Final Report on Telepresence for the Visually Impaired

Submitted To

Ryan Field
Dr. Christine Julien
UT Pharos Labs

Prepared By

Razik Ahmed
Thomas Brezinski
Stephen Hall
Zachary Lalanne
Ali Unwala

EE464 Senior Design Project
Electrical and Computer Engineering Department
University of Texas at Austin

Spring 2012
CONTENTS

FIGURES..............................................................................................................................iv
EXECUTIVE SUMMARY .....................................................................................................v

1.0 INTRODUCTION ....................................................................................................... 1

2.0 DESIGN PROBLEM STATEMENT ......................................................................... 2
  2.1 SPECIFICATIONS FOR SYSTEM ........................................................................ 2
  2.2 DESIGN PARAMETERS .................................................................................. 2

3.0 DESIGN PROBLEM SOLUTION ............................................................................. 3
  3.1 DESCRIPTION OF DESIGN SOLUTION ....................................................... 3
    3.1.1 Dynamic Map Solution .............................................................................. 4
    3.1.2 Static Map Solution ................................................................................... 4
  3.2 DESIGN DECISIONS ........................................................................................ 5
    3.2.1 Final Design Decisions ............................................................................... 5
    3.2.2 Alternate Designs ....................................................................................... 7
  3.3 SPECIFICATIONS MET .................................................................................. 8
  3.4 ECONOMIC ANALYSIS .................................................................................. 9

4.0 DESIGN IMPLEMENTATION ................................................................................. 9
  4.1 ROBOT DESIGN ............................................................................................. 10
  4.2 USER INTERFACE DESIGN ......................................................................... 11
    4.2.1 Inputs and Outputs .................................................................................. 11
    4.2.2 System Control ......................................................................................... 12
    4.2.3 Receiving Information ............................................................................. 12
    4.2.4 Remote Communication ........................................................................... 13
CONTENTS (Continued)

5.0 TEST AND EVALUATION ........................................................................................................... 13
5.1 COMPONENT LEVEL TESTING ............................................................................................ 13
  5.1.1 Tactile Feedback Matrix .................................................................................................. 13
  5.1.2 Joystick ............................................................................................................................ 14
  5.1.3 Voice Commands ............................................................................................................. 14
5.2 MODULE LEVEL TESTING .................................................................................................... 15
  5.2.1 Robot Module .................................................................................................................. 15
  5.2.2 User Interface Module ...................................................................................................... 16
5.3 SYSTEM LEVEL TESTING .................................................................................................... 16
6.0 TIME AND COST CONSIDERATIONS ...................................................................................... 18
7.0 SAFETY AND ETHICAL ASPECTS OF DESIGN ....................................................................... 19
8.0 RECOMMENDATIONS ........................................................................................................... 19
  8.1 RECOMMENDATIONS FOR FUTURE DEVELOPMENT ...................................................... 19
  8.2 RECOMMENDATIONS FOR FUTURE GROUPS ................................................................. 20
9.0 CONCLUSIONS ...................................................................................................................... 21
REFERENCES .................................................................................................................................. 23
APPENDIX A – BILL OF MATERIALS ............................................................................................. A-1
APPENDIX B – GANTT CHART .....................................................................................................B-1
FIGURES

1  Input / Output Diagram ............................................................................................................. 10
2  Robot Design .......................................................................................................................... 11
3  Test Environment .................................................................................................................... 17
EXECUTIVE SUMMARY

Our project aims to address a gap in current telepresence technology. Today’s systems do not provide support for a blind user. We attempted to create a telepresence platform that replaces the standard vision-based navigation system with a touch-based system. Our touch-based navigation system uses a tactile feedback matrix to display a map of nearby obstacles allowing the user to feel the surroundings of the telepresence platform.

The final decisions that we made included
- a Turtlebot design for the robot,
- a remote computer controlling the robot using a joystick,
- having the maps be displayed on the tactile matrix,
- a 4x4 tactile matrix made from servos and dowels, and
- both subsystems running ROS.

Our full system includes a robot and a user interface (UI). The robot deals with localizing itself in the environment and collecting data on obstacles around itself. It does this by using an iRobot Create base, a Microsoft Kinect, and a Dell mini 9 netbook. The UI’s two main components are the dowel matrix and the joystick. This matrix is a set of 16 wooden dowels put into a 4x4 matrix formation that are able to move up and down out of a plane. The goal of the feedback matrix is to create a real-time bird's-eye-view map of obstacles around the robot, using dowels to represent obstacles. Using this matrix and a simple joystick a blind user can navigate a robot around an room. Both these systems work together to create a real-time map that constantly updates and allows for fast robot navigation. The algorithm it uses is called SLAM which stands for Simultaneous Localization and Mapping. Once we made an accurate map, we then switched to the Adaptive Monte Carlo Localization (AMCL) algorithm which the robot used to place itself in that map. As a complete system our robot is able to create maps and return that data to the tactile matrix.

As we were working on the project, we needed to test specific parts of the user interface and robot. For the user interface, we had to test the tactile feedback matrix, joystick and voice commands. For the robot, we had to test that the software was correctly interfacing with the Microsoft Kinect to localize the robot properly within its environment. After the robot was mapping properly, we built our test environment that would allow a user to effectively navigate to a target, testing the entire system.

