on certain haptic cues between agents. The researcher can use this non-explicit communication to weight a function between the leader and follower roles. If programmers define this haptic language, or protocol of non-explicit communication, they could then direct the manipulation task and the motion of the arm. For this reason, the SCU team developed a haptic communication protocol based on a force sensing haptic interface device and the spatial information of

the object and operator in the workspace.

Any application that requires, or could benefit from, robots and humans to come into direct contact has application from the research done in this project, especially in industrial manipulation, household-assitive robots, robotic prosthesis, and robotic assembly. The research is driven by the need to have humans and robots collaboratively manipulate an object without communicating in the traditional ways. Instead, a new, novel means of communication was developed to interpret and communicate the intent of the user to the robotic controller so it could execute the desired task.

Large-scale robotic manipulators have an inherent safety risk to the human operators in their vicinity, which is why they have been traditionally housed in work cells and isolated from the operators. But by considering novel ways to track the user and ensure that the robot or the object being manipulated never comes into contact with the user, human-interaction with the robot in a more direct and organic fashion becomes possible. In the case of this research, the robot and the user communicate only through the object by means of the novel user interface device. The implication of physical interaction between the user and the robot is safe because of the concerns built into the controller.

Applications of this interaction research are scalable from large-scale robotic applications in industrial assembly, where the robots are manipulating extremely heavy parts through the workspace, down to personal robotics where a robot is assisting a user who may have a weakened constitution and is unable to manipulate everyday objects. Another non-traditional manipulation scenario is the human user working with an intelligent robotic prosthesis to manipulate an object with both a healthy hand and the

robotic hand. That manipulator (e.g prosthesis) is already taking nonverbal cues from the user, but utilizing the research outlined in this paper, and the nonverbal cues typically utilized in two-handedmanipulation are restored. Ultimately, the goal of the project is to improve the efficiency and accuracy of the robot-assisted manipulation task, regardless of the application.

Figure 3: SCARA Manipulator

1.3 Objectives and Goals

This project explored the combination of haptic feedback and relative spatial information between a user and an object as a means of directing a robot to manipulate the object. To do this, a new interactive protocol that exploits both of these types of information was developed in order to specify object motion in Cartesian space. This protocol was integrated with a Cartesian controlled, SCARA manipulator, and the resulting testbed was used to experimentally verify this new approach for robot-assisted manipulation. The larger objective of this study was to investigate the efficacy of a robotic system that could assist a human operator in manipulating an object through the workspace by taking natural and organic input from the user through the object being manipulated. As previously stated, the SCU team designed this system using an industrial style SCARA configuration manipulator, however the team maintained the scalability of the design to ensure it could be used with a different robotic configuration, with higher degrees of freedom.

1.4 User Scenario

While the details of specifying motion will be described in detail later in this

thesis, it is instructive to quickly review the overall strategy here. As seen in Figure 4, a user places both hands on the object to be manipulated with the robot fully supporting the object at some convenient grasp location. The location of the user's hands and and the nature of the user's grip are both sensed, in this case by an overhead vision system and by sensors embedded in gloves worn by the operator. Grip information is used to specify the desired motion of the object, whether that it be pushed away from the operator or pulled towards the operator. It can also used with the position information to specify that the object should take on a desired rotation path, i.e. about one of the user's hands. In this manner, the operator can very naturally direct the object's robot actuated motion by applying very light forces and torques on the object (Kitts, 5/11/2011).

Figure 4: Proposed User Scenario for Human-Robot Collaborative Manipulation

1.5 Reader's Guide

Chapter 1 – Introduction serves as an opening chapter and highlights the motivation and overall goals of the project. Chapter 2 – Robotic System Development looks at the requirements of the robotic system, the process of hardware integration, and the utilization of the external vision system to examine the development of the robotic

platform. Chapter 3 – Tactile Glove Development reviews the development of the human interface device, in this case the force sensing tactile glove. This chapter covers the steps taken to design and construct the device. Chapter 4 – Control System Development discusses the development of the various control strategies generated in this project, and how those strategies led to the final control system. Chapter 5 – System Integration and Experimentation discusses the integration of the robotic system with the human interface device and the subsequent tests done with the system as a whole. Chapter 6 – Summary and Conclusion summarizes the project and investigates potential future work.

2 Control System Development

2.1 Approach and Motivation

In order to gain accurate control of the robot and be able to utilize the sensed information correctly the team developed a controller using the iterative development approach. Because the controller and the mechatronic hardware were developed and integrated from scratch, the iterative development approach was deemed best and thereby caused the team to first develop a joint space controller in order to drive each degree of freedom of the robot. This was done through the integration of the motor controllers and Matlab, and writing the controllers to drive each of the motors to the desired locations at the desired rates. PID controllers and a joystick interface were used to design and develop a joint-space controller, however this was a brief exercise and not directly related to the research.

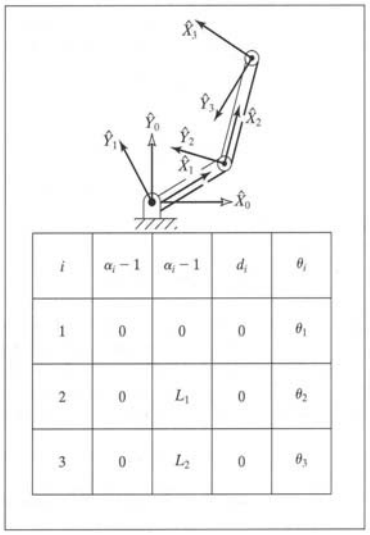
2.2 Kinematic Definitions

Once the team had established joint space control of the robot, the next step was to gain direct control of the position of the robot end-effector. This is done through a Cartesian space controller and requires knowledge of the kinematics of the robot. These kinematics are specific to the mechanism design and must be calculated at each cycle of the controller.

2.2.1 Position Kinematics

Since the team is only using the robot in the planer sense, the model is equivalent to a simple model from Craig (see Figure 5)(Craig, 109-112). The manipulator kinematics are defined from the link parameters of the robot and are specific to its current location. In order to define the link parameters the team selected the Denavit-Hartenberg

Parameters (DH Parameters) for the specific manipulator. These are derived by inference from the manipulator mechanism and are non-unique.



i	$\alpha_i - 1$	$\alpha_i - 1$	d_i	θ_i
1	0	0	0	θ_1
2	0	L_1	0	θ_2
3	0	L_2	0	θ_3

Figure 5: Definition of DH Parameters (Craig)

These parameters are then utilized to compute the individual transformations for each of the links. The details of the derivation are included in Appendix A, and result in the single transformation that relates the end effector frame (frame 3) to the base frame (frame 0).

$$ {}^{0}_{3}T = {}^{0}_{1}T \; {}^{1}_{2}T \; {}^{2}_{3}T \qquad [1] $$

$$ {}^{B}_{W}T = {}^{0}_{3}T = \begin{bmatrix} c_{123} & -s_{123} & 0 & l_1 c_1 + l_2 c_{12} \\ s_{123} & c_{123} & 0 & l_1 s_1 + l_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad [2] $$

These equations have defined the manipulator kinematics, that is, the orientation of the manipulator given specific values for the link variables.

In order to find the inverse kinematics, the team made certain that the necessary calculations and transformations were done to ensure that the user input coordinates were

a specification of the end effector with respect to the base frame. The goal of the inverse kinematic function is to specify the angle measures necessary to place the end-effector at a desired location. Therefore, the input required three pieces of information, the coordinates of the end-effector and the orientation of that frame, defined by: *x*, *y*, and ϕ respectively. This specification yields several equations, detailed in Craig and summarized in Appendix A, which leads to a set of three non-linear equations that must be solved for θ_1, θ_2, and θ_3. The solutions for the inverse kinematic equations are listed as

$$\theta_2 = Atan2(s_2, c_2) \quad [3]$$

$$\theta_1 = Atan2(y, x) - Atan2(k_2, k_1) \quad [4]$$

$$\theta_1 + \theta_2 + \theta_3 = Atan2(s_\phi, c_\phi) = \phi \quad [5]$$

And because θ_1 & θ_2 have already been found, θ_3 can be found easily.

2.2.2 Velocity Kinematics

In order to develop the necessary controller, the team had to determine the correct Jacobian matrix for the robot geometry. This is done by examining the mechanism of the robot, attaching reference frames and calculating the velocity of the end-effector in different reference frames. The reference frames are attached to the robot in the same configuration as was used in the definition of manipulator kinematics. The details of the calculations are laid out in Craig and summarized in Appendix A, however the results for the Jacobian are listed below and the inverse Jacobian is include in appendix A as size prevents it from being included here.

$$^0J = \begin{bmatrix} -L_1 s_1 - L_2 s_{12} - L_3 s_{123} & -L_2 s_{12} - L_3 s_{123} & -L_3 s_{123} \\ L_1 c_1 + L_2 c_{12} + L_3 c_{123} & L_2 c_{12} + L_3 c_{123} & L_3 c_{123} \\ 1 & 1 & 1 \end{bmatrix} \quad [6]$$

2.3 Cartesian Space Controller

2.3.1 End Effector Controller

Section 2.1.2 outlined the calculations necessary to implement an End-Point or Cartesian Space controller. These calculations are included in the "invKin" block in the control block diagram shown below and that code is included in Appendix B.

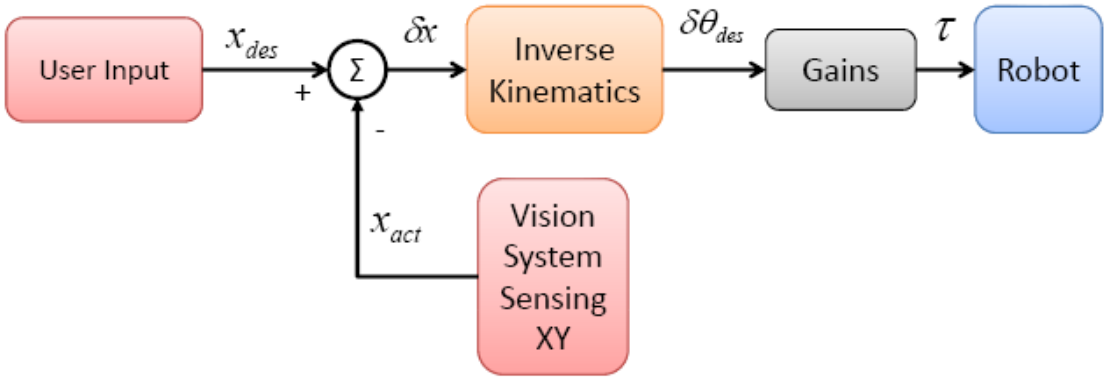


Figure 6: Control Block Diagram for Cartesian Space Controller

This controller utilizes the vision system to sense in the x and y positions of the joints of the robot. The controller then compares the desired x-y positions of the end-effector of the robot to calculate the error between the desired and actual x-y positions. The x-y error, is then plugged into the inverse kinematics function to compute the desired theta error. Gains are applied and the resulting joint torques are passed to the robot. The Cartesian controller was designed as a proportional controller, with correct gains to drive the robot to the desired positions, based on the feedback from the sensor, quickly and efficiently, but without excessive oscillation or overshoot as this could be a hazard to the user. The gains were tuned experimentally based on these requirements. The controller passes the desired drive forces to the robot block as percents of one hundred. The robot communication block remains the same in all of the controllers as it primarily contains

the mechatronic communication protocol with the robots motor controllers.

2.3.2 Obstacle Avoidance Controller

As a safety feature, the team modified its Cartesian space controller to ensure that the robot was incapable of colliding with the user. Because the sensing system for the robotic platform was the external vision system and was not only tracking the rigid bodies attached to the robot but also rigid bodies attached to the user's hands, the location of the user was available to the controller. In order to execute this controller, the team wrote a higher level function, indicated here as the Protocol block. This controller computed the distance between the user's hand and the end effector, and if the user got too close to the end effector, it would move away from the user.

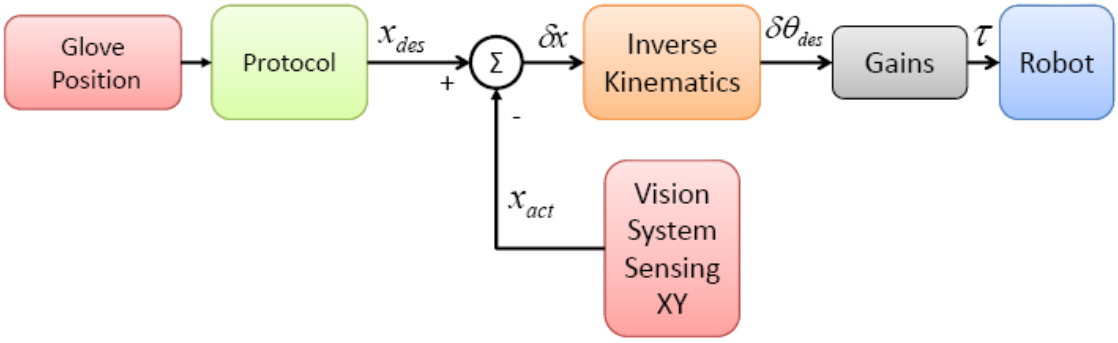


Figure 7: Control Block Diagram for Obstacle Avoidance Controller

This controller is very similar to the previous Cartesian Controller except that it has a front end function to compute the locations of the user's hands and to determine the desired end effector location. The desired end effector location was calculated to place the end effector outside the threshold level. The controller was designed in the same way as the controller in section 2.3.2, and was dynamic and would adjust to changes in user position very quickly. This controller was utilized loosely in the final controller, however the threshold at which the robot would avoid the operator was rarely triggered so it had

little effect on the final results of this research.

2.4 Jacobian Based Control Architecture

For the final experimentation and validation done on the project, the team intended to test the robot-assisted manipulation of an object based on user input via the force-sensing gloves. However in this context the user input is interpreted and ultimately specified to the controller as desired object velocities. Therefore it became necessary to design a controller that was based on velocity rather than position.

2.4.1 Implementation of Jacobian based Controller

The controller in Figure 8 shows the inverse Jacobian scheme. As the user grasps the object and exerts various forces on it, the force sensors in the gloves register that exertion. The Glove Protocol block interprets those force sensor inputs to determine the user's intention, in this case the desired motion of the object, specifically the velocity vector of the object in its object frame. This block requires knowledge of the current orientation of the robot and the object as well as the position of the gloves and the end-effector of the robot. In order to communicate accurately with the robot, that desired velocity needs to be converted from the object frame to the base frame. This is accomplished through the Frame Transformation block and the details are included in Appendix B. The desired velocity vector, as converted to the base frame, is then compared to the actual velocity vector, yielding the velocity error vector. This error vector is then matrix multiplied by the inverse Jacobian to determine the desired joint rates, which are communicated directly to the robot as percents of one hundred.

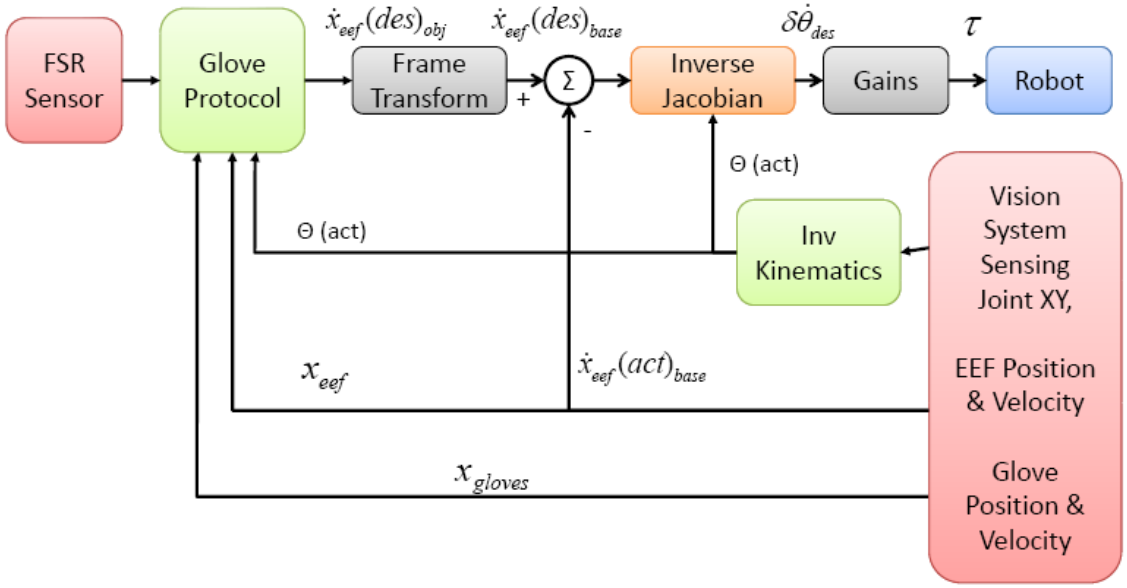


Figure 8: Jacobian Control Block Diagram

2.5 Tactile Glove Protocol

It was necessary to define a protocol to decode the meaning of the sensed grasping data and define the user intent based on that information. The team wrote a Matlab function to take the raw input from the tactile sensing gloves and convert it into a desired command to the robot in the object frame. The definition of the tactile glove protocol is laid out in the following section, while the source code is included in Appendix B.

2.5.1 Use of FSR Sensors to Register Grasping

The sensitivity of the resistor was rated at 1 MΩ when no pressure was exerted, with sensitivity able to sense forces between 100g – 10kg. The analog to digital conversion was set up to pass raw numbers that correlated to that change in resistance into Matlab and this was then converted to a percent-of-one-hundred. The team was able to register the forces in the user's grasp by taking input from the sensors on the thumb, index and middle fingers on both hands. The team defined a threshold to eliminate noise,

and so forces above this threshold would register as force exerted.

There were several different grasps defined within the tactile glove protocol. They were assessed on both hands and defined independently for each hand based on the force data from each glove. The grasps are defined in Table 1, with their required conditions to be met.

Sensor **Grasp**	Thumb	Index	Middle
Push	H	L	L
Pull	L	H	H
Grip	H	H	L
Squeeze	H	H	H

Table 1: Grasp definition, protocol is same for both hands

These grasps are sent to the next stage of the protocol where they are utilized to define the desired motion for the robot.

2.5.2 Use of Vision System to Register Hand Position

The external vision system was a valuable subsystem for this project. It allowed for accurate tracking and definition of the joint angles and allowed direct sensing of the end effector position and orientation. The vision system also ensured that the robot did not collide with the operator by tracking the location of the user. Tracking the user was also used in the glove protocol to determine the orientation of the user with respect to the object being manipulated. Just as the robot can approach a position from two possible orientations, so can the user in their orientation to the object. These possible orientations are illustrated in Figure 19.

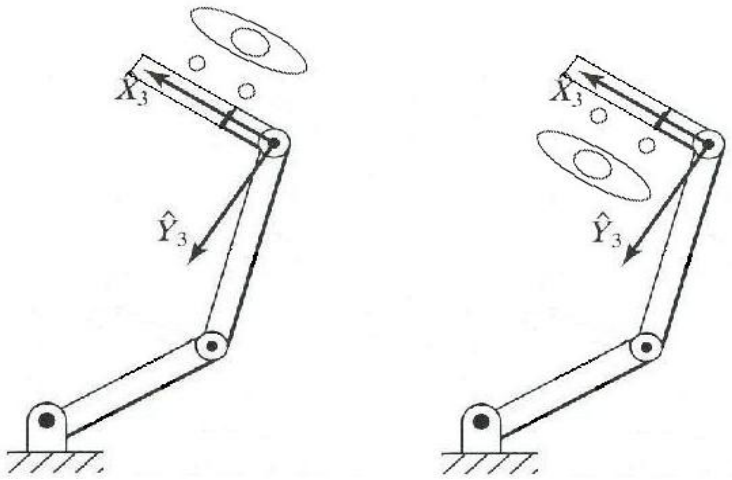


Figure 9: Possible Orientations of the User and Robot Defined as Left Handed and Right Handed

The possible orientations are defined as left and right handed, based on which of the user's hands are closer to the robot end-effector. It is necessary to define the difference in user orientation with respect to the object because of the possible ambiguity in user motion. For example, a push in the left hand orientation is not the same as a push in the right hand orientation, in fact it is a pull. Without a clear resolution of this ambiguity, it is necessary to divide the protocol into two different cases, based on the orientation of the user with respect to the object.