For most of the tested components, we were able to work around the problems and get to a working compromise or solution. Throughout our testing, we noticed the most issues with the iRobot Create base platform. Because we did not build the base, it was difficult to fix the problems and we had issues with the basic level sensors, for which we had no alternative.
1.0 INTRODUCTION

The purpose of this paper is to document the telepresence system for the visually impaired that was designed and built by our team over the last two semesters. From the beginning we aimed to create a system that would allow a blind user to remotely control a robot that was capable of projecting the user’s presence at another location. This originally included controlling an arm mounted on the robot, operating a light switch, displaying a audio/video feed from the user’s computer and more. Over the development process many changes were necessary to accommodate our schedule and budget. The system was intended for visually impaired users and as such could not rely on displaying visual data for the user. We developed a tactile feedback matrix to be used alongside a joystick to control the robot by a user without sight.

This project was proposed by Dr. Chien-Liang Fok and was sponsored by the UT Pharos Labs. Dr. Christine Julien worked as our faculty mentor providing us with assistance throughout the project. Our team contained five electrical engineers with backgrounds in embedded systems. Ultimately our team aimed to create a system that would allow a visually impaired user to control the robot as well as a sighted user. We worked from August 2011 through December 2011 to design and plan our system and from January 2012 through April 2012 to build, test, and document our final system. This report details our successes and failures throughout our design and implementation process as well the specifications of our final system.

In the design problem statement section we fully explain the engineering challenge we attempted to solve. In the design problem solution we discuss our planned approach and the various subsystems that made up our final project including the user-interface and Microsoft Kinect. After that, we discuss our chosen implementation and what parts we used. In the test and evaluation section, we will outline the methods we used to test the reliability of our project as well its subsystems including input latency and the overall ease of use of the user interface. In the section on time and cost considerations, we discuss how our budget and schedule influenced our design and implementation including changes such as scaling down the user interface resolution. In the following section on safety and ethical aspects, we discuss the safety concerns that we feel apply to our final system. Following that, we discuss possible future work
and related projects in the recommendations section including things like distributed autonomous mapping of an environment and changing the tactile feedback matrix design. Lastly we conclude our report with our final thoughts on our project including a summary of major points found throughout this report and our thoughts on project’s success.

2.0 DESIGN PROBLEM STATEMENT

The design problem we wanted to solve is to allow a user to control a robot in a remote environment without any visual cues. Controlling a robot involves knowing what the robot is experiencing and responding appropriately. When controlling a robot from a remote location, that same methodology applies. However, when the user is visually impaired, two challenges quickly arise. The first challenge is how to display this information, which is normally conveyed visually, to someone who cannot see. The second challenge is how to enable the blind user to control the robot. Making this two-way communication between the user and the robot becomes much more difficult when we remove the ability to convey information visually.

2.1 SPECIFICATIONS FOR SYSTEM

Our user interface needed to allow the operator to issue commands to the robot as well as receive and interpret data from the robot about its current environment. Our system needed to translate the gathered data into a form easily understood and usable by the visually impaired user. Once these requirements for the user interface and robot were met, we considered the project to be complete. Our team has demonstrated the completed project by creating a system that addresses both of these challenges, as well as obstacle avoidance and feedback systems for the user. We have shown that our system has passed our expectations and has been a success.

2.2 DESIGN PARAMETERS

In order for a blind user to telepresence using a mobile robot, there are several needs we addressed that were solution independent.

- Provide a means for the user to interact with the robot’s surroundings
- Allow the user to control the robot remotely
- Convey information about the location of the robot to the user
- Convey information about nearby obstacles to the user
- Convey information about the robot’s operational status
- Allow the user to intuitively control the robot

Many of our early decisions about how to approach our project defined how well we would meet the goals we listed. For instance deciding to replace vision with another sense became a big portion of how we came up with our solution. We believe our solution as described in sections 3 and 4 meet all these requirements.

### 3.0 DESIGN PROBLEM SOLUTION

There were two main problems that we set out to solve by creating a system that functions without video. The first problem is that, in the absence of visual information, our user needs to be able to feel the environment of a mobile system in a remote location. The other problem was if the mobile system lost the ability to return a video feed, but had a map of its location and was able to localize itself in that map, a user needs to be able to navigate the system to important points in that map. We solved these problems by having a workstation computer control a robot (mobile system) that can drive around and send distances to nearby objects back to the workstation. The workstation takes information from the robot and either creates a map that dynamically updates itself or localizes itself in a static, unchanging map. The map is then sent to a tactile matrix where raised dowels reflect the presence of nearby obstacles. The following sections will discuss the design decisions we made leading up to the creation of a prototype, alternate designs that we dismissed and reasons why, how many specifications we met with given parameters and constraints, followed by an economic analysis of our solution.

### 3.1 DESCRIPTION OF DESIGN SOLUTION

Our design solution meets many of the specifications laid out in the design problem statement section. We wanted a system that allowed both a visually impaired user to telepresence and a normal user to be able to continue teleprespencing without video feed. We created two solutions to these problems. One uses a dynamic map that marks obstacles around the robot in an unfamiliar environment. The second solution uses a static map of the environment that the robot can
localize itself in so that the exact location of the robot is known at all times. This static map has to have been created before the telepresence can be done.