2.5.3 Classification of User Intention Based on Sensor Input

Once the user orientation and grasp conditions have been established it is possible to define the user intention. User intention is defined as the motion of the object that the user intends to result from their physical interaction with the object. The possible classifications of user intent are motion in the positive or negative X or Y direction, in the object frame, or rotation about the user's closer hand in either the clockwise or counter clockwise direction. The following table shows the possible grasp combinations that can result in each of these intended motions. Motions are defined assuming a Right-Handed

user orientation.

Grip/ Motion	RH				LH			
	Push	Pull	Grip	Squeeze	Push	Pull	Grip	Squeeze
X	0	0	1	0	0	0	0	1
-X	0	0	0	1	0	0	1	0
Y	0	1	0	0	0	1	0	0
-Y	1	0	0	0	1	0	0	0
CW	0	1	0	0	1	0	0	0
CCW	1	0	0	0	0	1	0	0

Table 2: Definition of Intended Object Frame Motion Based on Sensed Grasp Forces

Once the user's intention is defined it must be converted to a manipulator action and then communicated to the controller.

Based on user intention, it was possible for the team to define a command vector for each of the possible intentions. Table 2 was completed for both orientations of the user, and while the grip combinations were the same, the resulting motions were different for some of the combinations based on the location of the user with respect to the object. There were several iterations of the vectors defined for each grasp to test for desired speed and to make the motions feel organic and efficient. This desired object frame motion, classified by the glove protocol block as a velocity vector was then passed on to the rest of the Jacobian based controller.

3 System Development

3.1 Robotic System Overview

3.1.1 System Level Requirements

In order to develop the robotic manipulation testbed, it is necessary to first develop a robotic platform. That platform must be robust enough to handle the loads placed upon it by interacting with the user, i.e. the pushing and pulling forces applied during communication of intent. It also must have a large enough workspace to sufficiently test manipulation tasks. Most importantly, in order to be controlled with classically defined techniques, the robot must be able to sense its position and orientation accurately and efficiently. The control architecture developed to govern the robotic motion, which was outlined in Chapter 2, is a computer algorithm putting out a control signal. Hardware to drive the robotic joints was selected with this in mind. From the hardware side, the primary input was this control signal from the PC and so it became necessary to find a way to bridge the gap between computer and machine.

3.1.2 Component Block Diagram

Figure 10, shows the diagram of communication between the four subsections that make up the robotic system, namely, the arm, the vision system, the control system and the gloves. All of these subsystems are integral to the success of the system as a whole and the team was responsible for developing the functionality of most of the subsystems from the ground up. The subsystems communicate in different ways, as indicated by the type of lines connecting the two. Each of the subsystems included different sections of the diagram, indicated in the component diagram by similar colors. The Glove Subsystem is in red, the Vision Subsystem is in orange, the Control Subsystem is in green and the Robotic Arm is in blue.

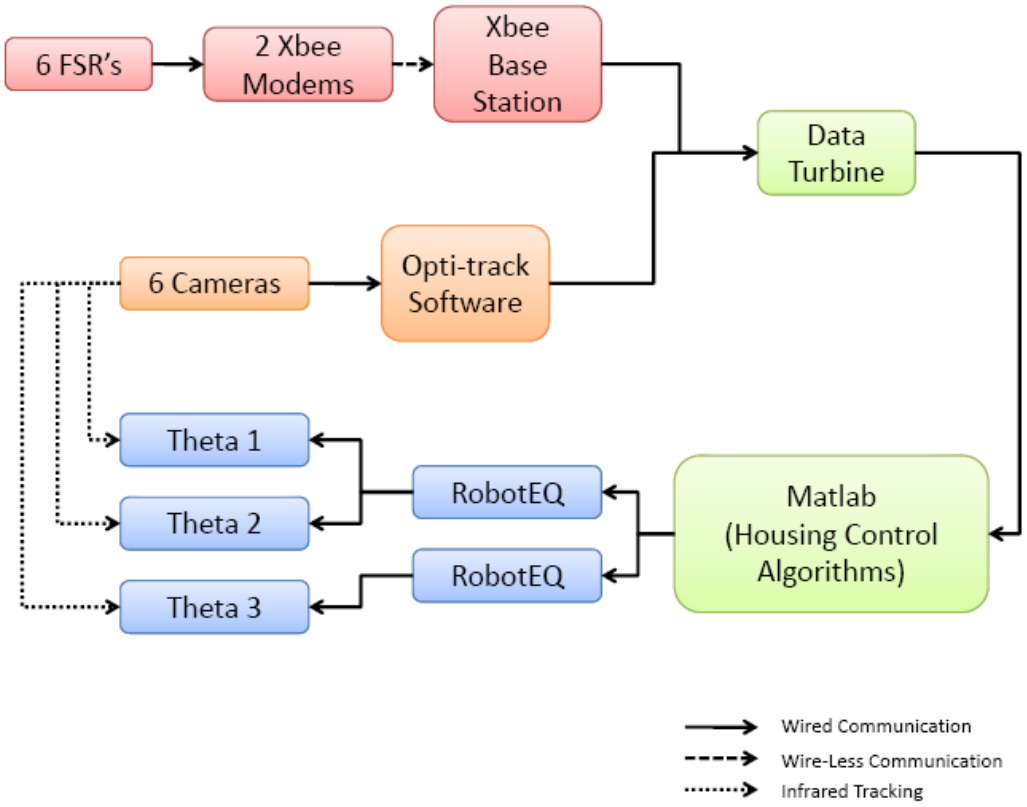


Figure 10: Component Block Diagram for Robotic System

3.2 Robotic Arm Subsystem

The robotic hardware was the initial hurdle the team needed to overcome, which began with the acquisition of a suitable robotic manipulator. Due to budget constraints and availability, the team selected a SCARA configuration robot, the IBM 7545. Attempts had been made at utilizing several Mitsubishi Manipulators, but the team was unable to establish a reliable channel with the robot due to factory installed safety cut-offs and the closed source nature of these robots. More modern robots have began to embrace the concept of open-source programming which espouses that users can almost always access and control even low functionality of the robotic arm, with considerations for safety and preserving the functionality of the manipulator. Because these more modern arms were significantly out of budget, the team selected an older arm that did not have these closed

source limitations, but this meant an older platform. The IBM 7545 had four degrees of freedom and was driven by brushed DC motors, however for the purposes of this project the team focused only on the planar case, thus controlling only three of the degrees of freedom. Its robust structure fit the needs of the team as did its large workspace.

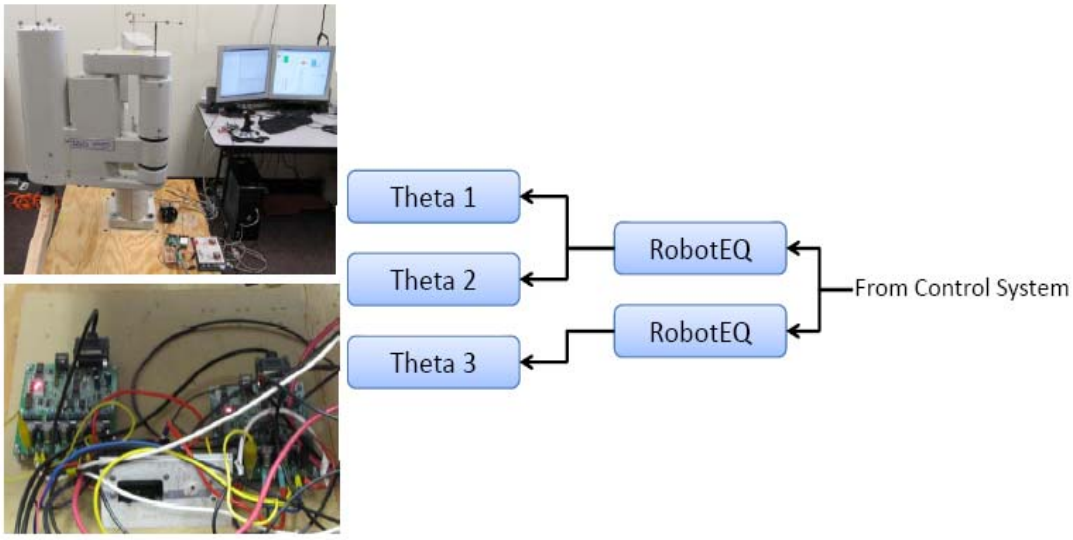


Figure 11: Robotic Hardware

The first step in integrating the new hardware into the system was to establish control of the robotic manipulator. To do so, the team selected the RobotEq Motor Controllers and connected them to the system with a custom made wiring harness. The motor controllers are designed to route designated amounts of power to each of the motors. These motor controllers took serial commands from a PC via a Matlab Script the team wrote, and which is visible in Appendix B. This script took commands from a USB Joystick and assessed how the power should be distributed, then sent the appropriate serial commands to the motors. The power sent by the motor controllers drives each of the four motors, moving the robot.

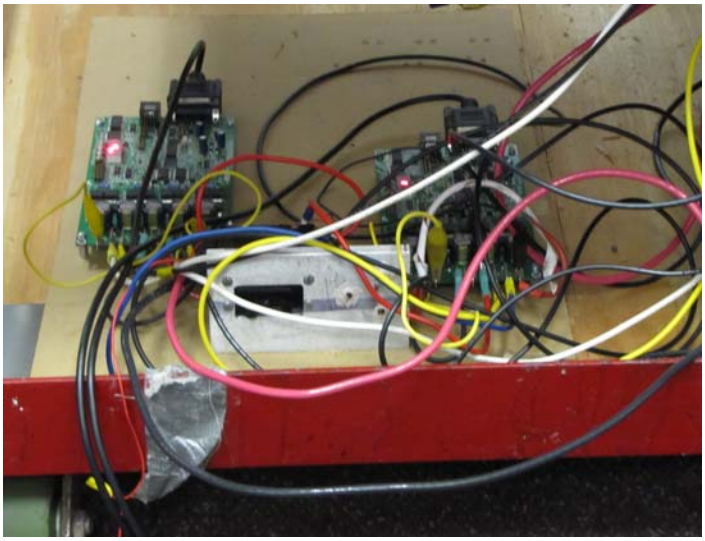


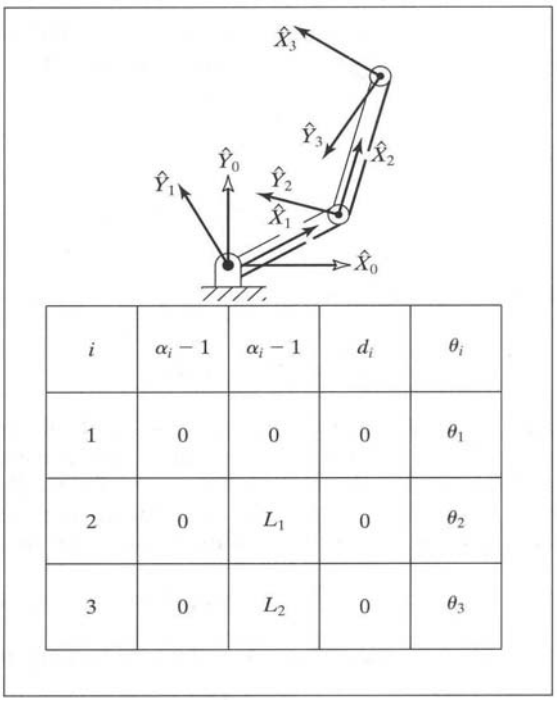
Figure 12: RobotEQ Motor Controllers

The motor controllers were equipped with an encoder module capable of reading and keeping track of the motor encoder counts. These counts could then be converted to angle measurements by a Matlab script. However, when the team began to investigate the integrity of the motor encoders, it became clear that several of the encoders were not functioning. Because the two parts of control are mobility and sensing, this presented a problem which the team overcame by utilizing the external vision system.

3.2.1 Definition of Object Space Reference Frames

Frame transformation was always an important consideration and it became necessary to define an object frame in which the user intent vectors would be given. The SCARA arm had been considered as a three-link revolute arm with a rotating frame at the end effector. Because this was the understanding of the arm, it was easy to assign the object frame as the rotating end-effector frame. Figure 13 shows the diagram of the arm with its assigned frames and DH Parameters from Table 1.

The object being manipulated in the test cases was simply a two-by-four fixed to the end-effector of the robot, as illustrated in Figure 4. The X-axis of the object reference frame, X_3 in the above diagram, is fixed in this object and rotates as the end-effector rotates. The Y-axis of the object frame is perpendicular to the object and is signified by Y_3 in the Figure 13. Section 2.4 discussed definitions of commands based on the user intent, and these commands were generated in the object frame and the controller was defined in the base frame, so the commands were converted from the object frame to the base frame.



i	$\alpha_i - 1$	$\alpha_i - 1$	d_i	θ_i
1	0	0	0	θ_1
2	0	L_1	0	θ_2
3	0	L_2	0	θ_3

Figure 13: Reference Frames and DH Parameter for a Three-Link Revolute Arm

3.3 External Vision System Integration

In order to measure the position and orientation of the robotic arm it was necessary to know the physical dimensions of the arm and the angle measures. This can be done via physical measurements at the axis of each of the joints and the absolute values of the encoders on each of the motors. This information coupled with the type of manipulator, which gives us the kinematic equations, can define the position and orientation within the physical workspace. However, because the manipulator used by the team in this project had encoders that were no longer functioning, the external vision system was utilized.

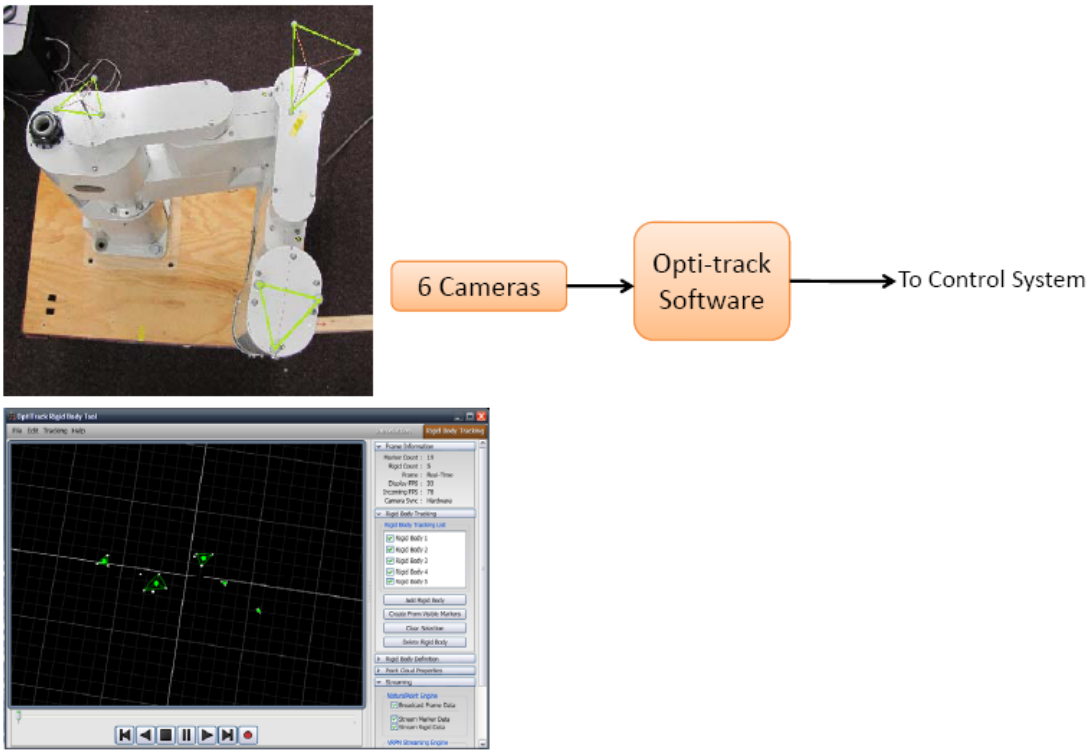


Figure 14: Rigid Body Assignments (marked in green) and the actual tracking data for Vision System Calculations

3.3.1 External Vision System Subsystem

The external vision system was a visual infrared motion capture system similar to those used to capture human motion in computer animation and simulation. The system used six infrared cameras that emitted a light invisible to humans. This light was reflected off markers installed at the joints of the robot as well as the user's hands. The cameras were designed to see and measure the location of each of the markers based on the light reflected back from each of the reflective markers and communicate that to the computer program. This program then took that data, and after a known calibration procedure was able to track the position of each of these markers within the workspace. The markers were arranged in triangular shapes because this constituted a “rigid body” which was a known entity that the computer could associate together and therefore track its position and orientation. As stated, these rigid bodies were placed on the joints of the

robot and the computer program was then able to return the position and orientation of those installed rigid bodies.

That software had the capability to stream those positions and orientations so that they could be accessed by other programs. Other members of the Santa Clara University Robotic Systems Lab had completed the software protocols necessary to stream this data into Matlab in an intelligible way. This was accomplished through a java plugin and Data Turbine. Data Turbine was implemented as a lab standard and has been used in many projects to funnel data to the necessary locations and programs that are using it. The system was set up to steer the vision system data into Matlab for a previous project that was monitoring the position and orientation of several small robots and for the purpose of performing experiments on multi-robot swarm control. However, for this project the team was able to utilize the protocol in place without much modification.

The vision system protocol funneled the position and orientation of each of the rigid bodies into Matlab. This provided the team with the necessary information to find, using geometry and calculus, the angle measures of each of the robotic joints. The vision system also yielded the position of the end-effector, which also proved useful later on in the control process. This information was known because a rigid body had been placed on the end of that robotic link.

3.3.2 Vision System in Matlab

As stated in the Section 3.3.1, the external vision system was used to track the position and orientation of the links of the robot as well as the position of the user and the object to be manipulated. The team was able to utilize java software written by a previous team to accomplish the flow diagram shown in Figure 15 and detailed below.

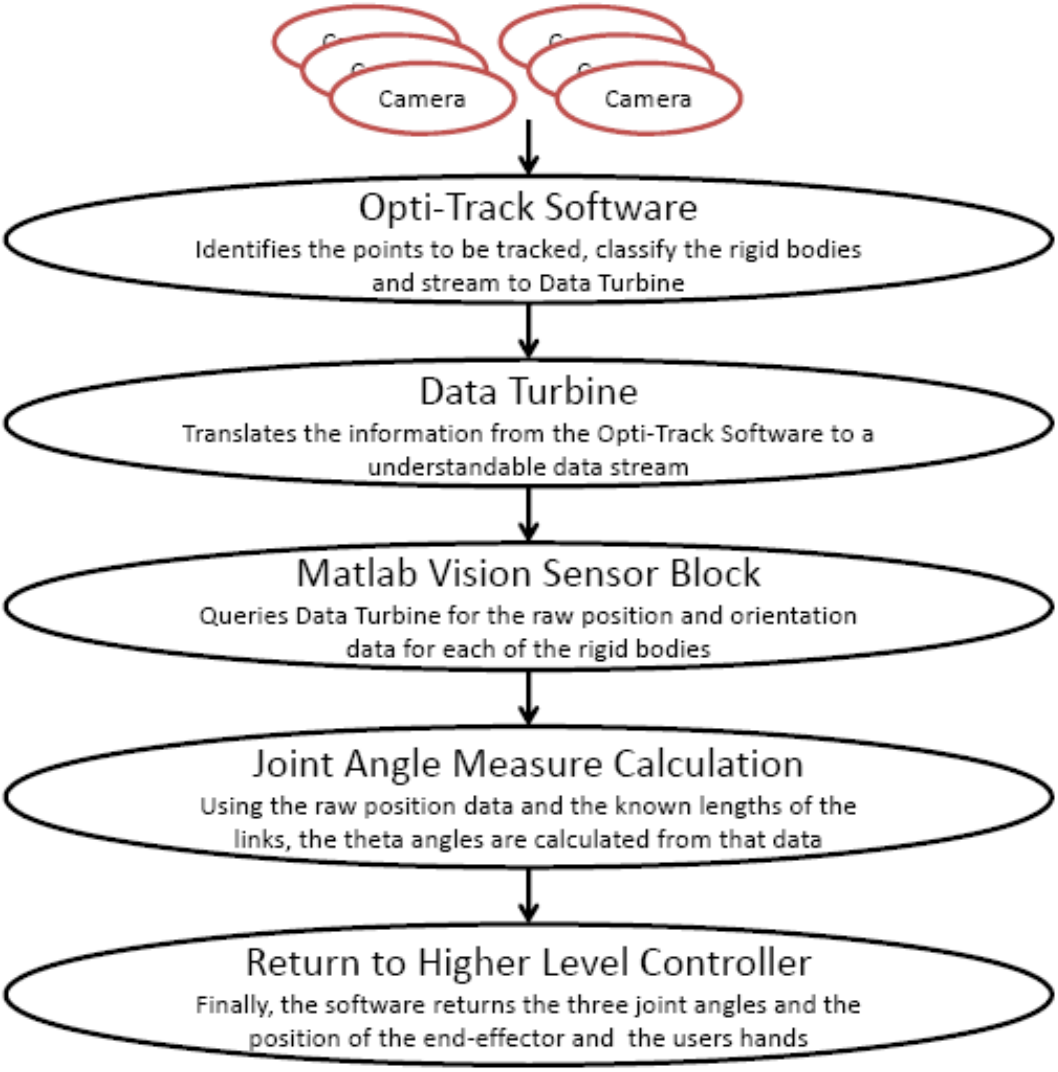


Figure 15: Flow diagram for vision system software

In order to ensure congruency through the controllers the team opted to develop a function to sense the angle measures of each of the joints in a repeatable fashion. This was developed in a control block within the controller that could be reused as the team wished. That block is shown in Appendix 2.8 and Appendix 2.9.