3.1.1 Dynamic Map Solution
One of our solutions required that we create and store a map of the robot’s surroundings. This map would then be translated onto a tactile dowel matrix representing a topographical view of the robot and its immediate surroundings. The map needed to be dynamic and therefore update itself because we anticipated that our system would be used in unfamiliar settings. To accomplish this we found a Simultaneous Localization and Mapping (SLAM) algorithm implementation using an iRobot Create base with a netbook running Robot Operating System (ROS) in addition to a mounted gyro and Microsoft Kinect. The Create generates odometry data using the gyro and built-in wheel encoders. The Kinect uses a laser scan to generate distances to points along a horizontal axis [1]. Together, the Create and Kinect return enough information for the computer to create maps of the robot’s surroundings and know where the robot is inside those maps. Once we had that data, we then picked out the map it created and rotated the map so that the tactile dowel matrix would always feel as though the heading of the robot is always pointed up. This solution addresses many of our requirements especially because we are able to translate the dynamic map onto the tactile matrix for the user. We spent most of our time optimizing this solution because we saved the map it created for our second solution.

3.1.2 Static Map Solution
The second solution requires that we first start with a pre-made map given to the workstation computer. The robot would then use the Adaptive Monte-Carlo Localization (AMCL) algorithm to place itself in the given map. Since the AMCL algorithm does not do any mapping, the pre-made map does not change and is therefore static. An advantage of using this solution is that we are able to mark important locations inside the map and provide sound cues to direct the user to those locations.

We are also able to use the dynamic map solution with the static solution to direct blind users because the dowel matrix is able to show immediate surroundings. For sighted users, there are visual graphical user interfaces (GUIs) that show both the map and the current estimated position
of the robot in the map. The GUI also has a feature that allows the user to choose a position and
direction that the robot can automatically navigate itself to. Together, both solutions are powerful
tools that allow users to navigate remote environments without visual cues.

3.2 DESIGN DECISIONS

Over the past two semesters we have constantly designed and redesigned our project to meet our
specifications. The two following sections discuss the process through which we made our final
decisions as well as alternate designs that we were unable to implement.

3.2.1 Final Design Decisions

Our project has three major components: the workstation, the robot, and the software. Many final
decisions for one component finalized decisions for the others.

We originally needed to map out a room so we researched various systems to do this. The one
with the lowest cost and easiest point of entry was using a Kinect with ROS to run a SLAM
algorithm. ROS only runs on the Linux Operating System. We chose Ubuntu 11.10 because the
makers of ROS recently released a new driver package called Electric. Electric is also used by
the makers of Turtlebot, an iRobot Create base with a mounted laptop and Kinect used by
hobbyists to make maps of indoor spaces. Our sponsor, the Pharos Labs, fortunately had a fleet
of Create bases. We then requested a Kinect and used our own Dell mini 9 netbook to connect
the two. Since everything fit into place, we decided to make a Turtlebot to act as our robot
component.

By choosing a Turtlebot as our robot solution, we needed the workstation to run ROS to
communicate with the robot over a wireless internet connection. Again, ROS Electric runs on
Ubuntu 11.10 so we installed that on each of our laptops. Once we began to map out a room, we
found that we needed to run a ROS program called rviz, a 3D visualization software. This
program is processor intensive and we were only able to consistently run it on an Alienware
laptop. It also ran on a Macbook Pro, but was slow and prone to crashes. Because of this we used
the Alienware laptop for the workstation computer.
Also part of the workstation is the UI which controls the robot, sends video and audio, and receives map data. We created a creative solution that leverages the user’s sense of touch to compensate for the inability to transmit data visually. The team did this by making a tactile feedback matrix of dowels which allows users to quickly feel what the robot sees, i.e. the map data. We decided a joystick would be the most intuitive way to control the robot at least more so than arrow keys on the keyboard. The joystick gives an analog control over the speed of the robot whereas a keyboard would be a constant speed. The keyboard input also caused the robot to jerk which decreased the accuracy of the laser scans so we decided to only use the joystick. We also set up a button on the joystick to give the user polar coordinates to the nearest point of interest over speakers. Just in case the user wanted to speak to someone in the remote environment, a webcam with integrated microphone would get audio and video from the user and send it to the netbook. We looked into video conferencing software and found that Google’s video chat had the lowest latency.

The dowel matrix briefly mentioned above went through several designs (air tubes, solenoids) but we eventually landed on a 4x4 dowel matrix where the dowels are pulled down using servos and pushed up using springs. This was our lowest cost, lowest power consumption idea but it was also only a proof of concept because anything larger than 4x4 greatly raises the cost and using solenoids raises the current draw. The servos are controlled by an Arduino using a Servo Shield because Arduinos are easy to program due to the massive open source libraries available. The shield allows for the simultaneous use of up to 16 servos, perfect for our matrix. The Arduino communicates with the workstation computer over serial. A ROS node that we created listens for an unsigned short corresponding to the state of the pins. The node serializes that data, sends it to the Arduino which activates and deactivates servos accordingly.

Since we chose Turtlebot as our robot base we needed to learn the inner workings of ROS to communicate with various nodes. We figured out that each node is like a thread of an operating system and the threads talk to each other over TCP/IP. Since the TCP/IP protocol was built in to ROS, we didn’t need to use any external software to communicate between the workstation and the robot over the internet. We built most of our software from scratch or taking key ideas from existing nodes. There was a SLAM gmapping node already made that created a map but we were
unable to access the map from other nodes, so we created our own node to listen for Kinect data and make our own map of the room. We also created a node to communicate with the Arduino. A text-to-speech node was created to say the coordinates over the speakers.

Out of all our design decisions, the tactile feedback matrix was the most difficult to finalize. Much of our other decisions were made quickly and early. From the beginning of our project we had a good idea of what needed to be done but were too ambitious so when we ran out of time we had to take off parts of our design like an arm, larger frame, RFID tagging, temperature sensing, and others. In the end we decided on a system that is cohesive and core to our specifications.

3.2.2 Alternate Designs
As this was a year-long project, we came up with other designs but ultimately dismissed them. One of our original hopes was to create a telepresence system capable of manipulating objects in the environment and give the user a high resolution tactile matrix to feel the either the 2D map or the view in front of the robot’s camera. Due to money, power, and time constraints we were unable to implement everything but this section will discuss other designs that could work.

Having a robotic arm on the robot eventually was dropped from the project because we ran out of time. We did not create an arm but we had designed a simple one out of servos and sheet metal. However, with our current robot at a height of about 1.5 feet, the arm may not have been all that useful. Because of that limitation we discussed adding a frame to heighten the robot and give the arm a more realistic height. In doing so, the iRobot Create base may become overburdened since its max load is only 20 pounds. Therefore, raising the height and adding an arm would require a more powerful, sturdier base. If we ever needed it, we discussed the possibility of switching to a Segway base. This base would be able to carry much more but would cost at least 10 times as much. We didn’t look into the actual specifications, but we also would have had to add encoders to the wheels and a gyro to the base to get enough information to run the SLAM gmapping algorithm. All of this would require much more time and money than we had this semester.
The other component that we wanted to upgrade was the tactile feedback matrix. Its final resolution was only 4x4 which we quickly learned is too low of a resolution. If we always put the robot in the center, there are no pins available to signify an object directly in front of the robot. We could move the assumed location of the robot but the we would lose resolution to the left or right. Therefore adding more dowels is the better solution, but servos take up too much space and adding 9 servos to bring the matrix up to 5x5 is just unfeasible. A better approach to the space limitations would be to use solenoids. The current spacing between our dowels is more than stacking solenoids side by side. The only problem with solenoids is that powerful ones capable holding their position against hand pressure draw 1 A each continuously when activated. Typical house fuses are 30A so anything above 6x5 is unfeasible if the interface is powered using a wall socket. Another design we considered uses two motors to move a tool along tracks that raises and lowers pin which lock into place. This is a low power, low cost solution but creating it would require machining small custom parts. Also the refresh rate would be slow because the motors move pins one at a time whereas servos and solenoids raise one pin each. Although a 4x4 matrix worked as a proof of concept, the technology available to us to feel a remote environment is just not adequate enough for a higher resolution.

3.3 SPECIFICATIONS MET
At a high level, our main specification for this project was creating a telepresence robot for a blind user within one year and under a $1000 budget. To do this we needed to create a system for allowing the user to issue commands to the robot as well as receiving and interpreting data from the robot about its current environment into a form easily understood and usable by the user.

As mentioned in the Design Problem Statement Section, in order for a blind user to telepresence using a mobile robot, there are several needs we addressed that were solution independent.

We were unable to provide a means for the user to interact with the robot’s surroundings aside from driving in the environment. This is the only specification that we half-completed. We discussed adding an arm to manipulate objects but did not have enough time. As for the remaining requirements, we are proud to say we completed the rest. The user was able to control the robot remotely using a joystick. When using a static map, we were able to give the robots
location as a distance from an important location. We were definitively able to give information about obstacles to the user. Our dowel matrix accurately matched the robot’s surroundings at the 4x4 resolution. The robot stored its own map of the environment and can be accessed using our software. This is demonstrated by using that map to create an ideal 23x23 matrix in software which is displayed by a GUI using Python. Although we did not incorporate the robot’s operational status into our feedback methods, it would be straightforward to modify the text-to-speech node to verbally convey that information on the push of a joystick button. As for allowing the user to intuitively control the robot, our mentor Prof. Julien was able to use our system blindfolded and navigate the robot to a predetermined location without trouble. Based upon meeting these individual goals, we consider our project a success.

3.4 ECONOMIC ANALYSIS
Due to the abundance of electronics lent to us by the Pharos lab and our group members, we only needed to spend about $300 on parts even though we used about $2000 worth of equipment. A detailed list of our parts is in Appendix A. Creating this system from scratch is a costly endeavor but compared to $42000 which is the cost of a Guide Dog, this isn’t that much[2]. We aren’t saying that our system can replace a guide dog, but it does have its uses. One of our alternate problems for this project was if the robot lost video, but the user was not visually impaired. If it is imperative that the user continue using the robot (e.g. a rescue mission, intelligence), then having $2000 worth of extra equipment as a backup is priceless.