The outputs of this block are numbered 1-7 and are described in Table 3. This sensor block was used in most of the controllers for the project. It utilizes several sub-functions to calculate the desired outputs.

Output Number	Label	Description
1	X3	The Position of the End-effector of the robot in XY coordinates
2	Theta	The vector of joint angles [T1;T2;T3]
3	Theta Dot	The vector of joint velocities [T1_dot;T2_dot;T3_dot]
4	L	The Length between the rigid body at T1 to T2, and at T2 to T3
5	X4	The XY Position of the user's right hand
6	X5	The XY Position of the user's left hand
7	X	The vector of joint positions [X1;X2;X3;X4;X5]

Table 3: Outputs from the Vision System Sensor Block

The first sub-function used by this block is the “Data From Vision System” block. Within this block are several sets of function calls to the Data Turbine and java functions written to retrieve telemetry data from the external vision system These blocks call the functions “Sync Telemetry” and “Vision Telemetry” for the specified object. These objects were defined by the user when starting up the system. The details of these functions are included in the Appendix B. The outputs from this block were the inputs of the higher level block, namely the positions of each of the rigid bodies on the robot and the coordinates of the user's hands.

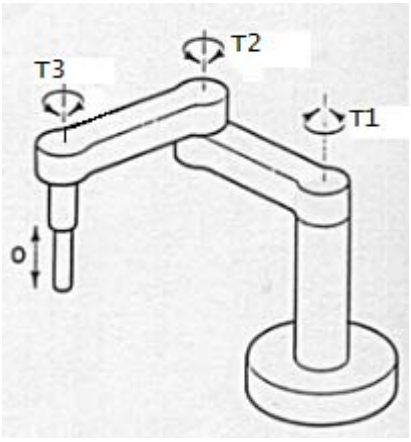


Figure 16: Diagram of DOF measured by the vision system

The other function within the sensor block is a call to a Matlab function titled “Matt_GloveController_SensorCalc” which is included in the Appendix. The function takes in as inputs, the coordinates of the rigid bodies and based on known link lengths computes the joint angles of the robot. In writing this function, there were several

complications, but once completed and tested, this function provided reliable and repeatable measurement of the angles through the entire workspace.

3.4 Tactile Glove Subsystem

One of the goals of this project was to test a novel user interface device to communicate user intent to the robot via a collaboratively manipulated object. In order to do this, that device had to be designed and constructed by the team. This section covers that process and the end result.

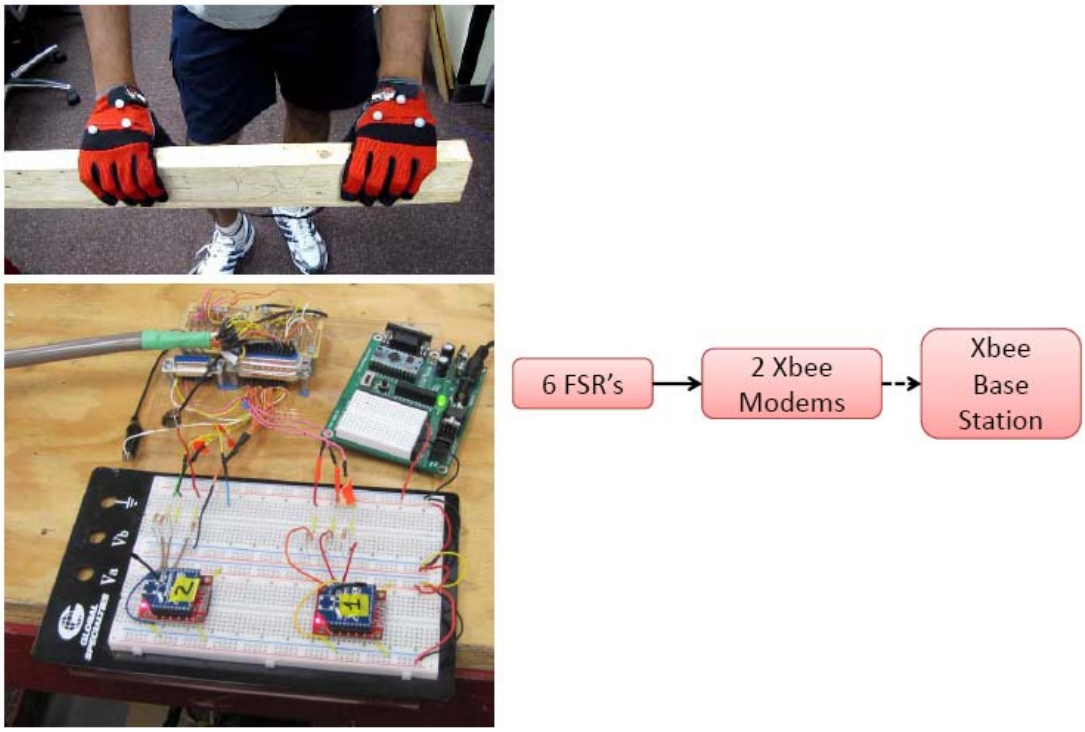


Figure 17: Force sensing glove subsystem

3.4.1 Sensor Selection

In order to develop a human interface device that would communicate intent of the user to the robot controller, the team investigated the fundamental functionality of other user interface devices on the market. However, a fundamental difference was identified. Devices like computer mice and joysticks were observed to primarily communicate the

user's motions and pass those to the computer. In both cases the user moved the device and it tracks those motions and sent them to the computer, but for the purposes of this project it was desirable to track the intent of the user without actually moving the object manipulated. The robot should be able to sense that intent and the robot should move the object.

The team classified several ways that the user could potentially interact with the object. These motions were limited to the horizontal plane due to the hardware limitations of the SCARA configuration robot. A user can push and pull the object with one or both hands. This action was classified as a motion towards or away from the torso, parallel to the forearms. They could drag the object up and down, which would be classified as motion parallel to the torso, perpendicular to the forearms. A twisting motion in the horizontal plane would be push and pull with opposing hands. Several of these motions would demand the ability to sense shear force interaction, however this was a very difficult sensing mode to achieve. Instead the team decided to utilize a sensing protocol, described in Chapter 2, to categorize these different motions.

In order to accomplish this sensing task, the team identified the need to use force or pressure sensors to measure grasping forces that the user exerts on the object. Again, due to budget issues, the team selected Force Sensitive Resistors as they were cheap but easy to implement into the system and would serve the needs of the team. In order to sense the forces exerted by the user, the team designed a pair of gloves with FSR's mounted in the tips of the middle and index fingers and the tip

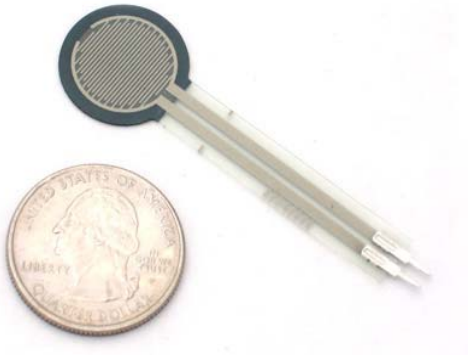


Figure 18: Force Sensitive Resistor Used in Tactile Sensing Glove

of the thumbs of both the right and left hands. Research has shown that humans use these three fingers primarily for manipulation and only use the ring and pinkie fingers for grasping. This was the reason the team deemed it necessary to only install sensors in those fingers so that the team's protocol could sense the grasping forces of these fingers.

The sensors selected were Force Sensitive Resistors with a round 0.5” diameter sensing area. The specifications of the FSR stated that it would vary its resistance depending on how much pressure is being applied to the sensing area, the harder the force, the lower the resistance. The sensitivity of the resistor was rated at 1 MΩ when no pressure was exerted, with sensitivity able to sense forces between 100g – 10kg. The team made several attempts at integrating these sensors into a workable circuitry that would allow the force sensors data to reach the computer with the highest accuracy and lowest latency.

The first attempt made at integrating the force sensitive resistors was a simple voltage divider to test the output of the resistor. In this case, the output was read by a voltage meter, as this was a simple test of output. The output was a measure of voltage from 1000 down to zero, however this was by exerting maximum pressure on the sensor, which was not feasible with human strength at the fingertip. In order to get this analog signal into the PC, and ultimately into the programming environment of Matlab, the team needed to investigate Analog to Digital Conversion methods.

3.4.2 Analog to Digital Conversion

Analog to digital conversion was the process of converting the analog voltage from the FSR to a digital number proportional to the magnitude of the voltage or current. This digital number can then be read by a PC and utilized in the software protocol.

3.4.2.1 Via Basic Stamp

The teams first attempt at A to D conversion was using the Basic Stamp Micro-Controller from Parallax. In order to do the A to D conversion, the Basic Stamp utilized a RC circuit which measured the discharge time of the capacitor. The R part of the circuit was also a voltage divider between a constant resistor and the varying FSR, as in the diagram below. P15 is the input/output pin on the Basic Stamp, which was used to both charge the capacitor for a set time period and also measure the discharge time. Utilizing this discharge time, the basic stamp could then draw a relationship to the relative force exerted on the FSR at that moment. This method was functional, and once the timing was worked out, was able to deliver accurate results on all 6 sensors with minimal latency.

Figure 19: Basic Stamp

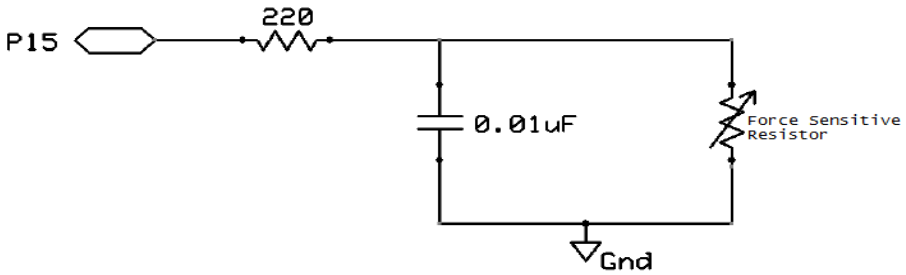


Figure 20: Circuit Diagram for A-D Conversion

The Basic Stamp would then communicate the relative force, based on the discharge time of the capacitor, to the PC via serial communication. Because Matlab was equipped to handle serial communication, a program could be written directly in Matlab to handle that exchange. However, Matlab has shown that its ability to accurately and efficiently handle serial data is limited and that proved to be a problem in this application as well. Writing a sophisticated packet handling scheme was out of the range of this

project and the capabilities of the Basic Stamp. Therefore when the system was implemented as a whole and the team attempted to add the glove sensors to an existing robotic controller, the process became entirely to slow to be considered accurate or efficient.

3.4.2.2 Via Xbee Wireless Modem

Due to the poor performance of analog to digital conversion via the Basic Stamp Micro-Controller, the team was forced to investigate another method of A to D conversion. As mentioned before in Section 2.3, other members had been experimenting with control of multi-robot systems. These robots were maneuvering wirelessly and communicating data back to the controlling PC using Xbee Modem Wireless Communication modules. These modules were capable of doing 3 channels of A to D conversion each, and so they were a logical alternative for the team. The option for wireless communication was a bonus and not necessary for the initial design, but in future iterations of the design, it could have become a key feature.

Figure 21: Xbee Wireless Modem

In order to provide a usable signal to the Xbee Module, the FSR was placed in a voltage divider circuit similar to Figure 22 below. The Xbee module took this analog signal, converted it to a digital signal and communicated it to the receiving modem attached to the PC. As with the external vision system mentioned earlier in the chapter, the Xbee Modules had been used by a previous group who had designed the necessary protocols to plug the digital output of the receiving Xbee module into Data Turbine and access the corresponding java object in Matlab. In order to get the necessary information out of that java script, the string parsing function written to split up the strings of data

into their intelligible parts had to be modified a bit, and that code is included in the appendix.

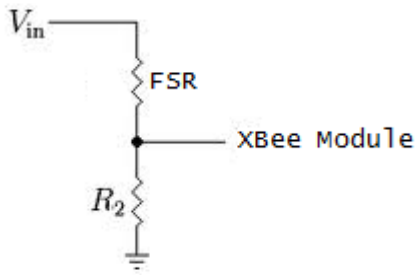


Figure 22: Xbee Wireless Modem Wiring Diagram

3.4.3 FSR Glove Protocol

Once the sensor data from the FSR gloves was communicated to the Matlab program, it was run through a classification protocol to interpret the user intent from the combined sensor data. This protocol is detailed in section 2.5 and was utilized for verification and validation of the system.

3.5 Controller Subsystem

The main parts of the controller subsystem are the control algorithms that have been discussed already in chapter 2, however the system to communicate the sensor data and output data to and from the other systems has not yet been described.

In order to route all the necessary data into the control algorithms, it was necessary to convert the signals using Data Turbine, a data stream processor that streamlines the flow of data from various sensing architectures into a simple stream that can be read by Matlab Software. As stated in section 3.3, much of the work done to establish reliable functionality between the vision system and Xbee Modems through Data Turbine to Matlab, was done for previous work with multi-robot systems. The

controllers utilized that sensor data to determine the necessary control signals to communicate to the robot and utilized the computer's serial ports to send drive commands to the motor controllers.

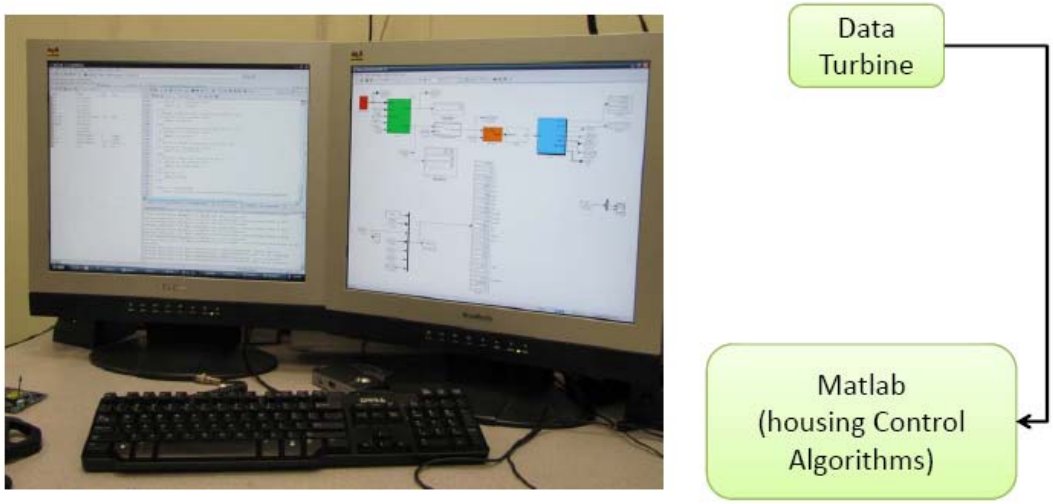


Figure 23: Controller Subsystem

4 Verification and Validation

In order to verify the performance of the controller and the tactile glove input it was necessary to develop a test protocol. The test protocol was designed to record both of these pieces of data efficiently for each of the tests.

4.1 Description and Protocol

The testing performed by the team to verify the robotic controller consisted of driving the robot through its prescribed motions and recording its actual positions. Because the robot controller and tactile glove protocol was designed to receive desired end point velocities in the frame of the object, it was necessary to view the time histories of these motions. Based on the time history plots, insight into the accuracy and effectiveness of the controller and protocol could be determined. To accomplish these tasks, the team drove the robot through the protocol and recorded its position at specific time intervals and exported them into a text file.

The possible motions capable within the designated protocol are translation in both the X and Y direction as well as rotation. Each of the motions was tested several times by the operator and the positions over time were recorded. The tests were also recorded with a video camera and the positions recorded from the vision system were compared to the video recording time stamps.

4.2 Results and Conclusions

The goal of the project was to develop a testbed for a novel human-interface device and then test such a device to ensure its accuracy. The following section covers the testing of the force sensing glove developed by the team and described in the above sections.

4.2.1 Positive X Direction

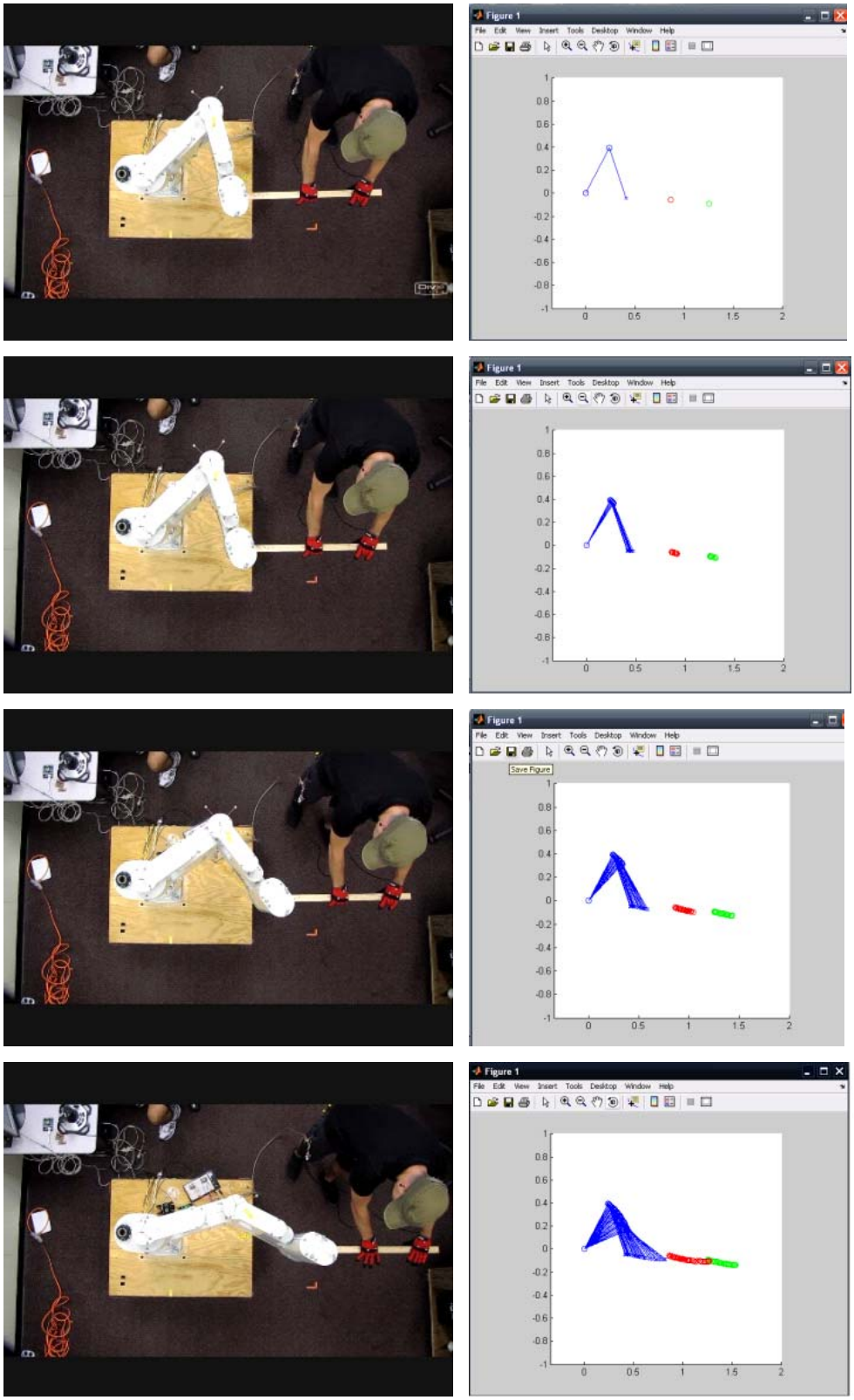


Figure 24: Results for Translation in the Positive X Direction

The first test run was translation in the positive x direction, designated in the frame of the robot. The results of this test were sufficient to determine that the protocol for this motion was working correctly. Within Figure 24, the blue lines indicated the two link SCARA Arm and the red and green circles indicate the left and right hands of the operator. It is apparent that the robot is translating the object in the X-direction. In order to communicate the intent to the robot, the operator grasped the object being manipulated based on the predefined grasp protocol. For this motion, the grasp protocol was to squeeze with the left hand and grip with the right hand. To grip, the operator exerted force on the index and thumb force sensor. To squeeze, the operator would exert force on all three force sensors. The protocol calculation recognized the user intent and communicated a velocity vector to the robot, driving it in the desired direction.