4.0 DESIGN IMPLEMENTATION
Our final product consists of two main subsystems, the robot, and the user interface. The user interface (UI) includes a joystick, a keyboard, a microphone, and a webcam as inputs from the visually impaired user. Outputs from the UI are a surround sound audio system and a dowel matrix system for feedback to the user. The robot incorporates input sensors including a Microsoft Kinect and a microphone to gather data about its environment. The robot contains speakers, a monitor, and motors as outputs to interact with its surroundings. The user interface connects wirelessly to the robot and uses the Robot Operating System (ROS) to send commands and receive environmental data to the robot. The robot responds in the same way and receives
the commands and sends back the environmental data. Shown below is a high level input/output
diagram that describes the different modules attached to our user interface system and the robot:

![Input / Output Diagram](image)

**Figure 1. Input / Output Diagram**

### 4.1 ROBOT DESIGN

The robot runs ROS and receives commands from the user interface and sends out environmental
data. The commands from the joystick on the user interface tell the robot how to drive the base.
The microphone on the robot sends audio to the user’s speakers and the user’s microphone also
sends audio back the robot’s speakers. The screen on the robot displays the user to the remote
environment. The Microsoft Kinect gathers distance data of obstacles in the environment and
returns it to the user interface. The robot is completely battery powered, and all of the
information transfers are coordinated by a small netbook mounted on the base. The robot and
interface are designed to be safe for the user and the environment. A picture of our final robot is
shown on the next page.
4.2 USER INTERFACE DESIGN

The user interface consists of three different parts. First, we have methods to control the robot. These methods include a joystick for direct movement control. For other actions, such as telling the user the heading of an objective we added dedicated button on the joystick. The second part is giving information back to the user. This part consists of two components: a matrix of dowels and a surround sound system, both of which rely on information from the robot. The final part of the interface allows the user to communicate remotely with people near the robot using a webcam, a monitor, speakers, and a microphone.

4.2.1 Inputs and Outputs

The user interface is the only point of interaction the user has with hardware and software. The user can sit down at the computer running the interface and have all the UI subsystems available. As shown in the I/O picture, the user has a keyboard for inputting commands, and a joystick for moving the robot around. One button on the joystick drives the robot, while another gives an
audio heading to the destination. The webcam displays the user to a monitor on the robot. There is also a microphone to speak to people in the remote environment. We included a software module that interprets some of the environmental data and uses a speech program to relay an objective’s heading to the user. For example if the user feels lost he can hit a button causing the UI to verbally say “The target is X degrees to the right and Y units away.” The interface takes distance data from the Microsoft Kinect and displays this data on a dowel matrix. The tactile system is a 4x4 matrix of dowels that are raised and lowered by servos. It provides a tactile 2D top-down view of the environment with the robot centered in the matrix. Raised dowels correspond to obstacles around the robot. This component of the interface is crucial to the success of our project as it displays the main form of environmental feedback to the user. The entire interface is powered using a wall socket.

4.2.2 System Control
The main interface between the user and the robot is the joystick. The user can control the robot directly with the joystick. Joysticks are an intuitive movement control because they provide an analog input which gives controlled acceleration. The learning curve for a joystick is small, even if our user has never used a joystick before.

Other than directly controlling the movement of the robot, the user can issue various commands to the robot. The keyboard is the primary method to issue commands. While some visually impaired users might have trouble with a keyboard at first, most computer savvy individuals can type without looking at the keyboard. The user can issue many different commands to the system, such as requesting the location of a destination. In this way, the user is able to request information about something specific from the robot when he or she chooses.

4.2.3 Receiving Information
The user can receive information from the system in two ways. The first method is a 4x4 matrix of dowel rods. The dowels retract when a servo activates, otherwise a spring returns the dowel to an extended position. The matrix’s main purpose is to inform the user of obstacles in the robot’s surroundings. The areas with extended dowels signify obstacles to the robot’s movement. For example, when the user puts his or her hand on the matrix and feels extended shafts to the left of
the center, the robot is seeing an obstacle to its left through which the robot cannot move. This matrix is always created so that the top of matrix corresponds to the direction the robot is looking, allowing for intuitive fast navigation.

4.2.4 Remote Communication
There are some situations in which the user needs to communicate with people on the robot’s end of the system. For those situations, the system takes advantage of a webcam on the user’s end and a monitor on the robot’s end of the system. The user can issue a command to start a live feed, the webcam then begins recording the user, audio and video, and transmitting that information to the robot. The robot then displays the video on a monitor and plays the audio through the speakers. The robot also constantly records audio from its surroundings and transmits it back to the user. This allows for direct communication between the user and others, with the user being able to hear people in the environment and people in the environment being able to hear and see the user.

5.0 TEST AND EVALUATION

We first tested each component of our project. After each component was working properly and had passed all of our tests and evaluations, we moved onto module testing, for which we separated the components into two groups, the user interface and the robot components. After each module was tested, we tested the entire system in our test environment.

5.1 COMPONENT LEVEL TESTING
The components that we needed to test our project were the tactile feedback matrix, joystick, and voice commands. Other components that we had in our project were ROS and localization. They are all tools that we were using at a low level. The components that we needed to test used these basic tools. We did not need to personally test these components because the ROS community had already extensively tested them.

5.1.1 Tactile Feedback Matrix
In order to test the tactile feedback matrix, we sent fake map data to the user interface. The UI
was then supposed to transform this data into something that the Arduino could easily use to control the matrix. Using our own fake map data allowed us to know exactly what the matrix should be showing. We could also test the range of the matrix by sending data with the area completely filled with obstacles, which should never happen in real use, or completely empty. This allowed us to measure the difference in each dowel and servo from fully retracted to fully extended. The ideal result was that all of the dowels were at the same height when they were given the same value. We allowed for a minor difference in height as long as that difference was negligible to the human touch. After the range of each servo and dowel had been calibrated, we moved on to testing the translation from the map to the matrix. Because our matrix was relatively low resolution, there weren’t too many alternatives to test. When we sent it a certain set of map data, it would try to copy that map as well as a 4x4 matrix could. That meant that obstacles raised the dowels and free space lowered the dowels. The matrix performed as reliably as expected for such a low resolution. Because the matrix was a proof of concept, rather than a viable navigation option, the testing did not need to be extensive.

5.1.2 Joystick
The joystick operated the robot via teleoperation. We tested both ends of this communication. By connecting the joystick to the workstation, we made sure that the ROS node was seeing the joystick properly and sending the correct input to the robot. On the robot end, we sent fake joystick data to the ROS teleoperation node and made sure that the robot moves the correct amount, based on our configuration. Because the joystick teleoperation was already written for us, we spent our time mapping the right joystick buttons to the correct teleoperation inputs. The joystick operated exactly as it was supposed to.

5.1.3 Voice Commands
Testing the voice commands was relatively straightforward compared to the matrix. The code for the voice commands took in parameters from the map and output an audio command. These parameters were the robot position and heading and the location of the target. The audio command told the user how far away the object was as well as what direction it was in. Testing the voice commands involved sending fake positions and headings and making sure the UI output the correct command. What the command said should match exactly with the results of
doing the heading calculations by hand, rounded to the nearest degree and unit. Because the code involved simple mathematics, the testing progressed painlessly and we had no issues with the voice commands.

5.2 MODULE LEVEL TESTING
In our project there were two major module level items that we needed to test for our final system to be functional: the robot and the UI. These modules worked closely together in our project.

5.2.1 Robot Module
The robot module was tested by giving it data we expected to get from a remote UI. The inputs to this module were the joystick data and a video/audio feed. The outputs of the robot were audio, down-sampled environment data, and location data. The audio data was representative of real data to be sent to the UI. For environment localization we needed to test that the robot is sending back the data we were expecting as it localized itself to the environment. To do this we used a pre-made demo environment and catalogued the data the robot sends out. To see what the localization looked like we looked at published localization data in ROS. Comparing this data to what we knew the robot saw allowed us to fine tune configurations to get the robot to localize better. To test the joystick input to the robot we created a list of fake joystick movements and sent them to the robot and measure how far the robot went and at what speed. Using this data we were able to fine tune how close an obstacle needed to be before we sent back data telling the robot there is an obstacle up ahead. This allowed for better real time control of the robot. The final output was a video feed, which we expected to have few issues. However, we did need to do a stress test to see how much bandwidth it took to get to our remote UI. With the robot and UI on the same network, we had little to no issues with bandwidth. To iteratively test the localization, we started with an empty room and made sure that the robot could see 4 walls as obstacles. Then we added an obstacle in the center of the room and made sure that the robot saw that as well. This methodology allowed us to increase the complexity of the environment, yet continue to have the robot work correctly. To test robustness of the robot we needed to test how correctly the data that we get from the robot represents the real environment. This allowed us to inform the user how accurate each of the steps of the feedback matrix is.
Most of the problems we ran into with the robot were with the iRobot Create base. The base has two sensors that we used, a gyro and an odometer for each of the motors. Both of these sensors were unreliable. Because the we were just using someone else’s platform, rather than one we made on our own, we could not fix the sensors. The data that these sensors collect were integral to how well our robot performed, meaning that we could not find a workaround. This resulted in us having to supply a static map to the robot, rather than having the robot map the environment itself. By doing so, we could rely less on the reliability of the sensors and more on the accuracy of the static map.

5.2.2 User Interface Module
The UI consisted of a computer workstation, the Arduino microcontroller, and the tactile feedback matrix. These components combined as a module needed to be tested. The inputs to this module were the outputs from the robot module. The outputs to the robot were joystick data, video data, and audio data. To test this module we fed it a known map of the environment and some staged audio data. The test for the audio data was focused around making sure that the data contained enough information to guide the user to the objective. We input our own audio from a remote location and made sure we are able to hear it on the speakers attached to the UI. A large part of the audio testing was checking the robustness to make sure that the audio input and the map-data/video outputs did not have high latency. We also needed to test that the environment was displayed correctly. This tested how the environment was encoded and passed from the workstation to the Arduino and then to the matrix. Making sure that these three components worked together was the main goal of this test. We added iterative maps of the environment simulating a path the robot took through the environment. This allowed for a more robust testing process since we not only saw how the map was rendered on the feedback matrix but also how the map changed as obstacles were encountered.

5.3 SYSTEM LEVEL TESTING
To test for the overall functionality of the design we created a static environment and maneuvered our robot to the objective using only the feedback matrix and audio commands. A
mockup of the test environment is shown below in Figure 3. This tested the maneuverability of the robot as well as the user interface’s ability to express data to the user.