Human error in loading the force sensors and disturbances added by contact with the operator contributed to a small amount of error in desired trajectory, indicated by the time history of the positions of the operator's hands, however it was deemed negligible.

4.2.2 Negative X Direction

The next test run was translation in the negative x direction, designated in the frame of the robot. In order to communicate the intent to the robot, the operator again grasped the object being manipulated based on the predefined grasp protocol. For this motion, the grasp protocol was to squeeze with the right hand and grip with the left hand. To grip the operator exerted force on the index and thumb force sensor. To squeeze, the operator would exert force on all three force sensors. The protocol calculation recognized the user intent and communicated a velocity vector to the robot, driving it in the desired direction.

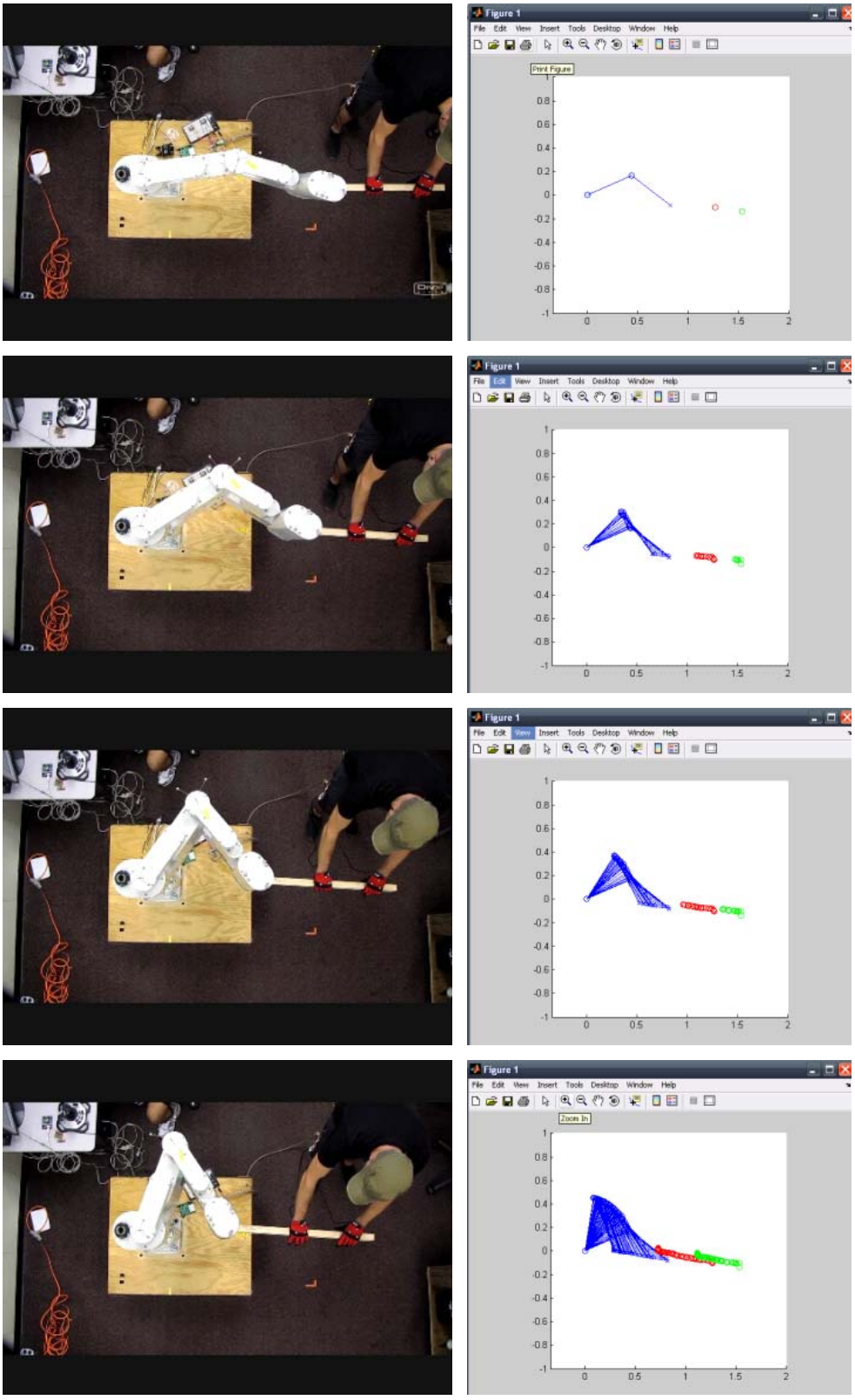


Figure 25: Results for Translation in the Negative X Direction

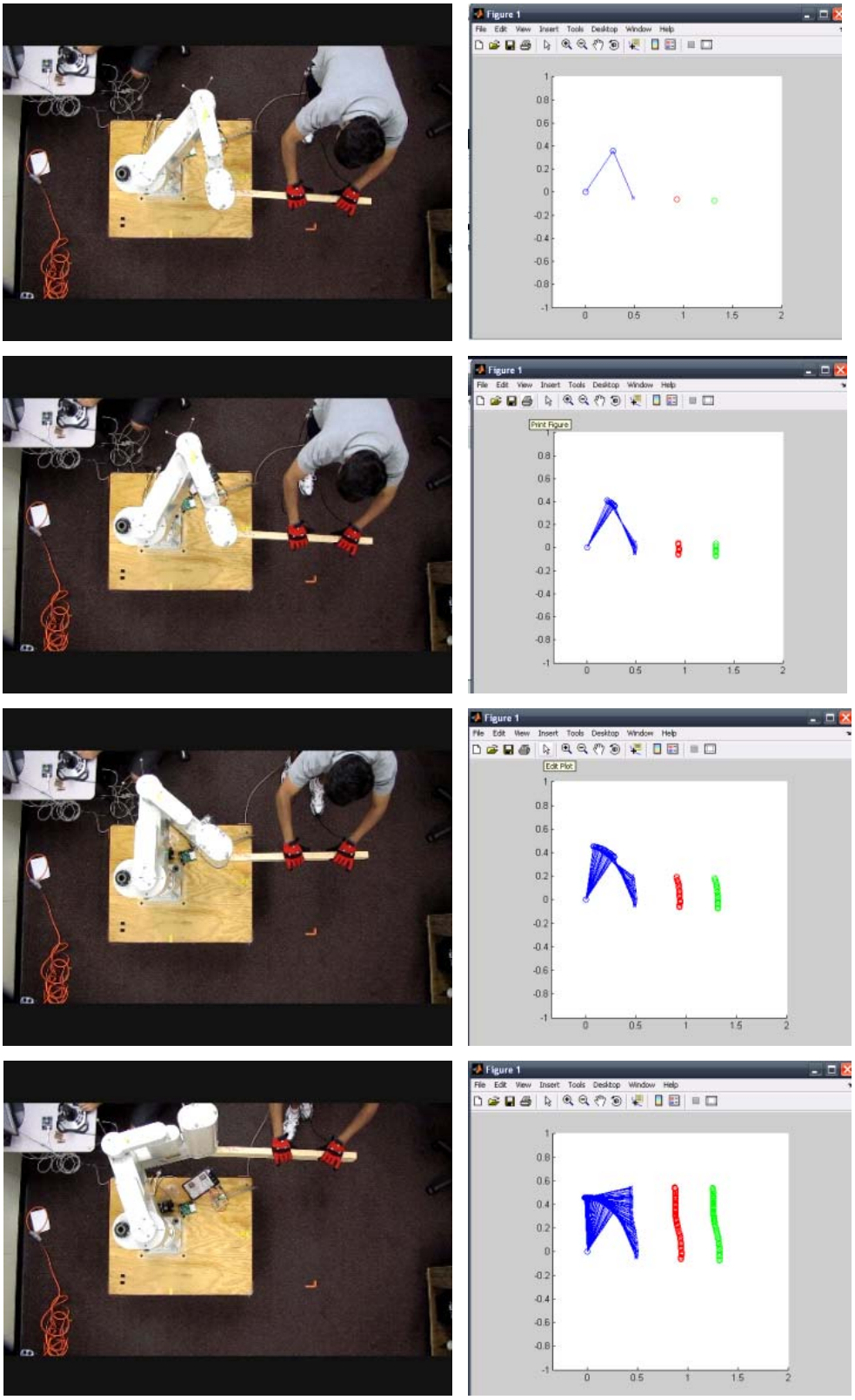


Figure 26: Results for Translation in the Positive Y Direction

The results of this test were sufficient to determine that the protocol for this motion was working correctly. Within Figure 25, the blue lines indicated the two link SCARA Arm and the red and green circles indicate the left and right hands of the operator. It is apparent that the robot is translating the object in the X-direction. Human error in loading the force sensors and disturbances added by contact with the operator contributed to a small amount of error in desired trajectory, indicated by the time history of the positions of the operator's hands, however it was deemed negligible.

4.2.3 Positive Y Direction

The next test run was translation in the positive y direction, designated in the frame of the robot. In order to communicate the intent to the robot, the operator again grasped the object being manipulated based on the predefined grasp protocol. For this motion, the grasp protocol was to pull the object towards the operator. To grasp, the operator exerted force on the force sensors on the index and middle fingers of both hands. The protocol calculation recognized the user intent and communicated a velocity vector to the robot, driving it in the desired direction.

The results of this test were sufficient to determine that the protocol for this motion was working correctly. Within Figure 26 the blue lines indicated the two link SCARA Arm and the red and green circles indicate the left and right hands of the operator. It is apparent that the robot is translating the object in the Y-direction.

4.2.4 Negative Y Direction

The next test run was translation in the negative y direction, designated in the frame of the robot. In order to communicate the intent to the robot, the operator again grasped the object being manipulated based on the predefined grasp protocol.

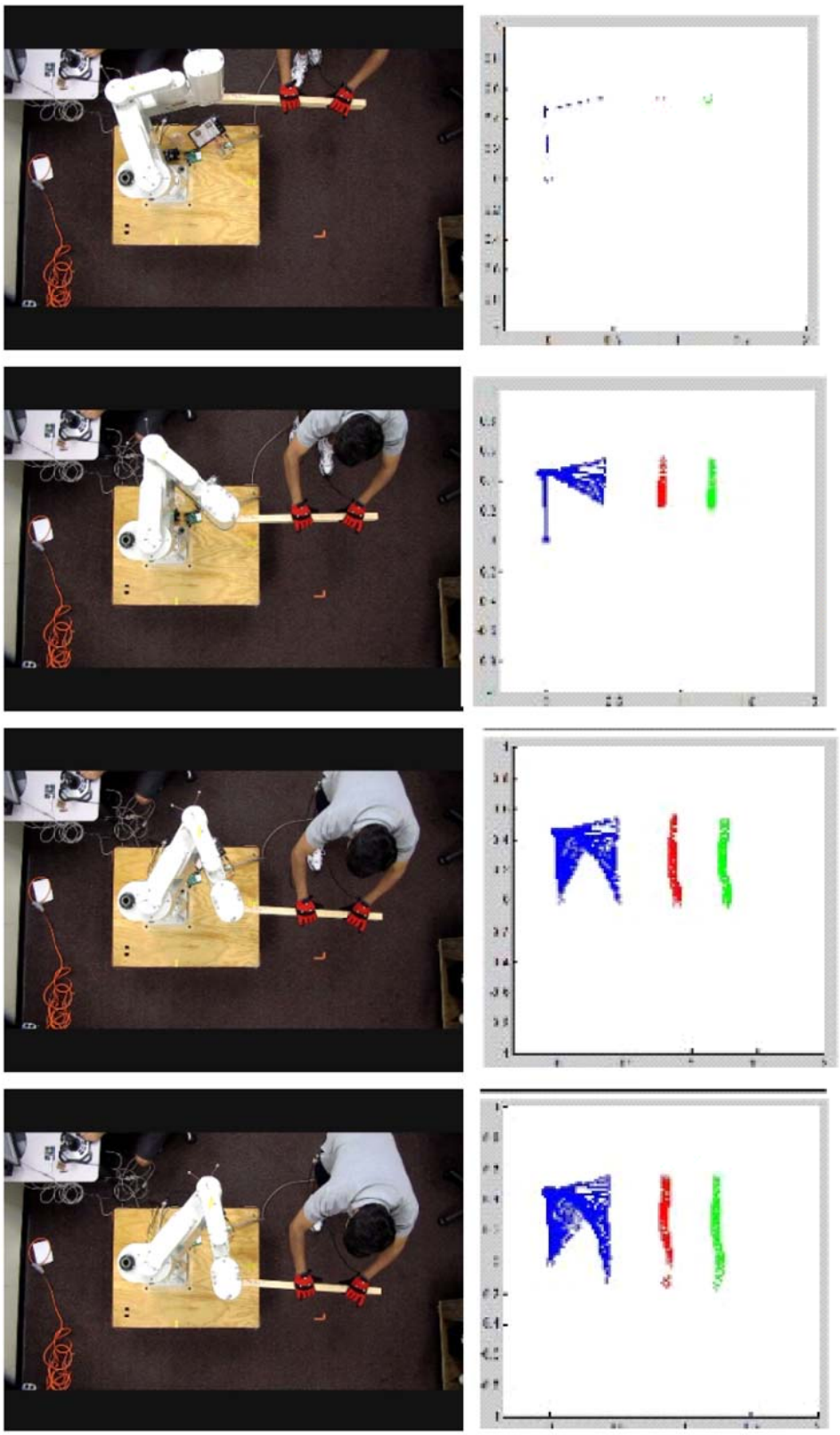


Figure 27: Results for Translation in the Negative Y Direction

For this motion, the grasp protocol was to push the object away from the operator with both hands. To grasp, the operator exerted force on the thumb force sensor on both hands. The protocol calculation recognized the user intent and communicated a velocity vector to the robot, driving it in the desired direction.

The results of this test were sufficient to determine that the protocol for this motion was working correctly. Within Figure 27, the blue lines indicated the two link SCARA Arm and the red and green circles indicate the left and right hands of the operator. It is apparent that the robot is translating the object in the Y-direction. Error in the Y-direction tests was considered negligible.

4.2.5 Rotation in the Clockwise Direction

The next test run was rotation in the clockwise direction, designated in the frame of the robot and centered on the user's hand closest to the robot. In order to communicate the intent to the robot, the operator again grasped the object being manipulated based on the predefined grasp protocol. For this motion, the grasp protocol was to pull with the right hand and push with the left hand. To grasp correctly, the operator exerted force on the thumb force sensor on the left hand and on the sensors mounted on the index and middle fingers of the right hand. The protocol calculation recognized the user intent and communicated a velocity vector to the robot, driving it in the desired direction.

The results of this test were sufficient to determine that the protocol for this motion was working correctly. Within Figure 28, the blue lines indicated the two link SCARA Arm and the red and green circles indicate the left and right hands of the operator. It is apparent that the robot is rotating the object about the operator's hand closest to the robot. Error in the Y-direction tests was considered negligible and due to human error.

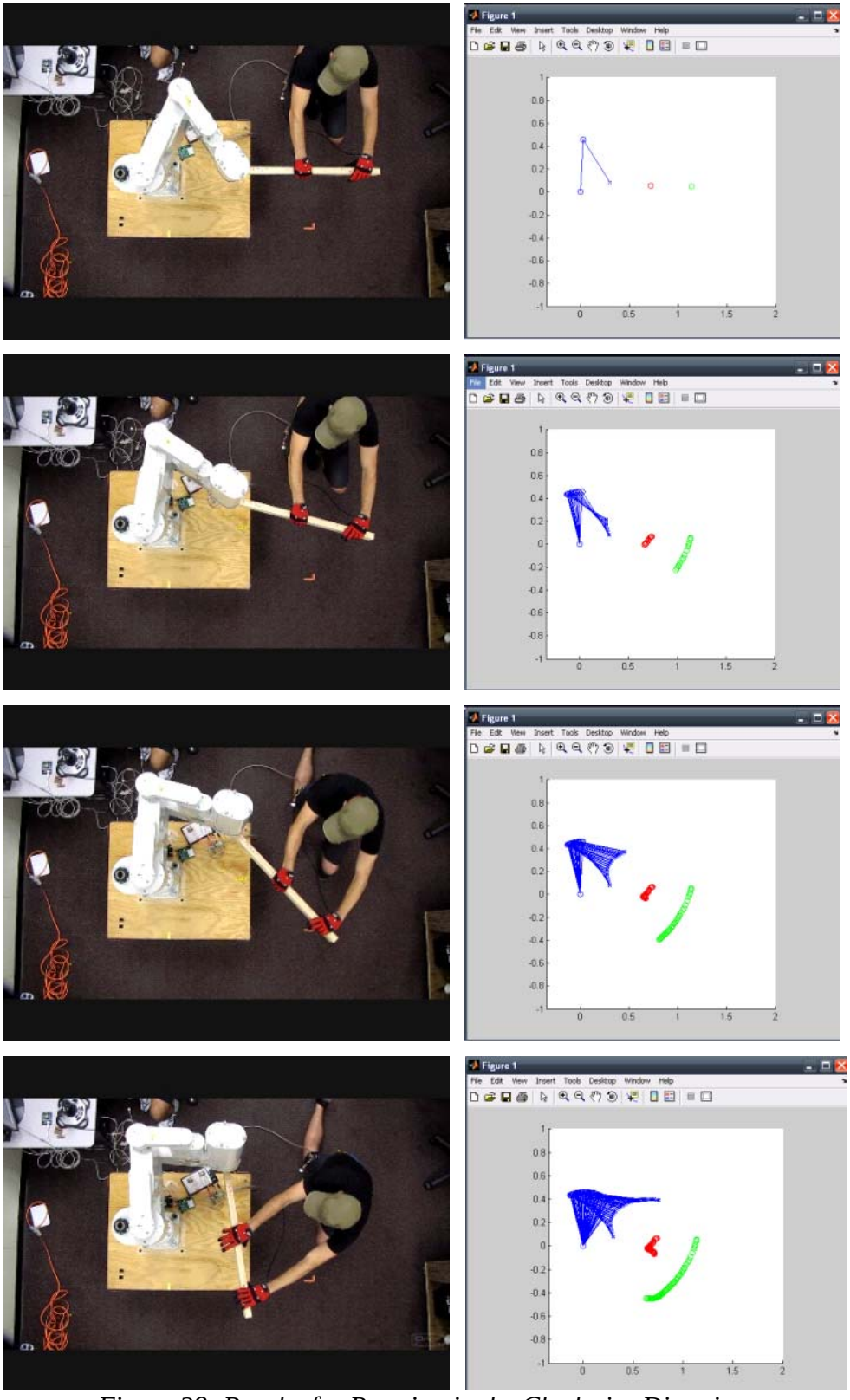


Figure 28: Results for Rotation in the Clockwise Direction

4.3 Validation of Platform

Once it was verified that the system was working effectively and could be accurately controlled by the operator, it was necessary to validate the usefulness of the system to the operator. In order to do so, the team used volunteers to test the system and provide feedback about the overall user experience. Because the goal of the project was to develop a natural and organic user interface as compared to the classical devices, the tests compared the use of the force-sensing gloves to control the robot versus using a joystick to control the robot. The users were then asked for feedback about the overall experience.

4.3.1 Validation Test Protocol

The test protocol for the validation experiment involved asking test operators to complete specific tasks with both user interface devices, the joystick versus the gloves, and requesting feedback on the experience via a written survey. The operators spanned several demographics and varied substantially in their familiarity with robotics. In order to ensure that the users were adequately prepared to complete the tasks, they were given time and guidance to familiarize themselves with both user interface devices and fully understand the tasks. They were allowed to interact with the robot via each interface device until they felt comfortable.

The operators were asked to complete two tasks, a positioning task and an insertion task. Both tests involved rotation and translation to ensure an adequate spread of challenges. The positioning task took the object from an initial position and orientation through subsequent positions to a final position. This eliminated the need for the operator to do their own path planning and allowed the operator to focus on evaluating the performance of each interface device for the individual motions. The insertion task asked

the operator to sequence their moves in such a way that the end of the object would be inserted into the goal location without interference or overshoot. The joystick interface was set up so that the commands were given in the object frame, thus matching the command frame for the glove interface. The tasks were judged loosely for accuracy and speed, however those metrics were not quantified.

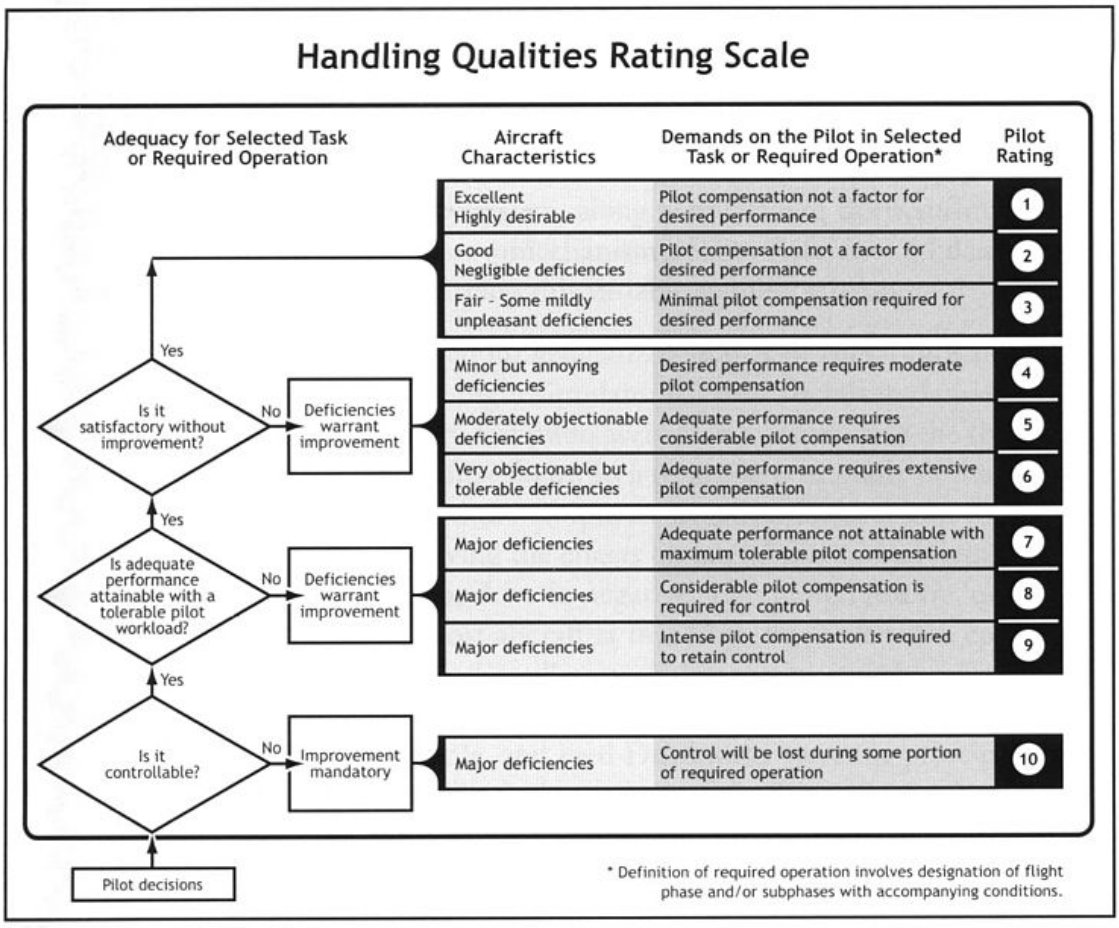


Figure 29: Cooper Harper Scale

Once the two tasks were completed, the operator was asked to complete a brief survey. The template for the survey and the completed surveys are included in Appendix C. The survey was developed from a test survey used to evaluate pilot's opinions and the overall effect of using auto-pilot. The Cooper-Harper Scale is a scale based on feedback from test pilots to gauge the efficacy and controllability of aircraft. Both of these were

blended to give a good overall idea of the usability of the glove interface versus the joystick. The overall results of the surveys are discussed in the following section.

4.3.2 Validation Results

The results of the validation test were the user feedback to the team via a brief survey. That survey, as stated before was a combination of two surveys regarding usability and controllability of aircraft directed at pilots and test pilots. It also utilized the Cooper-Harper Handling Scale, which yields a number on a scale of one to ten based on answers to a few simple questions. The results overall indicated that the gloves interface was intuitive and usable, offering good control and accurate motion.

The first four question were control questions to gather pertinent information from the user about their familiarity with robotics and the process of familiarization with the robotic system. In an attempt to get a good read on the general population the team attempted to gather individuals with a varying degree of familiarity with robotics. To that end the operators vary in familiarity with robotics from one, being very unfamiliar, to ten being very familiar. The test operators were members of the lab with experience working with similar robotic systems, to students in the graduate robotics class, to theology and music majors. The survey also attempted to ensure that the test operators had sufficient time to familiarize themselves with the systems and to interact with the interface devices. All the testers felt that they were given sufficient time to familiarize, but they varied on opinions of difficulty of familiarizing with each system. The glove protocol is designed to be intuitive but it is an unfamiliar interaction device and so it seems to have taken the operators longer to get used to than the more classical joystick. However, the fifth question asked which interface device the user was more comfortable with, and they unanimously agreed on the gloves.

The next three questions regarded the first test, which involved movements with no sequencing, and the difficulty of those moves on a scale of one to five. The operators all rated the moves with the gloves as easier than the moves with the joystick, especially the second move which was to rotate the object about a point on the object. They all agreed that the glove interface made that move easier then the joystick and rated it one to two points easier across the board. The team attributed this to the fact that the glove interface tracks the user's hands and therefore more naturally handles motions that rotate about points that can be specified by hand location. Therefore, once the protocol was explained and the operator was familiar with that motion, it became very easy to rotate about a set point because all the operator must do is place their hand on that point. All test operators also agreed that they felt more in control with the gloves than with the joystick.

The next few questions dealt with the insertion task, and the difficulty of that task with both interface devices. Four of the five operators rated the gloves a full point, on a scale of one to five, easier to use to complete that task than the joystick. The fifth user rated them equally easy to do. The operators commented on the efficiency of sequencing with the gloves and the fact that there was a more natural haptic feedback with the glove interface.

Lastly, the team asked the test operators to rate adjectives they agreed with for each interface device, and complete the Cooper Harper scale for the glove interface. The adjective were, usable, repeatable, intuitive, natural, efficient and accurate, and each was given a rating from one to five, one being untrue, five being true. The scores were averaged and are include in Table 4 below.

Adjective	Test Pilot 1	Test Pilot 2	Test Pilot 3	Test Pilot 4	Test Pilot 5	Average
Usable	4	3	5	5	5	4.4
Repeatable	4	3	5	5	5	4.4
Intuitive	5	3	5	5	5	4.6
Natural	5	4	4	4	4	4.2
Efficient	5	3	5	5	5	4.6
Accurate	4	3	5	4	4	4

Table 4: Adjective Ratings from Test Operator Surveys for Glove Interface

If five is classified as true, all of the adjectives are within the range of being considered true and thus the team felt justified in calling the system intuitive and efficient, usable and repeatable, natural and accurate. In comparison to the ratings for the joystick, included in Table 5, the team felt justified in saying that the gloves outperformed the joystick in the context of this test.

Adjective	Test Pilot 1	Test Pilot 2	Test Pilot 3	Test Pilot 4	Test Pilot 5	Average
Usable	4	3	5	3	3	3.6
Repeatable	2	2	3	2	2	2.2
Intuitive	2	2	2	3	3	2.4
Natural	2	2	2	2	2	2
Efficient	3	2	3	2	1	2.2
Accurate	2	3	2	2	2	2.2

Table 5: Adjective Ratings from Test Operator Surveys for Joystick Interface

Lastly, the operators were asked to complete the Cooper-Harper Scale Handling Scale designed for determining and rating the handling of test aircraft. The scale is a series of questions regarding the handling of the system and it yield a number from one to ten. The test operators completed the scale and returned the results in the following table.

Tester	Cooper-Harper Scale	Meaning
Test Pilot 1	2	Good, Negligible Deficiencies
Test Pilot 2	2	Good, Negligible Deficiencies
Test Pilot 3	4	Minor but annoying deficiencies
Test Pilot 4	2 to 3	Good to Fair, Some Deficiencies
Test Pilot 5	2	Good, Negligible Deficiencies

Table 6: Cooper-Harper Rating from Test Operators

Overall the Cooper-Harper Scale determined that the system was controllable and usable, however there were some deficiencies that made the system slightly harder to use. Through their comments, the test operators identified these deficiencies. The chief complaint was lack of dynamics in the sensors, which was a known issue and is discussed greater in the Future Works section of Chapter 5. Also one of the test operators pointed out that the gloves were the wrong size for them and thus placed the sensors in a less than optimal position for grasping. This made it difficult to trigger the sensors accurately and have the robotic controller register the correct user intent.

Overall, having the test operators validate the performance and usefulness of the glove interface was a success. The team had claimed that the glove interface would be more accurate and efficient, usable and intuitive, than manipulating an object with a joystick and the user feedback confirmed that. The team thanks the testers for their participation and feedback.

5 Conclusion

5.1 Summary

The main objective of the project was to demonstrate a novel approach to human robot interaction and robot-assisted manipulation, utilizing haptic and spatial information to specify robot actuated object motion. By developing a organic human interface device to communicate operator intent to the robot through interaction with the object being manipulated, the team attempted to improve the nature of human-robot interaction for robotic assisted manipulation. To do this, a protocol was written to translate the grasping forces exerted on the object by the operator as well as the position of the operator grasp on the object, into user intent and desired motion. Jacobian based controllers were used to translate user intent into velocity vectors defining the motion of the object, and were also utilized as a safety protocol to ensure that the robot and/or the object being manipulated were never driven into the operator. In order to track the position of the robot, the object being manipulated and the the operator, the team took advantage of the external vision system. The team utilized a SCARA configuration manipulator to demonstrate the effectiveness of their human-robot interaction.

The team tested this robot-assisted manipulation in several cases and illustrated the results of those tests, as well as gathering feedback from a group of test operators to validate the usability of the gloves as a human interface device versus a more classical joystick human interface. The verification tests, discussed in detail in chapter 4, show that the controllers worked well and the object was driven accurately in the direction defined by the user. The validation test showed that the system was a usable and intuitive human interface to control the robot. In fact the test operators' feedback

indicated that there is an actual benefit to this system over the joystick interface. The operators needed to use fewer moves and completed tasks more naturally and with less trouble than with the joystick device, thus showing that it made the operator's job easier when manipulating an object with the assistance of a robot. Overall, the project was successful in illustrating the value of this and similar human interface devices for robot-assisted manipulation tasks.

In the future, robots will take on a much more significant role in our everyday lives and the lives of people around us by taking on many of the repetitive, dangerous and mundane jobs that we have to do today. As this occurs, the necessity for natural, organic communication between humans and robots is more and more important. By attempting to mirror the way human communicate intention during manipulation tasks the team has made a small contribution to the large body of work that will ultimately be the human-robot interaction protocol of the future.

5.2 Future Work

The team accomplished the goals it set for itself for this project, however there is much that can be done in the future with this project. One of the potential future areas of work is using the human interface force sensing glove with a more complex robot with more degrees of freedom. The control algorithms developed for this project are scalable to a more complex robot and that would provide a more effective and useful manipulation experiment as well as offering a greater range of motion and more interesting manipulation tasks. Realistically, the team would like to move out of the single plane and into more interesting, three-dimensional manipulation.

The glove itself lacked a dynamic range and modification of the sensors used in

the glove to register the user intent would provide a more organic user interaction and make for smoother and more natural looking motions. The ability to sense sheer forces at the finger tip would allow for a much more intuitive user protocol and grasp environment. If it was possible to move the sensing off of the gloves and into a wrist sensor, it would allow the robot to grasp an object and the operator to interact with any object without the need to put special gloves on. There are many improvements that can be made to the sensing part of this system now that a successful proof of concept has been completed.

One of the limitations highlighted early in this paper was the need for accurate external position sensing of the object and the human operator. The team utilized an external vision system to accomplish this, however this limits the user to operating within the range of that vision system. As technology advances, the use of stereo vision cameras, optical encoders and resistive flex sensors could be combined to register the position and orientation of the robot, object and operator. This would allow the system to be utilized in a more mobile context would definitely increase usefulness.

The team felt that the project has shown itself to be a successful proof of concept. Human-robot interaction will one day be a very integral part of our every day lives and the more natural and easy it is to interact, the better our lives can be.

Bibliography

Caldwell, D.G.; Kocak, O.; Andersen, U.; , "Multi-armed dexterous manipulator operation using glove/exoskeleton control and sensory feedback," *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on* , vol.2, no., pp.567-572 vol.2, 5-9 Aug 1995
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=526273&isnumber=11433

Calinon, S.; Evrard, P.; Gribovskaya, E.; Billard, A.; Kheddar, A.; , "Learning collaborative manipulation tasks by demonstration using a haptic interface," *Advanced Robotics, 2009. ICAR 2009. International Conference on* , vol., no., pp.1-6, 22-26 June 2009
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5174740&isnumber=5174665

Craig, John J. *Introduction to Robotics: Mechanics and Control*. 3rd ed. Upper Saddle River, NJ: Pearson/Prentice Hall, 2005. Print.

De Carli, D.; Hohert, E.; Parker, C.A.C.; Zoghbi, S.; Leonard, S.; Croft, E.; Bicchi, A.; , "Measuring intent in human-robot cooperative manipulation," *Haptic Audio visual Environments and Games, 2009. HAVE 2009. IEEE International Workshop on* , vol., no., pp.159-163, 7-8 Nov. 2009
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5356124&isnumber=5356106

Evrard, P.; Gribovskaya, E.; Calinon, S.; Billard, A.; Kheddar, A.; , "Teaching physical collaborative tasks: object-lifting case study with a humanoid," *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on* , vol., no., pp.399-404, 7-10 Dec. 2009
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5379513&isnumber=5379509

Kosuge, K.; Yoshida, H.; Fukuda, T.; , "Dynamic control for robot-human collaboration," *Robot and Human Communication, 1993. Proceedings., 2nd IEEE International Workshop on* , vol., no., pp.398-401, 3-5 Nov 1993
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=367685&isnumber=8413

Monje, C.A.; Pierro, P.; Balaguer, C.; , "Pose control of the humanoid robot RH-1 for mobile manipulation," *Advanced Robotics, 2009. ICAR 2009. International Conference on* , vol., no., pp.1-6, 22-26 June 2009
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5174765&isnumber=5174665

Martin, T.B.; Ambrose, R.O.; Diftler, M.A.; Platt, R., Jr.; Butzer, M.J.; , "Tactile gloves for autonomous grasping with the NASA/DARPA Robonaut," *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on* , vol.2, no., pp. 1713- 1718 Vol.2, April 26-May 1, 2004 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1308071&isnumber=29025

Mingrino, A.; Bucci, A.; Magni, R.; Dario, P.; , "Slippage control in hand prostheses by sensing grasping forces and sliding motion," *Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on* , vol.3, no., pp.1803-1809 vol.3, 12-16 Sept 1994 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=407614&isnumber=9158

Reed, K.B.; Peshkin, M.A.; , "Physical Collaboration of Human-Human and Human-Robot Teams," *Haptics, IEEE Transactions on* , vol.1, no.2, pp.108-120, July-Dec. 2008 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4624256&isnumber=4694814

Siciliano, Bruno. *Robotics Modelling, Planning and Control*. London [etc.: Springer, 2009. Print.

Tarchanidis, K.N.; Lygouras, J.N.; , "Data glove with a force sensor," *Instrumentation and Measurement, IEEE Transactions on* , vol.52, no.3, pp. 984- 989, June 2003 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1213693&isnumber=27281

Tickel, T.; Hannon, D.; Lynch, K.M.; Peshkin, M.A.; Colgate, J.E.; , "Kinematic constraints for assisted single-arm manipulation," *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on* , vol.2, no., pp. 2034- 2041 vol.2, 2002 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1014840&isnumber=21842

Appendix A:

Position Kinematics:

The manipulator kinematics are defined from the link parameters of the robot and are specific to its current location. In order to define the link parameters the team selected the Denavit-Hartenberg Parameters (DH Parameters) for the specific manipulator. These are derived by inference from the manipulator mechanism and are non-unique.

These parameters are then utilized to compute the individual transformations for each of the links. In order to do this, the DH Parameters are plugged into the following equation

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & \alpha_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i s\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

and then those frame transformation are are concatenated to find the single transformation that related the end effector frame (frame 3) to the base frame (frame 0).

$$ {}^{0}_{3}T = {}^{0}_{1}T \; {}^{1}_{2}T \; {}^{2}_{3}T \quad (2) $$

$$ {}^{B}_{W}T = {}^{0}_{3}T = \begin{bmatrix} c_{123} & -s_{123} & 0 & l_1 c_1 + l_2 c_{12} \\ s_{123} & c_{123} & 0 & l_1 s_1 + l_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

Inverse Kinematics:

The goal of the inverse kinematic function is to specify the angle measures necessary to place the end-effector at a desired location. Therefore, the input required three pieces of information, the coordinates of the end-effector and the orientation of that frame, defined by: x, y, and ϕ respectively. This specification converts the previous equation to

$$ {}^{0}_{3}T = \begin{bmatrix} c_\phi & -s_\phi & 0 & x \\ s_\phi & c_\phi & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4} $$

which leads to a set of four non-linear equations that must be solved for θ_1 , θ_2 , and θ_3 .

$$ c_\phi = c_{123} \tag{5} $$

$$ s_\phi = s_{123} \tag{6} $$

$$ x = l_1 c_1 + l_2 c_{12} \tag{7} $$

$$ y = l_1 s_1 + l_2 s_{12} \tag{8} $$

In order to solve these equations, it was necessary to first square equations to obtain

$$ x^2 + y^2 = l_1^2 + l_2^2 + 2 l_1 l_2 c_2 \tag{9} $$

where the following identities were used

$$ c_{12} = c_1 c_2 - s_1 s_2 \tag{10} $$

$$ s_{12} = c_1 s_2 + s_1 c_2 \tag{11} $$

solving equation 9 for c_2

$$ c_2 = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2} \tag{12} $$

at this point in the calculation, the solution algorithm would check to ensure that the desired position is within reach of the manipulator by checking to make sure that the solution of equation 12 is between -1 and 1. If this condition is not met, then the desired position is outside the robots workspace and can never be reached. Assuming the desired position is with in the workspace, the next step is to solve for s_2 .

$$ s_2 = \pm\sqrt{1 - c_2^2} \tag{13} $$

and from here the angle θ_2 can be calculated using the two argument arc-tangent.

$$\theta_2 = Atan2(s_2, c_2) \tag{14}$$

Having found θ_2 , solutions for equations 7 and 8 can be solved for θ_1 by first changing their format to

$$x = k_1 c_1 - k_2 s_1 \tag{15}$$

$$y = k_1 s_1 + k_2 c_1 \tag{16}$$

where

$$\begin{aligned} k_1 &= l_1 + l_2 c_2 \\ k_2 &= l_2 s_2 \end{aligned} \quad . \tag{17}$$

After a bit of manipulation, which can be found in detail in Chapter 4 of Craig's book, the solution for θ_1 can be found.

$$\theta_1 = Atan2(y, x) - Atan2(k_2, k_1) \tag{18}$$

Lastly, the solution to θ_3 can be found using equation 5 and 6.

$$\theta_1 + \theta_2 + \theta_3 = Atan2(s_\phi, c_\phi) = \phi \tag{19}$$

And because θ_1 & θ_2 have already been found, θ_3 can be found easily.