The input for this test was the environment to be navigated through. We planned on starting with a static environment and if time permitted, expanding the project to a more dynamic environment that better represents the real world. We ended up having to just use the static map. Within the static map, however, adding more obstacles did not affect how the system performs unless there was no path from the starting position of the robot to the target. By adding more obstacles and tightening the path between the starting position of the robot and the target we were able to evaluate the robustness of the system. The tactile feedback matrix could not display enough data with its resolution. To compensate, we created a virtual GUI that simulated the ideal resolution of the matrix if we had the resources to build it to a proper specification. Our test users could then use the virtual GUI to navigate, while also noticing how the virtual GUI scaled down to the physical matrix. Although most of the system level testing was done by the members of the group, we realized that this was not an effective way of testing the system, since we all knew the
environment very well. At the Open House demonstration of the project, however, we were able to expose others to the system and they were able to successfully navigate the robot to the target, without too much of a problem.

6.0 TIME AND COST CONSIDERATIONS

Even though we were able to complete a working project by the senior design open house, there were a lot of components of our original design that were removed due to time constraints. During the course of our project we had frequent hardware malfunctions that delayed our development progress. To overcome these issues we kept several replacement components nearby so we were able to switch them out when we suspected a malfunction. Also, the lab our group worked in is known for its poor wireless communication, which also caused delays in our development. Unfortunately, the test environment we were using to run our project was located in the lab and was cumbersome to move around campus. Due to these delays, we were never able to test our user interface on a visually impaired user, resulting in less data to use to complete and refine our design. After the first design iteration of the tactile feedback system we discovered that the angle the servo motors were pulling on the dowels would cause movement problems. We then redesigned the tactile feedback system to be more fluid and not restrict the movement of the dowels. This mechanical design delay caused us to change the resolution of our tactile feedback system from an 8 x 8 grid of dowels to a 4 x 4 grid. This loss in resolution repurposed our project into a proof of concept of a tactile feedback system for the visually impaired. These delays also resulted in several features that we originally intended but cut due to a lack of implementation time such as user voice commands and a robotic arm. The previously created Gantt chart is located in Appendix B.

Overall, we were able to keep in a reasonable budget for our project. Most of the hardware we used was provided by the Pharos Labs including the robot base, Kinect and servos for the user interface. One of our original designs for the user interface called for using 64 solenoids, but it was cut due to their high cost and current draw. The software was either from the open-source community driven operating system ROS or written by us during the development of the project.
7.0 SAFETY AND ETHICAL ASPECTS OF DESIGN

There were few safety and ethical concerns that we had to deal with. One such concern was making sure the feedback matrix could not harm the user’s hand. We dealt with this by using servos. Servos inherently move relatively slowly, meaning that it had little chance of causing any damage by moving too fast. We also sanded the surface of the matrix and each dowel, in order to avoid loose wood and metal causing splinters. By having the user to control the robot, we avoided any accidents that could be caused by autonomous robot control. Even when we lost internet connection, the robot would not go out of control.

In terms of the ethical use of the project, the robot-workspace communication has too much latency to be effective in timing critical applications. With the communication protocols we used, the connection between the workspace and the robot would sometimes drop, resulting in no response from the robot. This could potentially lead to major problems if timing is a major part of the application.

8.0 RECOMMENDATIONS

Even though we successfully completed our project, there are many aspects of our original design that did not make the final product. Future groups could try to incorporate new features to increase the robustness of our design or try to recreate our project with information we learned throughout the development process.

8.1 RECOMMENDATIONS FOR FUTURE DEVELOPMENT

There are many interesting projects that could be undertaken related to our work. First off, future projects could encompass multiple robots to map an environment and communicate with each other to autonomously map a large environment. This would encompass the room mapping aspect of our project but instead, using a distributed system to communicate data and build a global map. To further expand on our project the robot could use an RFID reader to locate key targets in the environment dynamically so that the robot would not need to know the coordinates of all the targets before it begins the telepresence session. This would resemble a more real-time
system that protects against target objects being moved within the environment. Finally, a robotic arm could be added to the robot to allow it to interact with the environment in a more robust manner. This would allow the user to not only navigate a remote location but interact with the objects in the environment.

A group could also expand on the user interface we created for movement and feedback of the robot. A future project could be centered around creating a tactile feedback system that uses air pressure to give the visually impaired user feedback about their environment. Other feedback systems could be researched and compared to provide the best form of feedback for a visually impaired user. Also, voice commands could be implemented into the user interface so that the user can speak commands for the robot to perform. Lastly, our user interface system only gives a top-down view of the environment of the robot, if the user interface could better display what the robot is looking at, the user would theoretically be able to better navigate and interact with the environment.

8.2 RECOMMENDATIONS FOR FUTURE GROUPS
After completing the project, we have many recommendations for anyone who is undertaking an identical project. First off, we ran into many hardware issues that stunted the development of our project. During the design and experimentation of the project it would have been useful to have duplicates of many hardware components on hand so we can readily swap out parts we thought were giving us incorrect or inconsistent data. This includes having multiple charging stations for each power source used in the project, this way we would lose no development time when batteries die. Due to the nature of the project, writing and testing software heavily depended on using the robot itself, making it difficult to test the software without a power source.

Additionally, more research should be done on the hardware used to build the robot, specifically the robot platform that will be controlling the motors and communicating with ROS. While the iRobot base served our purpose well, we ran into frequent hardware issues that could have been possibly avoided with using a different robot base.

Another suggestion for future groups is learn everything they can about ROS. While most of the team went through some tutorials created by the ROS community, all team members should be
required to complete all tutorials about ROS before the development for the robot begins. This way all members have a firm foundation on ROS and have a better sense on where in the source code to start looking for answers to ROS related questions. Also, if a group has a member already with experience in ROS they would have an advantage in software development.