Velocity Kinematics:

In order to develop the necessary controller, the team had to determine the correct Jacobian matrix for the robot geometry. This is done by examining the mechanism of the robot, attaching reference frames and calculating the velocity of the end-effector in different reference frames. The reference frames are attached to the robot in the same configuration as was used in the definition of manipulator kinematics. Because the link transformations will be used in the calculations of velocity propagation we will compute them

$$
{}^{0}_{1}T=\begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}
$$

$$
{}^{1}_{2}T=\begin{bmatrix} c_2 & -s_2 & 0 & l_1 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{20}
$$

$$
{}^{2}_{3}T=\begin{bmatrix} c_3 & -s_3 & 0 & l_2 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}
$$

Then, the following equations are used sequentially form link to link to propagate the velocity through the serial chain manipulator.

$$
{}^{i+1}\omega_{i+1}={}^{i+1}_{i}R\,{}^{i}\omega_i+\dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} \tag{21}
$$

$$
{}^{i+1}v_{i+1}={}^{i+1}_{i}R\,({}^{i}v_i+{}^{i}\omega_i\times{}^{i}P_{i+1}) \tag{22}
$$

These calculations ultimately yield a matrix representation of the velocity in the end-effector frame. This matrix representation can be converted to another frame using the standard frame transformation protocol. In order to define the Jacobian matrix, we use the following Jacobian identity.

$$
{}^{i}v={}^{i}J\,(\Theta)\dot{\Theta} \tag{23}
$$

In the case of the three link planar arm, which is the model used for the SCARA manipulator in this project., the Jacobian in the base frame is defined as

$$
{}^{0}J=\begin{bmatrix} -L_1 s_1-L_2 s_{12}-L_3 s_{123} & -L_2 s_{12}-L_3 s_{123} & -L_3 s_{123} \\ L_1 c_1+L_2 c_{12}+L_3 c_{123} & L_2 c_{12}+L_3 c_{123} & L_3 c_{123} \\ 1 & 1 & 1 \end{bmatrix} \tag{24}
$$

however for the desired controller, it was necessary to express the relationship

$$
\dot{\Theta}={}^{i}J^{-1}(\Theta)v={}^{i}J^{-1}(\Theta)\,\dot{X} \tag{25}
$$

where $\dot{X}$ is the vector representation of the velocity of the end-effector. Calculation of the inverse of the Jacobian matrix is tricky and changes with each new orientation of the robot. In order to calculate the inverse Jacobian, the team wrote a Matlab sub-function that is included in the appendix. The symbolic result of this function yield the following matrix

which is used with equation 25 to to calculate the desire joint rates from the desired velocities in the end effector frame. These joint rates are then converted to joint torques, which are passed to the robot block and motor controllers.

$$J^{-1} = \begin{bmatrix} \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{c_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{s_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{-l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{-(l_1 \times c_1 + l_2 \times c_{12})}{c_{12}} & \frac{-(l_1 \times s_1 + l_2 \times s_{12})}{s_{12}} & \frac{-l_3 \times (-c_{123} \times l_2 \times s_{12} + s_{123} \times l_2 \times c_{12}) - (l_1 \times c_1 + s_{123} \times l_2 \times c_{12})}{-l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{c_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{s_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{-l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{(l_1 \times c_1 + l_2 \times c_{12})}{c_{12}} & \frac{(l_1 \times s_1 + l_2 \times s_{12})}{s_{12}} & \frac{l_3 \times (c_{123} \times l_2 \times s_{12} + s_{123} \times l_1 \times c_1 + s_{123} \times l_2 \times c_{12}) + (l_1 \times c_1 + s_{123} \times l_2 \times c_{12})}{l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{c_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{s_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{-l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{c_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{s_{12}} & \frac{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)}{-l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})} \\ \frac{(-l_2 \times c_{12} \times s_1 - l_2 \times s_{12} \times c_1) + l_2 \times s_{123} \times c_1)}{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)} & \frac{(-l_2 \times c_{12} \times s_1 + l_2 \times s_{12} \times c_1) + l_2 \times s_{123} \times c_1)}{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)} & \frac{l_3 \times (c_{123} \times s_{12} - s_{123} \times c_{12})}{l_1 l_2 (-c_{12} \times s_1 + s_{12} \times c_1)} \end{bmatrix} \quad (26)$$

Appendix B:

1 Matlab Source Code

1.1 initialize.m

```
% Initialize

javaaddpath('c:\Program Files\RBNB\V3.1\bin\rbnb.jar');
J1=optiTrackDTConnect('localhost','3333','1');
J2=optiTrackDTConnect('localhost','3333','2');
J3=optiTrackDTConnect('localhost','3333','3');
J4=optiTrackDTConnect('localhost','3333','4');
J5=optiTrackDTConnect('localhost','3333','5');
% J6=optiTrackDTConnect('localhost','3333','6')
glove1=FSRGloveDTConnect('COM2','localhost','3333','0001');
glove2=FSRGloveDTConnect('COM2','localhost','3333','0002');
```

1.2 IBMSerialInitA.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Constructs the serial port object for theta1 and theta2. %
    global s1;
    s1 = serial('COM1', 'BaudRate', 9600, 'DataBits', 7);        %
    set(s1, 'stopbits', 1, 'Parity', 'even', 'Terminator','CR'); %
%                                                                %
%   Connects the serial port object to the serial port:          %
    fopen(s1);                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%s1
%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.3 IBMSerialOut.m

```
function y = IBMSerialOutA(m1,m2)

%Serial Port Objects
global s1;

%Round Values
m1 = round(m1);
m2 = round(m2);

%Motor 1
if(m1>=0)
    fprintf(s1, '!A%0.2X\r', m1);    %Sends motor1 Command
else
    m1 = abs(m1);
    fprintf(s1, '!a%0.2X\r', m1);    %Sends motor1 Command
end

%Motor 2
if(m2>=0)
```

```
    fprintf(s1, '!B%0.2X\r', m2);    %Sends motor2 Command
else
    m2 = abs(m2);
    fprintf(s1, '!b%0.2X\r', m2);    %Sends motor2 Command
end


y=0;
```

1.4 cascadeDTConnect.m

```
% cscadeDTConnect.m

% dataTurbine matlab file for test
%
% < Tests Matlab connections to DataTurbine and gets Optical Tracking
Data >

%function cscadeDTConnect(ipAddress, port, robotName)
function self = cscadeDTConnect(ipAddress, port, robotName)
self = create_object;

client_name = 'controller';
client_name_source = 'XBeeModemPlugin';
cscade_name = 'CSCADE_source';
%controller_name = 'controller_source';

%javaaddpath('C:\Program Files\RBNB\V3.2B1\bin\rbnb.jar');

import com.rbnb.sapi.*;

name = strcat(cscade_name, '/', robotName);

srcTest = Source(1, 'none', 0);
snkTest = Sink;

%snkTest.OpenRBNBConnection(ipAddress:port, channelName);
snkTest.OpenRBNBConnection(strcat(ipAddress, ':', port), client_name);
srcTest.OpenRBNBConnection(strcat(ipAddress, ':', port),
client_name_source);

rmap = ChannelMap;

rmap.Add(name);
%rmap.Add('robotName');

snkTest.Request(rmap, -10.0, 20.0, 'newest');
%snkTest.Subscribe(rmap, 0, 1, 'newest');

cmap = ChannelMap;
%cmap.Add(controller_name);
cmap.Add(client_name_source);

dataTable = hashtable;
%m_received = true;
```

```
dataTable.('Disabled') = false;
dataTable.('received') = false;
dataTable.('time') = 0.0 ;

dataTable.('rb_ID') = 0;
dataTable.('rb_x') = 0;
dataTable.('rb_y') = 0;
dataTable.('rb_z') = 0;
dataTable.('rb_qx') = 0;
dataTable.('rb_qy') = 0;
dataTable.('rb_qz') = 0;
dataTable.('rb_qw') = 0;
dataTable.('rb_num') = 0;


srcTest.Register(cmap);


snkTest.Monitor(rmap, 0);

disp(strcat('Current Channel:', m_getName()));


%Change for longer update time
%for j = 1:1000
%    m_sync();
%end


%m_disconnect();

%end of main function
%-----------------------
%Helper functions below

%m_getName
%
%@returns robot name string
%
function [robotName] = m_getName()
    robotName = name;
end

%m_disconnect
%
%Disconnects the robot from Data Turbine
%
function m_disconnect()
    snkTest.CloseRBNBConnection();
    srcTest.CloseRBNBConnection();
end

%m_getTelemetry
%
% @returns telemetry:hashtable containing the latest telemetry
%
```

```
function [telemetry] = m_getTelemetry(keys)
    telemetry = repmat(double(0), 1, length(keys));

    for i = 1:length(keys)
        telemetry(i) = double(dataTable.(keys{i}));
    end
end

%m_send
%
%Sends velocity commands to robot
%
%@param inputVector: 1x2 vector [forwardVelocity rotVelocity]
%
function m_send(inputVector)
    cmap.PutTimeAuto('timeofday');
    cmap.PutDataAsByteArray(0, inputVector);
    srcTest.Flush(cmap, true);
end

%m_sync
%
%Fetches the latest telemetry data from the robot
%
function m_sync()
    testMap = snkTest.Fetch(40);
    if(testMap.NumberOfChannels < 1)
        dataTable.('received') = false;
        %disp('Number of Channels =');
        %disp(dataTable.('received'));
        return;
    end

    result = testMap.GetData(0);
    dataTable.('received') = true;

    dataTable.('rb_ID') = swapbytes(typecast(result(1:4), 'single'));
    dataTable.('rb_x') = double(swapbytes(typecast(result(5:8),
'single')));
    dataTable.('rb_y') = double(swapbytes(typecast(result(9:12),
'single')));
    dataTable.('rb_z') = double(swapbytes(typecast(result(13:16),
'single')));
    dataTable.('rb_qx') = double(swapbytes(typecast(result(17:20),
'single')));
    dataTable.('rb_qy') = double(swapbytes(typecast(result(21:24),
'single')));
    dataTable.('rb_qz') = double(swapbytes(typecast(result(25:28),
'single')));
    dataTable.('rb_qw') = double(swapbytes(typecast(result(29:32),
'single')));
    dataTable.('rb_num') = swapbytes(typecast(result(33:36), 'single'));

end


end
```

1.5 create_object.m

```
function self = create_object

 stk = dbstack('-completenames');
 mname = stk(2).file;
 fcn_names = scan(mname, {'m_[A-Za-z0-9_]*'});
 fcns = evalin('caller',['@(){' sprintf('@%s ', fcn_names{:}) '}']);
 fcns = fcns();
 for i = 1:length(fcns);fcn = fcns{i};
   self.(fcn_names{i}) = fcn;
 end
end
```

1.6 FSR_initializ.m

```
% FSR_initialize

global ser_obj;
ser_obj=serial('COM1','baudrate',9600);
ser_obj.terminator = 'CR';
fopen(ser_obj);
```

1.7 FSR_close.m

```
% FSR_close


fclose(ser_obj);
delete(ser_obj)
clear ser_obj
```

1.8 Matt_GloveController_SensorCalc.m

```
function output = Matt_GloveController_SensorCalc(u)


x1=u(1);
y1=u(2);
x2=u(3);
y2=u(4);
x3=u(5);
y3=u(6);
x5=u(7);
y5=u(8);

L1=sqrt((y2-y1)^2+(x2-x1)^2);
L2=sqrt((y3-y2)^2+(x3-x2)^2);
```

```
theta1=(atan2((y2-y1),(x2-x1)));

% Define theta2
theo = (((atan2((y3-y2),(x3-x2))))-theta1);
theo1 = theo*180/pi;
if (theo1 <= -200)
    theta2deg=theo1+360;
else
    theta2deg = theo1;
end
theta2 = theta2deg/(180/pi);

% define theta3
the = (((atan2((y5-y3),(x5-x3))))-theta2-theta1);
% the1 = the*180/pi;
% if (the1 <= -200)
%     theta3deg=the1+360;
% else
%     theta3deg = the1;
% end
% theta3 = theta3deg/(180/pi);
theta3 = the;
output = [theta1;theta2;theta3;L1;L2];
```

1.9 FSR_glove_protocol.m

```
function output = FSR_glove_protocol(u)

%this function is written to define the action to be taken by the
robotic
%arm, aka to communicate the users intent by mesuring the force exerted
%on the manipulated object.
% u = [gl1 gl1 gl1 gl2 gl2 gl2 xgl1 ygl1 xgl2 ygl2 xdot_gl1 ydot_gl1
xdot_gl2 ydot_gl2 x3 y3 theta1 theta2 theta3 l1 l2]
%       1    2   3  4   5   6   7     8   9    10     11       12
13      14    15 16   17     18     19   20 21
action = 0;
force1 = 60;
force2 = 50;

Xdes = [0;0;0];
% first manipulate the 0-1023 values from the sensors to a %-of-100
RH_index_raw    = u(1);
RH_middle_raw   = u(2);
RH_thumb_raw    = u(3);

LH_index_raw    = u(4);
LH_middle_raw   = u(5);
LH_thumb_raw    = u(6);

% first manipulate the 0-1023 values from the sensors to a %-of-100
RH_index  = round((RH_index_raw/1023)*100);
RH_middle = round((RH_middle_raw/1023)*100);
RH_thumb  = round((RH_thumb_raw/1023)*100);
```

```
LH_index  = round((LH_index_raw/1023)*100);
LH_middle = round((LH_middle_raw/1023)*100);
LH_thumb  = round((LH_thumb_raw/1023)*100);

% Define the location of the users hands on the object as distances away
% from the end effector along the object.
x_eef = u(15);
y_eef = u(16);

x_g1 = u(7);
y_g1 = u(8);
x_g2 = u(9);
y_g2 = u(10);

L_RH = sqrt((x_g1 - x_eef)^2 + (y_g1 - y_eef)^2);
L_LH = sqrt((x_g2 - x_eef)^2 + (y_g2 - y_eef)^2);

% Define kinematic equation for arm
Theta = [u(17);u(18);u(19);u(20);u(21)];

R03 = [cos(Theta(1)+Theta(2)+Theta(3)) -sin(Theta(1)+Theta(2)+Theta(3))
0;
    sin(Theta(1)+Theta(2)+Theta(3)) cos(Theta(1)+Theta(2)+Theta(3)) 0;
    0 0 1];

R30 = transpose(R03);

% Define wheter the hand is moving along the object
rh_dot = [u(11);u(12);0];
lh_dot = [u(13);u(14);0];

RH_dot = R30*rh_dot;
LH_dot = R30*lh_dot;

% Define hand motions
% initalize seperate hand motions
RH_push = 0;
RH_pull = 0;
RH_sqze = 0;
RH_slde = 0;
RH_grip = 0;

LH_push = 0;
LH_pull = 0;
LH_sqze = 0;
LH_slde = 0;
LH_grip = 0;

% define hand motions
% ------Right Hand
-----------------------------------------------------
if (RH_thumb >= force1)
    RH_push = 1;
end
if ((RH_index >= force1) && (RH_middle >= force2))
    RH_pull = 1;
```

```
end
if ((RH_index >= force1) && (RH_middle >= force2)&&(RH_thumb >= force1))
    RH_sqze = 1;
    RH_push = 0;
    RH_pull = 0;
end
if (abs(RH_dot) > .9)
    RH_slde = 1;
    RH_sldeDirection = sign(RH_dot);
end
if ((RH_index >= force1)&&(RH_thumb >= force1)&&(RH_middle <= force2/2))
    RH_grip = 1;
elseif ((RH_index <= force1/2)&&(RH_thumb >= force1)&&(RH_middle >=
force2))
    RH_grip = -1;
end
% ------Left Hand
-----------------------------------------------------------
if (LH_thumb >= force1)
    LH_push = 1;
end
if ((LH_index >= force1) && (LH_middle >= force2))
    LH_pull = 1;
end
if ((LH_index >= force1) && (LH_middle >= force2)&&(LH_thumb >= force1))
    LH_sqze = 1;
    LH_push = 0;
    LH_pull = 0;
end
if (abs(LH_dot) > .9)
    LH_slde = 1;
    LH_sldeDirection = sign(LH_dot);
end
if ((LH_index >= force1)&&(LH_thumb >= force1)&&(LH_middle <= force2/2))
    LH_grip = 1;
elseif ((LH_index <= force1/2)&&(LH_thumb >= force1)&&(LH_middle >=
force2))
    LH_grip = -1;
end

% %
-------------------------------------------------------------------------
--
% % Then we can begin to define actions based on the user input.
% % first if the users rh is closer to the end effector.
if (L_LH > L_RH)
    % Action 1: Motion perpendicular to the object towards the user.
    if (RH_pull == 1)&&(LH_pull == 1)
        action = 1;
    end
%     Action 2: Motion Perpendicular to the object away from the user.
    if (RH_push == 1) && (LH_push == 1)
        action = 2;
    end
    % Action 3: CW Rotation about the user's closer hand (RH Closer)
    if (RH_sqze == 1) && (LH_push == 1)
        action = 3;
```

```
    end
    % Action 4: CCW Rotation about the user's closer hand(RH Closer)
    if (RH_sqze == 1)&&(LH_pull == 1)
    action = 4;
    end
    %Action 5: Motion parallel to the object with RH closer.
    if (RH_sqze == 1) && (LH_grip > 0)
        action = 5;
    elseif (RH_sqze) && (LH_grip < 0)
        action = 6;
    end
end
%
% % % users left hand is closer than the users right hand
if (L_RH > L_LH)
%     Action 7: Motion perpendicular to the object towards the user.
    if (RH_pull) && (LH_pull)
        action = 7;
    end
%     Action 8: Motion Perpendicular to the object away from the user.
    if (RH_push == 1) && (LH_push == 1)
        action = 8;
    end
%     Action 9: CW Rotation about the user's closer hand (LH Closer)
    if (LH_sqze == 1) && (RH_pull == 1)
        action = 9;
    end
%     Action 10: CCW Rotation about the user's closer hand (LH Closer)
    if (LH_sqze == 1) && (RH_push == 1)
        action = 10;
    end
%     Action 11: motion parallel to the object with left hand closer
    if (LH_sqze == 1) && (RH_grip > 0)
        action  = 11;
    elseif (LH_sqze == 1) && (RH_grip < 0)
        action = 12;
    end
end
%
Output-----------------------------------------------------------------
% Define the desired object frame vector based on the action (some
actions
% yield the same vector)
% Vector in the positive y direction: actions 1 & 8
if (action == 1)||(action == 8)
    Xdes = [0;.15;0];
end
% Vector in the negative y direction: actions 2 & 7
if (action == 2)||(action == 7)
    Xdes = [0; -.15;0];
end
% Vector in the positive x direction: action 6 & 12
if (action == 6)||(action == 12)
    Xdes = [.15;0;0];
end
% Vector in the negative x direction: action 11 & 5
if (action == 5)||(action == 11)
```

```
    Xdes = [-.15;0;0];
end
% Vector to induce CCW rotation: action 4 & 10
if (action == 4)||(action == 10)
    Xdes = [0; -0.15;1.8];
end
% vector to induce CW rotation: action 9 & 3
if (action == 3)||(action == 9)
    Xdes = [0; 0.15;-1.8];
end
if (action == 0)
    Xdes = [0;0;0];
end

output = [action;Xdes];
% output =
[action;RH_index;RH_middle;RH_thumb;LH_index;LH_middle;LH_thumb;Xdes];
```

1.10 FSR_glove_DTConnect.m

```
function self=FSRGloveDTConnect(comID, rbnbIP,port, remoteXBAddr)

%,factor)


%old RobotDTConnection file, used with
%c:\SCREEM\SerialPortTurbine\SerialPortTurbine.jar connection file
global pwmL_prev;
global pwmR_prev;
self = create_object;

    import com.rbnb.sapi.*;
    name=comID;

    source=Source(1,'none',0);
    source.OpenRBNBConnection(strcat(rbnbIP,':',port),strcat('ClusterCo
ntrollerSource-',comID));
    cmap=ChannelMap;
    cmap.Add(comID);

    dataTable = hashtable;

    m_received = true;
    dataTable.('x1')=0;
    dataTable.('y1')=0;
    dataTable.('z1')=0;
    dataTable.('x2')=0;
    dataTable.('y2')=0;
    dataTable.('z2')=0;
    dataTable.('x3')=0;
    dataTable.('y3')=0;
    dataTable.('z3')=0;
    dataTable.('zero')=0.0;

    source.Register(cmap);
```

```
    sink=Sink;
    sink.OpenRBNBConnection(strcat(rbnbIP,':',port),
strcat('ClusterControllerSink-',comID));
    rmap=ChannelMap;

    rmap.Add(strcat('CSCADE_source','/',comID));

   sink.Monitor(rmap,0);

    function m_stop()
        %m_bStopped=true;
        dataTable.('Disabled')=true;
        m_send([0 0]);
    end

    function m_go()
        dataTable.('Disabled')=false;
        %m_bStopped=false;
    end

    % m_getName
    %
    % @returns robot name string
    %%
    function [comID]=m_getName()
       comID=name;
    end

    % m_disconnect
    %
    % Disconnects the robot from the Data Tubine
    %%
    function m_disconnect()
        sink.CloseRBNBConnection();
        source.CloseRBNBConnection();
    end

    % m_getTelemetry
    %
    % @returns telemetry:hashtable containing the latest telemetry
    %%
    function [telemetry]=m_getTelemetry(keys)
       telemetry=repmat(double(0),1,length(keys));
       for i=1:length(keys)
           telemetry(i)=double(dataTable.(keys{i}));
       end
    end

    %verifying if sink is connected crashes the data turbine for some
    %reason...
    function [result]=m_isConnected()
        result=source.VerifyConnection();
    end

    % m_send
```

```
%
% Sends velocity commands to the robot
%
% @param inputVector:1x2 vector [forwardVelocity rotVelocity]
%%
function m_send(u)

    vTrans = u(1);
    vRot = u(2);    %rot left is negative!!

    velL = floor(127*u(1)/10)-floor(127*u(2)/10);
    velR = floor(127*u(1)/10)+floor(127*u(2)/10);

    pwmL = velL + 127;
    pwmR = 127 - velR;
    strL = strcat({'sv0 '},num2str(pwmL),'\n');
    strR = strcat({'sv1 '},num2str(pwmR),'\n');

    charL = char(strL);
    charR = char(strR);

    if pwmL ~= pwmL_prev        %leftPWM has been updated
        if pwmR == pwmR_prev    %rightPWM stayed the same
            %update L motor only
            pwmL_prev = pwmL;
            send_char('gb\n');
            pause(0.08);
            send_char(charR);
            pause(0.04);
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.08);
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.4);
        else
            %update both motors
            pwmL_prev = pwmL;
            pwmR_prev = pwmR;
            send_char('gb\n');
            pause(0.08);
            send_char(charL);
            pause(0.08);
            send_char(charR);
            pause(0.04);
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.08);
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.4)
        end
    elseif pwmL == pwmL_prev
        if pwmR == pwmR_prev
            %both motors same as before, update telemetry only
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.08);
            send_char('tc 190 250 200 250 0 50\n');
            pause(0.4);
        else
            %update R motor only
```

```
                pwmR_prev = pwmR;
                send_char('gb\n');
                pause(0.08);
                send_char(charR);
                pause(0.04);
                send_char('tc 190 250 200 250 0 50\n');
                pause(0.08);
                send_char('tc 190 250 200 250 0 50\n');
                pause(0.4);
            end
        end

        pwmL_prev = pwmL;
        pwmR_prev = pwmR;
    end

    function m_send_bBot(u)
        vTrans = u(1);
        vRot = u(2);    %rot left is negative!!

        velL = floor(127*u(1)/10)-floor(127*u(2)/10);
        velR = floor(127*u(1)/10)+floor(127*u(2)/10);

        pwmL = sprintf('%.2x',velL + 127);
        pwmR = sprintf('%.2x',127 - velR);

        strBBot = strcat('00070100',remoteXBAddr,'01',pwmL,pwmR);
%         selBBot =
strcat('0010170000000000000000000',remoteXBAddr,'02443105');
%         send_byte_raw([selBBot]);
%         pause(0.02);
%         send_byte_raw([selBBot]);
        pause(0.02);
        send_byte_raw([strBBot]);
        pause(0.02);
        send_byte_raw([strBBot]);

    end

    function m_send_char(input_str)
        send_char(input_str);
    end

    function send_char(input_str)
       if(dataTable.('Disabled'))
           inputVector = [0 0];
       end
       return_char = '';

       input_str = lower(input_str);
       str_len = length(input_str);

       if (input_str(str_len-1) == '\') && (input_str(str_len) == 'n')
           return_char = '0d';
           input_str = input_str(1:str_len-2);
       end
```

```
    inputVector = [double(input_str) hex_to_int8(return_char)];
    cmap.PutTimeAuto('timeofday');
    packet = packetAssembler(inputVector);

    for i=1:length(packet)
     cmap.PutDataAsByteArray(0, packet(i));
    end

    source.Flush(cmap,true);

end

%send everything except frame delimiter and checksum
%string representation of hex
function m_send_byte_raw(inputVector)
    if(dataTable.('Disabled'))
        inputVector = [0 0 0 0];
    end
    cmap.PutTimeAuto('timeofday');

    inputHexVector=hex_to_int8(inputVector);
    for i=1:length(inputHexVector)
     cmap.PutDataAsByteArray(0, int8(inputHexVector(i)));
    end
     source.Flush(cmap,true);
end

function send_byte_raw(inputVector)
    if(dataTable.('Disabled'))
        inputVector = [0 0 0 0];
    end
    cmap.PutTimeAuto('timeofday');

    inputHexVector=hex_to_int8(inputVector);
    for i=1:length(inputHexVector)
     cmap.PutDataAsByteArray(0, int8(inputHexVector(i)));
    end
     source.Flush(cmap,true);
end

function frameID=set_frameID()
     frameID=1;
end

function pLen = get_packetlength(vector)
     pLen = length(vector);
end

function packet = packetAssembler(inputVector)
     % packet should have the format:
     % 7e msb lsb apiID frameID data checksum
     % 0x7E: start deliminator
     % MSB: most significant bit
     % LSB: least significant bit
     % apiID: API command identifier
     % data: desired data to be sent
     % checksum: checksum of packet -start deliminator and length are
```

```
        %   not included in the calculation
        apiID=1;
        ATCmd=1;
        data=[apiID set_frameID() hex_to_int8(remoteXBAddr) ATCmd
inputVector];
        length=get_packetlength(data);
        msb=bitshift(length, -8);
        lsb=bitand(255, length);
        packet=[msb lsb data];

    end

    function checksum = get_checksum(dataArray)
        checksum=-1-sum(dataArray);
    end

    function m_sync_temp()

        testMap = sink.Fetch(1);
        %testMap=sink.Fetch(-1);
          if(testMap.NumberOfChannels < 1)
                data.received=false;
            return;
          end

          result=testMap.GetData(0);
          testMap.Clear();
          result=int8_to_hex(result);
          result=hex2dec(result);
          [x y z e readXB]=parse_adc(result);
          %display('Read')
          if(e==0)
              if(readXB==1)
                dataTable.('x1')=x;
                dataTable.('y1')=y;
                dataTable.('z1')=z;
              elseif(readXB==2)
                dataTable.('x2')=x;
                dataTable.('y2')=y;
                dataTable.('z2')=z;
              elseif(readXB==3)
                dataTable.('x3')=x;
                dataTable.('y3')=y;
                dataTable.('z3')=z;
              else
                  display('Unknown Xbee')
              end
          elseif(e==2)
%               display('Two strings')
              result1=result(1:18);
              result2=result(19:36);
              [x y z e readXB]=parse_adc(result1);
              if(readXB==1)
                  dataTable.('x1')=x;
                  dataTable.('y1')=y;
                  dataTable.('z1')=z;
              elseif(readXB==2)
```

```
                    dataTable.('x2')=x;
                    dataTable.('y2')=y;
                    dataTable.('z2')=z;
                elseif(readXB==3)
                    dataTable.('x3')=x;
                    dataTable.('y3')=y;
                    dataTable.('z3')=z;
                else
                    display('Unknown Xbee')
                end
                [x y z e readXB]=parse_adc(result2);
                if(readXB==1)
                    dataTable.('x1')=x;
                    dataTable.('y1')=y;
                    dataTable.('z1')=z;
                elseif(readXB==2)
                    dataTable.('x2')=x;
                    dataTable.('y2')=y;
                    dataTable.('z2')=z;
                elseif(readXB==3)
                    dataTable.('x3')=x;
                    dataTable.('y3')=y;
                    dataTable.('z3')=z;
                else
                    display('Unknown Xbee')
                end
                clear result1 result2
            end
%           sync_temp_vector=int8_to_hex(result);
%           [x y] = parse_xy(sync_temp_vector,remoteXBAddr);
%           dataTable.('x') = x;
%           dataTable.('y') = y;
%           disp(result)
    end

end %object
```

1.11 parse_adc.m

```
function[a1 a2 a3 e readXB]=parse_adc(xb_str_int)

Length_str=length(xb_str_int);
global er
global ir
global lr
global sr
ind_tld = find(xb_str_int == 126);      %0x7E = '~'
ind_adc = find(xb_str_int == 131);       %0x83 = special char
a1=0;
a2=0;
a3=0;
e=1;
readXB=0;
if(Length_str~=18)
```

```
    if(Length_str<18)
        sr=sr+1;
        %display('Short String')
        er=er+1;
        return;
    elseif(Length_str>18)
        lr=lr+1;
        %display('Long String')
        if(Length_str==18*2)
        e=2;
        end
        return;
    end
end
rdXB=xb_str_int(5:6);

readXB=rdXB(1)*256+rdXB(2);
%display(readXB)
if isempty(ind_adc)
    er=er+1;
    %display('ERROR Empty')
    ind_adc = 0;
    return;
elseif(Length_str==18 && ind_tld(1)==1 && ind_adc(1)==4)
    ir=ir+1;
    e=0;
    a1d=xb_str_int(12:13);
    a2d=xb_str_int(14:15);
    a3d=xb_str_int(16:17);

    a1=a1d(1)*256+a1d(2);
    a2=a2d(1)*256+a2d(2);
    a3=a3d(1)*256+a3d(2);
    %display('Data Read');
    return;
else
    er=er+1;
    %display('ERROR Else')
    return;
end


end
```

1.12 Matt_GloveController_Jinv.m

```
function output = Matt_GloveController_Jinv(u)

%input : u = [theta1 theta2 theta3 l1 l2 dx dy w]
t1 = u(1);
t2 = u(2);
t3 = u(3);
l1 = u(4);
l2 = u(5);
l3 = 0;
```

```
s1 = sin(t1);
c1 = cos(t1);
s2 = sin(t2);
c2 = cos(t2);
s3 = sin(t3);
c3 = cos(t3);
s12  = sin(t1+t2);
s123 = sin(t1+t2+t3);
c12  = cos(t1+t2);
c123 = cos(t1+t2+t3);

J11 = (-l1*s1) - (l2*s12)-(l3*s123);
J12 = (-l2*s12) - (l3*s123);
J13 = (-l3*s123);
J21 = (l1*c1)+(l2*c12)+(l3*c123);
J22 = (l2*c12)+(l3*c123);
J23 = (l3*c123);

Jbase=[J11 J12 J13;J21 J22 J23;1 1 1];

Jinv = [c12/l1/(-c12*s1+s12*c1) s12/l1/(-c12*s1+s12*c1) -l3*(c123*s12-
s123*c12)/l1/(-c12*s1+s12*c1);
    -(l1*c1+l2*c12)/l2/l1/(-c12*s1+s12*c1) -(l1*s1+l2*s12)/l2/l1/(-
c12*s1+s12*c1) -l3*(-c123*l1*s1-
c123*l2*s12+s123*l1*c1+s123*l2*c12)/l2/l1/(-c12*s1+s12*c1);
    c1/l2/(-c12*s1+s12*c1) s1/l2/(-c12*s1+s12*c1) (-l2*c12*s1-
l3*c123*s1+l2*s12*c1+l3*s123*c1)/l2/(-c12*s1+s12*c1)];

output = Jinv;
```

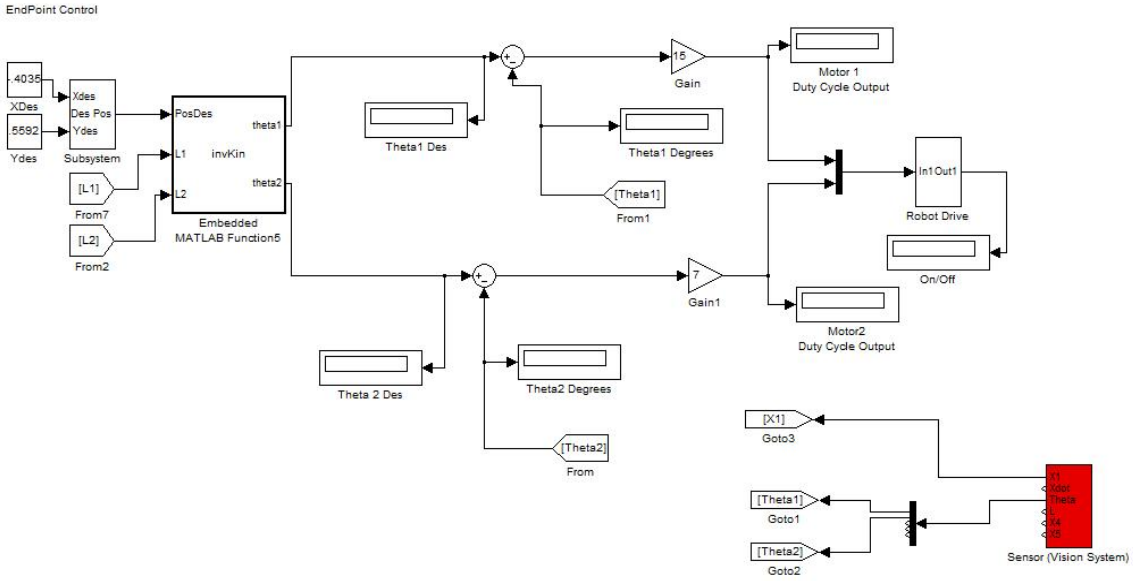
1.13 ThesisPlot.m

```
% Script plots overhead view with history


axis('square'); axis([-0.35 2 -1 1]); axis manual; hold on;

% t=dataOut(1,:);
% for z=1:1:length(t)
% plot(dataOut(2,z), dataOut(3,z), 'o')
% plot(dataOut(4,z), dataOut(5,z), 'o')
% plot(dataOut(6,z), dataOut(7,z), 'x')
% plot([0;dataOut(2,z);dataOut(4,z)],[0;dataOut(3,z);dataOut(5,z)])
% plot([0;dataOut(4,z);dataOut(6,z)],[0;dataOut(5,z);dataOut(7,z)])
%
% plot(dataOut(8,z), dataOut(9,z), 'or')
% plot(dataOut(10,z), dataOut(11,z), 'og')
% % pause
% end

t=dataOut(:,1);
for z=1:1:length(t)
```

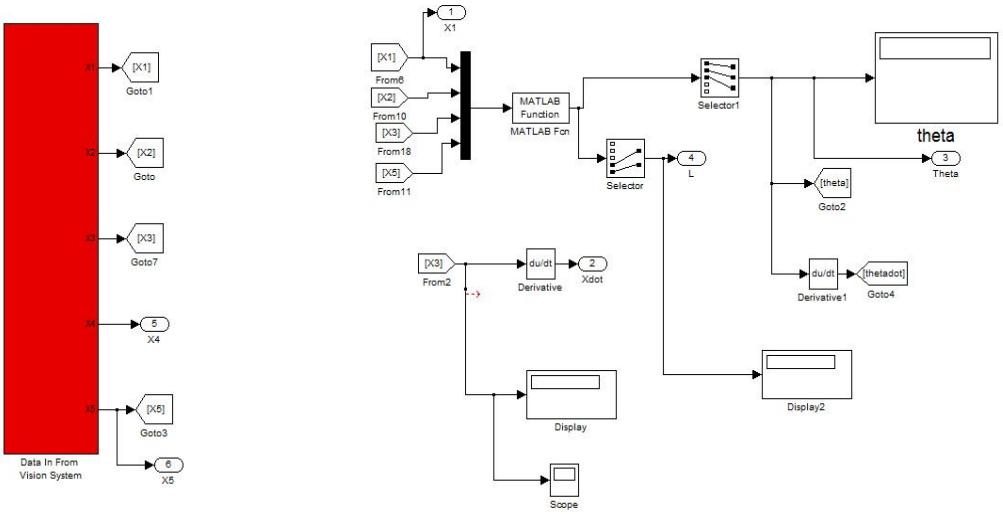
```
plot(dataOut(z,2), dataOut(z,3), 'o')
plot(dataOut(z,4), dataOut(z,5), 'o')
plot(dataOut(z,6), dataOut(z,7), 'x')
plot([0;dataOut(z,2);dataOut(z,4)],[0;dataOut(z,3);dataOut(z,5)])
plot([0;dataOut(z,4);dataOut(z,6)],[0;dataOut(z,5);dataOut(z,7)])

plot(dataOut(z,8), dataOut(z,9), 'or')
plot(dataOut(z,10), dataOut(z,11), 'og')
pause
end
```

Appendix C: Simulink Models

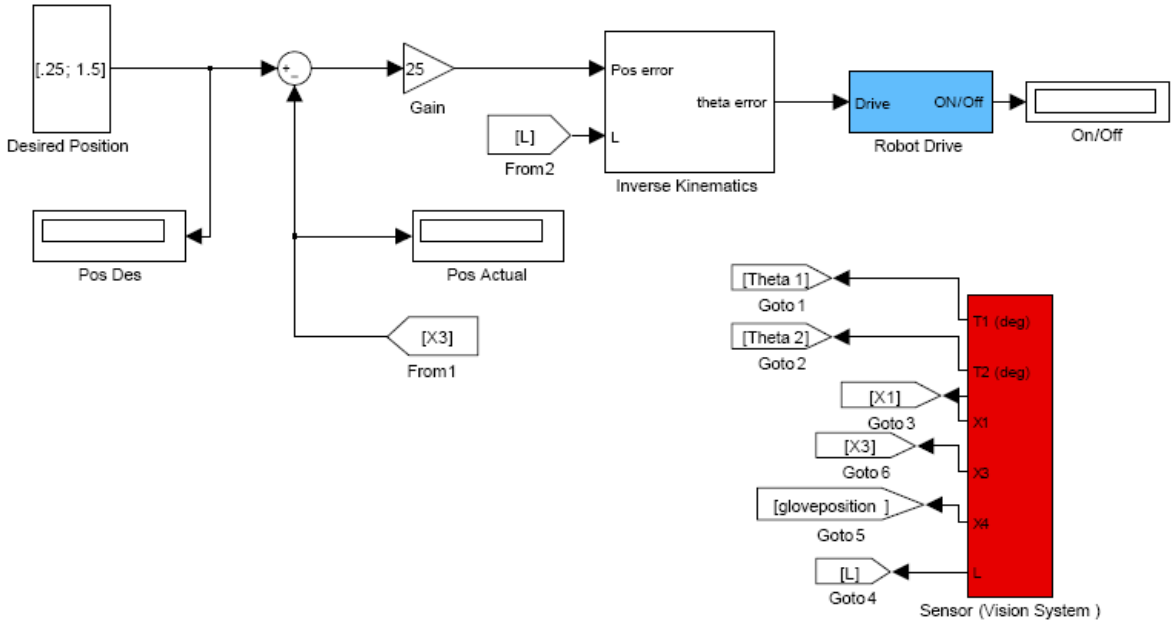
1.14 Cartesian Space Specified Joint Space Controller Controller



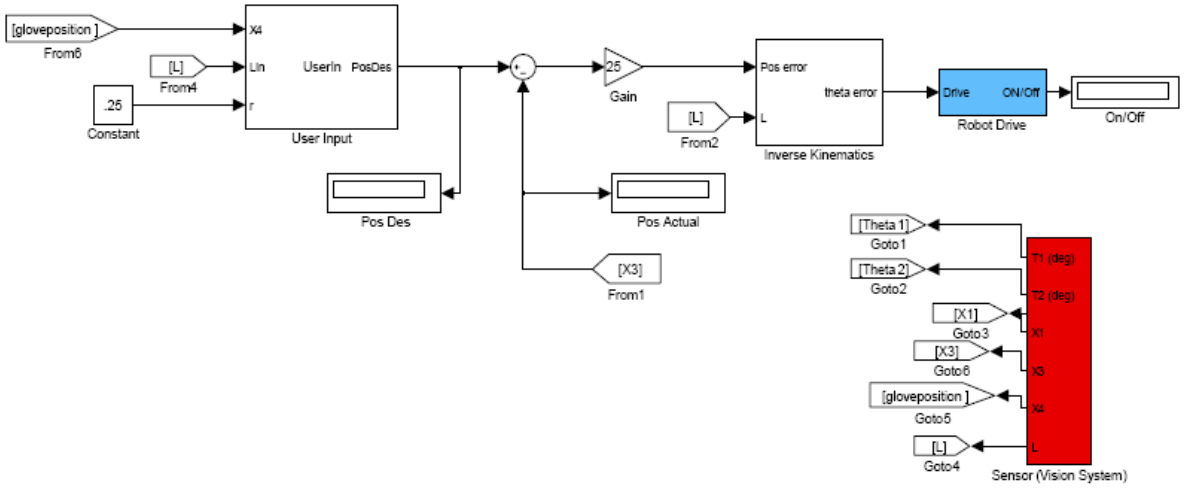
1.15 Cartesian Space Specified Joint Space Controller Sensor