During development we also ran into many wireless communication issues when communicating between the robot and the user interface. Unfortunately, the lab we worked in has known wireless issues and our testing environment was difficult to move and recreate. Future groups can either request to work in a lab that has better wireless communication or purchase better wireless communication hardware to try and remedy the situation.

From a research perspective, more investigation should be done on how the visually impaired use their senses to gather data about their environment. From this research, future groups may be able to better design a type of tactile feedback system that would benefit a visually impaired more than our final design. Finally, more research could be done on choosing the gyroscope and odometer used on the robot. Our group quickly learned that a small error in one of these sensors compounded quickly when trying to create an accurate map for a visually impaired user to navigate remotely. Doing more research and purchasing higher quality sensors could have resulted in a higher quality static map to use for navigation. Although we consider our project a success there are many aspects of it that could have been improved to decrease our development time to allow us more time for testing and evaluation.

9.0 CONCLUSION

Our group wrote this report to document our senior design project and outline the successes and failures of it for future students. Our team consisted of Razik Ahmed, Thomas Brezinski, Stephen Hall, Zachary Lalanne, and Ali Unwala; all of us are fourth year electrical engineers specializing in embedded systems. We worked under the guidance of Dr. Chien-Liang Fok and Dr. Christine Julien over the last two semesters to create a telepresence system for the visually impaired. The UT Pharos Labs provided most of the materials used as well as an environment for us to work in.
Throughout the design and implementation process our team learned a lot about time management and work division amongst team members. During the initial planning stages of our project, we greatly overestimated what our team had time to accomplish. We made assumptions that certain aspects of the project, such as getting the robotic base operational, would be trivial tasks and they proved to be much more complicated. We attempted to build as much slack time into our schedule as possible during the planning phase, but the incorrect estimations for how long many of the project aspects would take ultimately consumed all of our slack time. We ended up removing many aspects of our project that we wish we had been able to keep. These included the user studies, attaching an arm to the robot to operate a light switch, including voice commands, and dynamically finding targets using RFID tagging. The removal of these aspects was not detrimental to our final project, but they would have greatly enriched our final system and provided more material to any groups that wish to take our project further in the future.

Our team found that the engineering challenge we were attempting to solve, providing an intuitive telepresence system for the visually impaired, has practical solutions, but is sensitive to cost, size, and power requirements. Our tactile feedback matrix was too low resolution to provide accurate information to the user, but any higher resolution was infeasible given our servo design due to the cost per servo. A more scalable solution in terms of size and cost would have been to use solenoids in place of the servos, but this solution also requires much more power to operate. In the future, any team wishing to improve upon our project should spend time investigating more alternate designs for the user interface. Also the base we used, the iRobot create, had unreliable odometers that made map creation difficult. Improved sensors would have improved our final system’s performance.

Ultimately, our team feels that our final telepresence system met all the core requirements we set out to meet. It enabled a user to remotely control a robot with a high degree of confidence and navigate to a target destination without visual information. There is certainly room for improvement upon our design and we hope future groups of students are able to take away some of the lessons we learned and improve upon our system.
REFERENCES


# APPENDIX A – BILL OF MATERIALS

## USER INTERFACE

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino Mega 2560 Micro-Controller*</td>
<td>$46.55</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
<tr>
<td>Joystick*</td>
<td>$19</td>
<td><a href="http://www.amazon.com/Saitek-PS40U-Aviator-Joystick-Console/dp/B001EYU1VY">http://www.amazon.com/Saitek-PS40U-Aviator-Joystick-Console/dp/B001EYU1VY</a></td>
</tr>
<tr>
<td>Desktop Computer*</td>
<td>$479</td>
<td><a href="http://www.bestbuy.com/site/Dell+Inspiron+1570-9445NBK+Desktop+%26+20%22+LED+Monitor+Package/9999170400050022.p?id=pcmprd170100050023&amp;skuId=9999170400050022">http://www.bestbuy.com/site/Dell+Inspiron+1570-9445NBK+Desktop+%26+20%22+LED+Monitor+Package/9999170400050022.p?id=pcmprd170100050023&amp;skuId=9999170400050022</a></td>
</tr>
<tr>
<td>Matrix servos</td>
<td>$160</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
</tbody>
</table>

## ROBOT

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>iRobot Create Movement Base*</td>
<td>$129.99</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
<tr>
<td>Dell Mini 9 Inspiron 910 Laptop</td>
<td>$150</td>
<td><a href="http://www.amazon.com/Dell-Inspiron-Laptop-Notebook-Black/dp/B0024MB1A0">http://www.amazon.com/Dell-Inspiron-Laptop-Notebook-Black/dp/B0024MB1A0</a></td>
</tr>
<tr>
<td>Arduino Mega 2560 Micro-Controller</td>
<td>$46.55</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
<tr>
<td>XBOX Kinect</td>
<td>$150</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
<tr>
<td>Battery Pack</td>
<td>$170</td>
<td><a href="http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List">http://pharos.ece.utexas.edu/wiki/index.php/Hardware_List</a></td>
</tr>
<tr>
<td>ROS</td>
<td>$0</td>
<td><a href="http://www.ros.org">http://www.ros.org</a></td>
</tr>
</tbody>
</table>