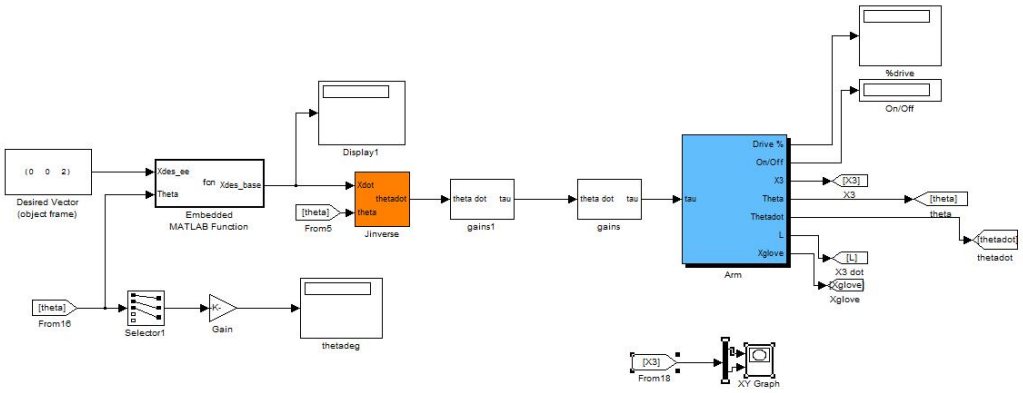
1.16 Cartesian Space Controller



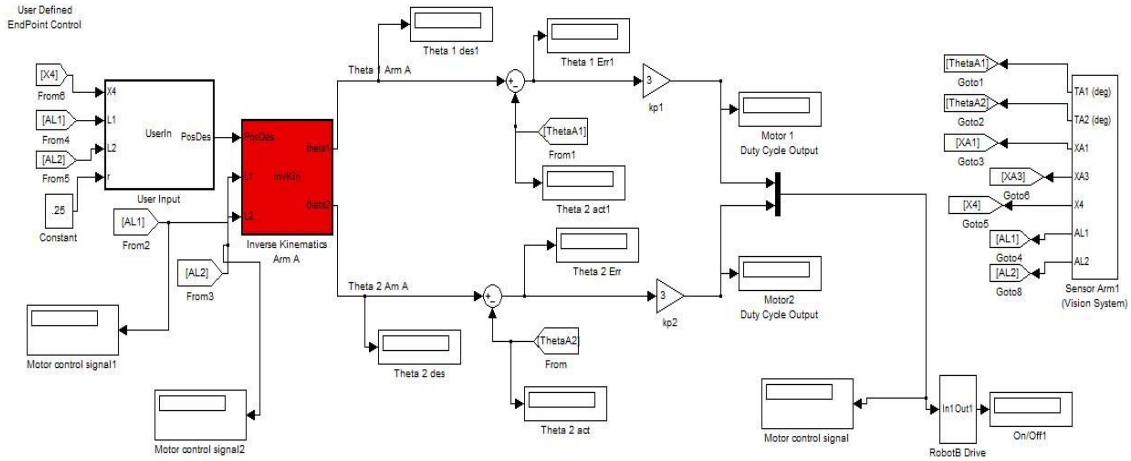
1.17 Obstacle Avoidance Controller



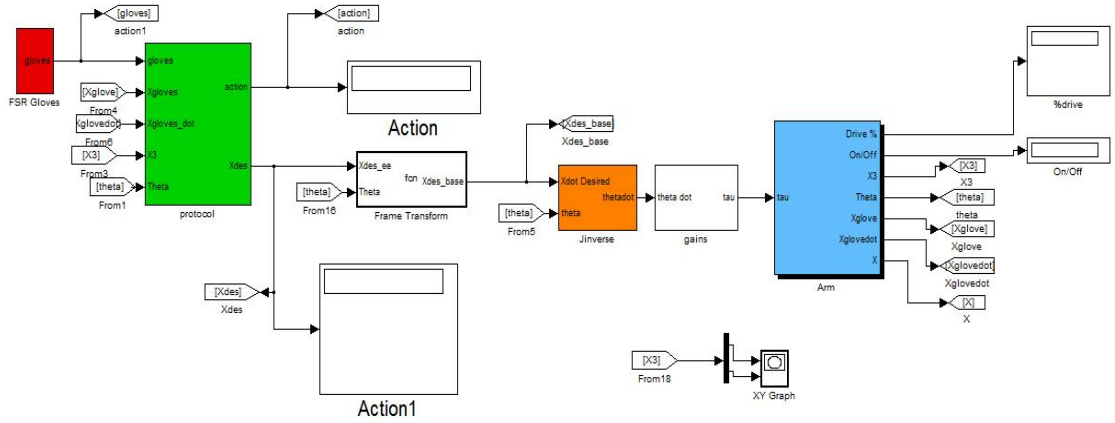
1.18 Jacobian Inverse Controller



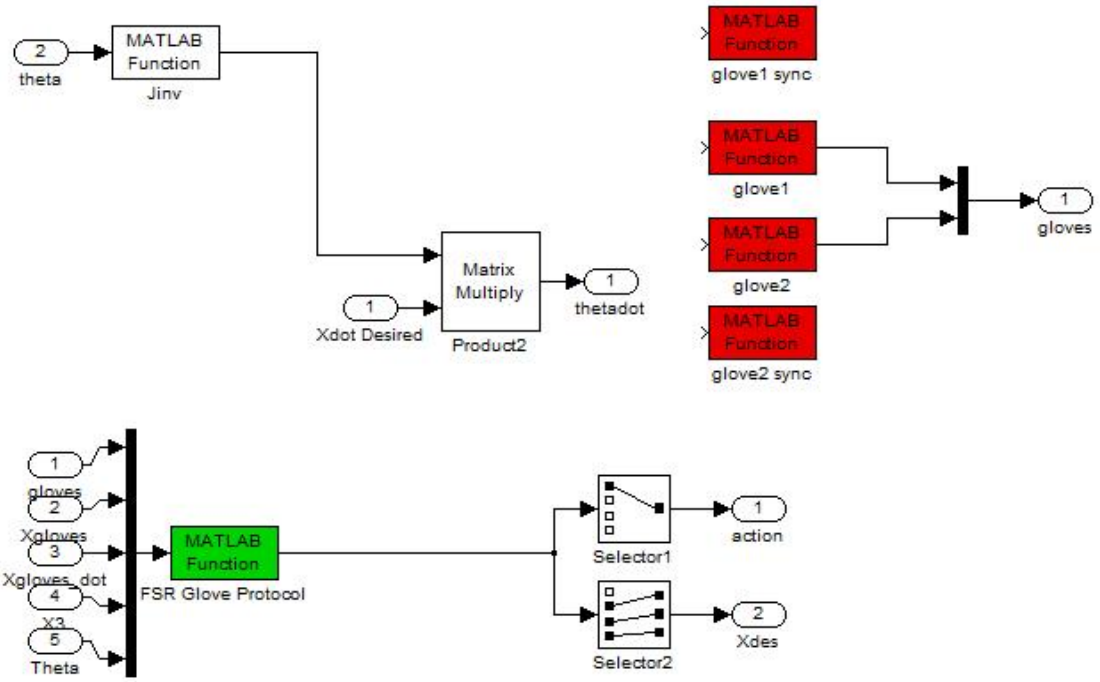
1.19 Obstacle Avoidance Controller



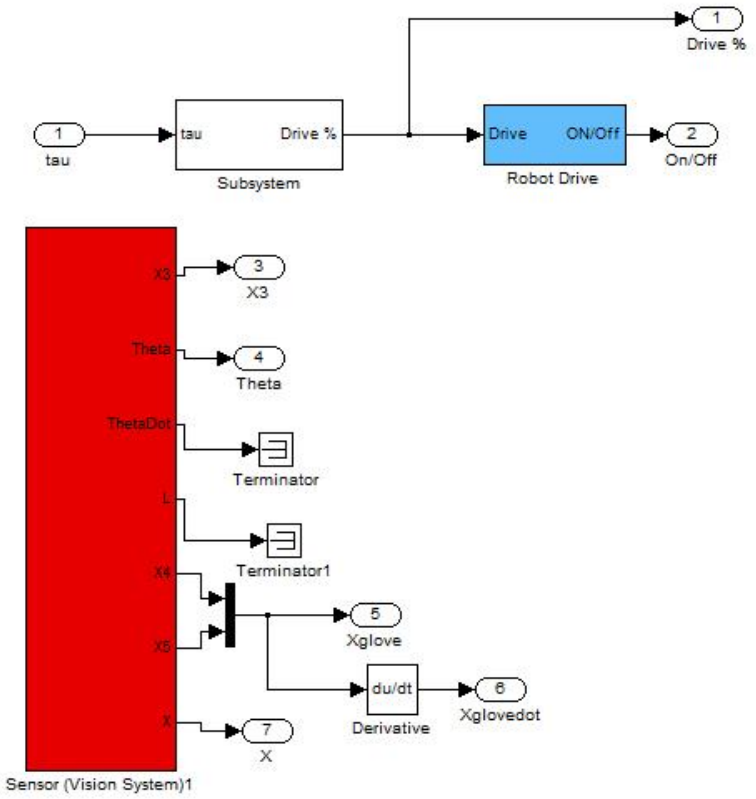
1.20 Jacobian Controller (User Input)



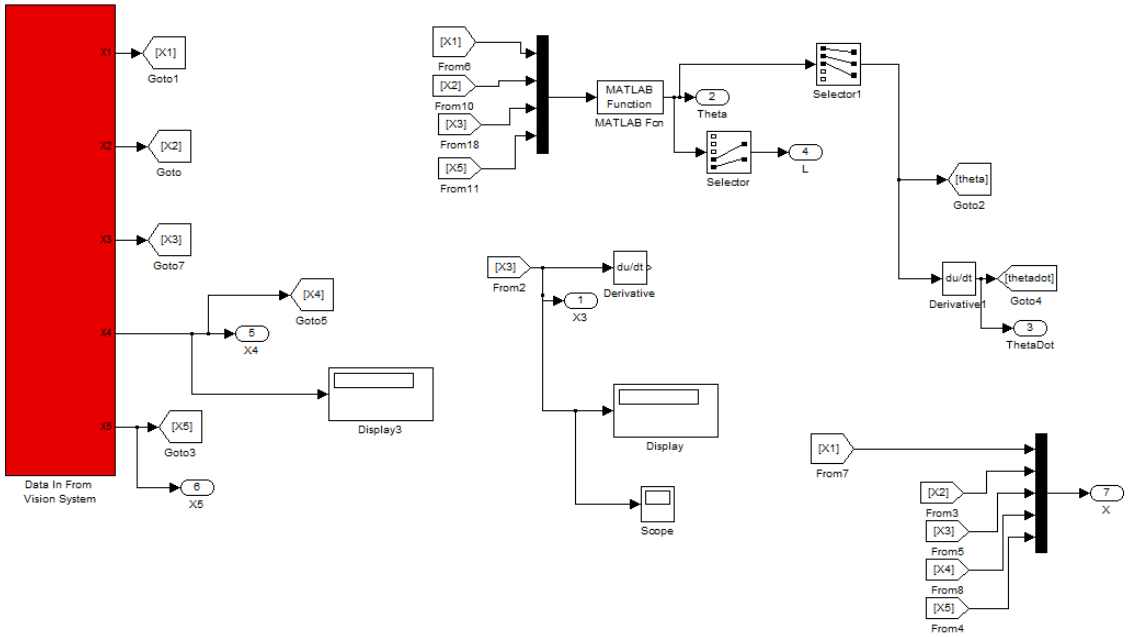
1.21 Jacobian Controller (User Input) Subsystems



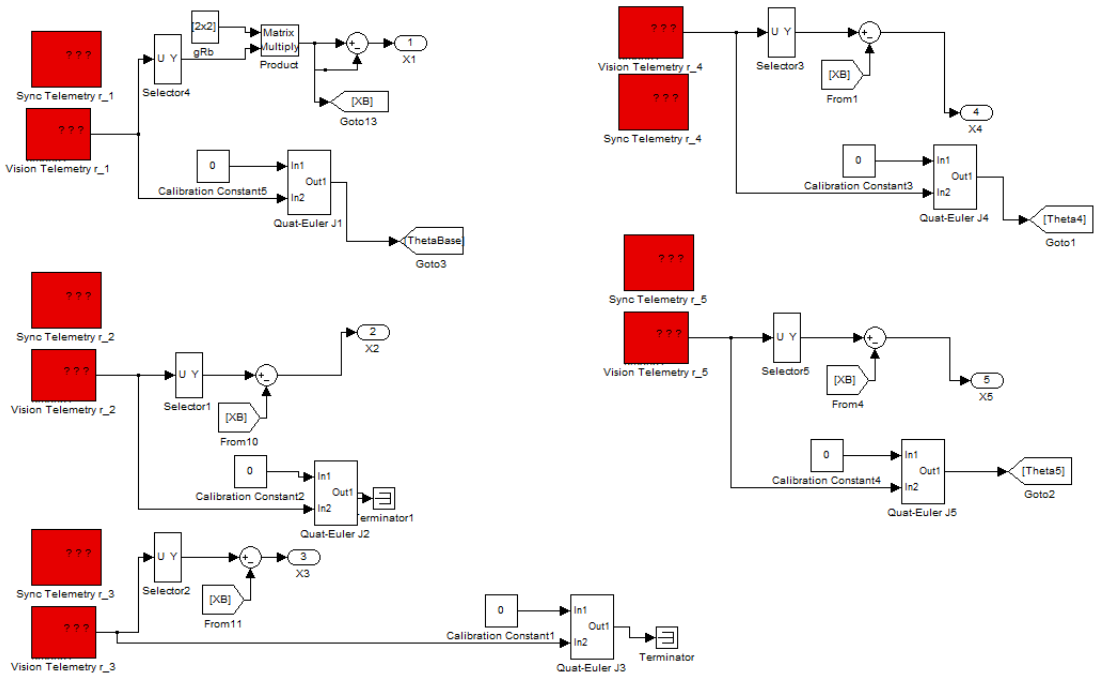
1.22 Jacobian Controller (User Input) Robot Block



1.23 Diagram For External Vision System Sensor



1.24 Diagram of Vision System in Matlab



Appendix D:

User Test Data: Template

This test is designed to measure the difficulty of three moves and a standard insertion task. The moves are outlined below, and the insertion task is detailed by the research team and documented photographically.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the glove interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

9:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

10: When Using the interface, did you feel in control?
Joystick:
Yes No
Gloves:
Yes No

11: Was the action-reaction protocol clear?
For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:
Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5

Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:

Comments:

User Test Data:

The goal of this project is to develop a more natural and organic user interface for robot assisted manipulation of the object, specifically allowing the user to communicate their intent through the object being manipulated. In order to validate this claim, we are attempting to get user feedback on the effeminacy of using the FSR gloves versus using the joystick protocol. The Scenario the user is intended to recreate is, as follows.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the gloves interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: When Using the interface, did you feel in control?

Joystick:

Yes No

Gloves:

Yes No

9: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

10:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

11: Was the action-reaction protocol clear?

For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:

Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:
2
Comments:

Easier to work with. More interactive. Gloves are natural haptic feedback so faster.

User Test Data:

The goal of this project is to develop a more natural and organic user interface for robot assisted manipulation of the object, specifically allowing the user to communicate their intent through the object being manipulated. In order to validate this claim, we are attempting to get user feedback on the effeminacy of using the FSR gloves versus using the joystick protocol. The Scenario the user is intended to recreate is, as follows.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the glove interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

9:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

10: When Using the interface, did you feel in control?

Joystick:

Yes No

Gloves:

Yes No

11: Was the action-reaction protocol clear?

For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:

Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:
2-3

Comments:

Moving the object using the gloves was a lot more fluid then with the joystick. When I used the joystick, I found it easement to move in single directions at a time, and had to stop, evaluate position, then adjust my movement iteratively in order to reach the target. With the gloves, I was able to insert the object into the target using complex motions that only required two moves.

User Test Data:
This test is designed to measure the difficulty of three moves and a standard insertion task. The moves are outlined below, and the insertion task is detailed by the research team and documented photographically.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the gloves interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

9:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

10: When Using the interface, did you feel in control?

Joystick:

Yes No

Gloves:

Yes No

11: Was the action-reaction protocol clear?

For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:

Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5

Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:
4 glove size and sensor dynamics

Comments:

Gloves are not one size fits all. Difficulties triggering the sensors when gloves don't fit properly.

Easier to move using gloves than joystick given that the sensors in the gloves are triggered properly.

User Test Data:

The goal of this project is to develop a more natural and organic user interface for robot assisted manipulation of the object, specifically allowing the user to communicate their intent through the object being manipulated. In order to validate this claim, we are attempting to get user feedback on the effeminacy of using the FSR gloves versus using the joystick protocol. The Scenario the user is intended to recreate is, as follows.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the gloves interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

9:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

10: When Using the interface, did you feel in control?

Joystick:

Yes No

Gloves:

Yes No

11: Was the action-reaction protocol clear?

For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:

Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:
2
Comments:
Very Cool

User Test Data:

The goal of this project is to develop a more natural and organic user interface for robot assisted manipulation of the object, specifically allowing the user to communicate their intent through the object being manipulated. In order to validate this claim, we are attempting to get user feedback on the effeminacy of using the FSR gloves versus using the joystick protocol. The Scenario the user is intended to recreate is, as follows.

Starting Position:

Theta1: 0 Theta2: 90 Theta3: -90

Move 1:

Translation in the negative Y Direction until end effector Yeef = 0

Move2:

pure rotation about the object center, rotation 90

Move 3:

translation in the negative y direction until Yeef = -0.5

Team Member will demostrate the glove protocol and the joystick protocol, give the tester adequate time to familiarize themselves with each of the test protocols. The tester can attempt the moves several times with each human interface device, then provide feedback for the motion.

Feedback:

1: On a scale of 1 – 10, how familiar are you with robotic systems, one being not familiar, ten begin very familiar?

1 2 3 4 5 6 7 8 9 10

2: How easy was it to familiarize yourself with the joystick interface, five being difficult and one being easy?

1 2 3 4 5

3: How easy was it to familiarize yourself with the gloves interface, five being difficult and one being easy?

1 2 3 4 5

4: Did you feel that you were given adequate time to familiarize yourself with the two interface systems?

Yes No

5: Which interface did you feel more comfortable with, if either?

Joystick Gloves Neither

6: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the joystick.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

7: Rate each Move on a scale of 1 to 5, five being difficult one being easy, when performed with the glove interface.

Move 1:	1	2	3	4	5
Move 2:	1	2	3	4	5
Move 3:	1	2	3	4	5

8: Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the joystick interface.

1 2 3 4 5

9:Rate the insertion task on on a scale of 1 to 5, five being difficult and one being easy, when performed with the glove interface.

1 2 3 4 5

10: When Using the interface, did you feel in control?

Joystick:

Yes No

Gloves:

Yes No

11: Was the action-reaction protocol clear?

For the Joystick: Yes No

For the Gloves: Yes No

12: On a scale of one to five, rate the following features of each interface system, one being untrue,5 being true:

Joystick:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

Gloves:

Useable	1	2	3	4	5
Repeatable	1	2	3	4	5
Intuitive	1	2	3	4	5
Natural	1	2	3	4	5
Efficient	1	2	3	4	5
Accurate	1	2	3	4	5

13: Cooper Harper Scale:
2
Comments:

I liked the user interface, however the translational move was not very intuitive. Another version of that move would be better and probably more efficient. But overall a cool interface.