Telepresence in a University Environment

Selena Groh, Serena Thoma, and Trung Truong

Introduction

Imagine a child breaks her leg, missing months of the school year to recuperate. She comes back behind on her academics and isolated from her peers. Imagine a scientist needs to diagnose a critical problem in a system 1,000 miles away and all the flights are sold out. The system fails without the guidance of the expert. Imagine further that a young man is medically quarantined for years. He grows withdrawn from lack of interpersonal contact.

These problems are unique, but each case could benefit from a system which makes remote attendance possible. Such telepresence technology is the subject of this project. Utilizing video calling functionality as well as a browser-based navigational interface, we will allow a remote user to control a TurtleBot2's movement and see and hear what is happening in its environment. Others in the environment will be able to see the remote user and interact with them through a tablet interface. By implementing this technology, we hope to allow students who cannot come to Halligan to attend lectures or work with a partner remotely.

Related Work

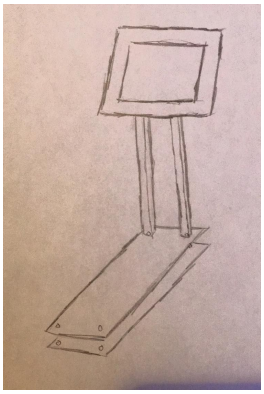
The paper "Teleoperation with Intelligent and Customizable Interfaces" focuses on the larger applications and challenges with teleoperation robots [1]. The study branched out to the larger application of using a teleoperational robot with manipulator that allowed users to perform tasks with robot. The paper discusses the relationship between the robot and its operator, specifically the extent with which the robot is carrying out commands autonomously or with user control. Almost every task that a human carries out can be broken into sub-tasks, and the same is true for the action of a robot. Although the robot in our project will not be manipulating the environment, we will experiment with different levels of autonomous and user-controlled navigation to see how different types of commands affect both the accuracy with which the robot carries out the actions and the experience of the user when giving commands.

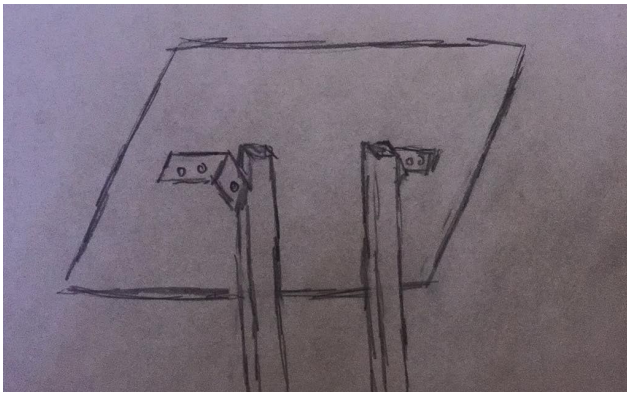
Meanwhile, the "Building a Telepresence Robot Based on an open source Robot Operating System and Android" documents how one can use ROS and Android package to control an iRobot Create robotic base [2]. This article includes both the software and hardware implementation of the telepresence robot, and we expect to explore and experiment at least one of the method described in this article in our robotics model.

Problem Formulation and Technical Approach

Hardware

For our project of a telepresence robot, we decided to add a tablet interface to the turtlebot in addition to the laptop running ROS. The design for a structure to mount the tablet onto the turtlebot is shown below. The structure will consist of two plates on the bottom that will be screwed together around the top shelf on the existing turtlebot, two rods extending upward from the bottom plates, and a frame at the top that will hold the tablet for viewers to see.





Figures 1-2: Design for Tablet Turtlebot Mount

The larger plates will be laser cut from ⅛ inch acrylic, which is available for student use in Bray Labs. The exact dimensions of the parts will be determined once the tablet arrives.

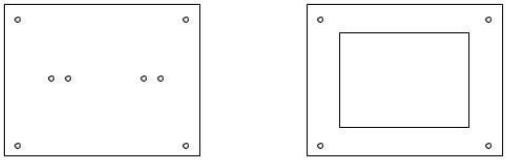


Figure 3: Tablet Frames

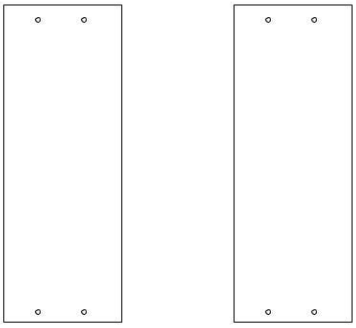


Figure 4: Bottom Mount Plates

Additional parts include two L-brackets, two metal bars of length 2’ each, and approximately 24 nuts and 24 bolts of varying sizes. All of these parts are available for student use in the Bray Shop.

Video Calling

Several video chatting applications were considered for this project as a way to provide the user with video and audio of the surroundings of the robot. Many applications currently exist that are easily accessible, have intuitive interfaces, and perform the bidirectional video and audio connection that is crucial, not only for the operator of the robot to be aware of their surroundings, but also for users near the robot to be able to experience the presence of the operator. Since many applications are specific to an operating system, the two applications that would work best for this project are Skype and Google Hangouts. Both of these applications are free and generally familiar to the public. Skype is the ideal video chatting application for this project because its interface can be easily accessed on both a desktop computer and a tablet computer, while Google Hangouts is mostly designed for desktop computers and contains features that are not applicable to this project.

Tablet Selection

Of the hardware needed for this project, a tablet is the most important and complex to consider because it both defines the experience of human users in the presence of the robot and also limits the capabilities available to the user operating the robot. The user needs are outlined below:

- Video chatting application that is compatible with tablet software
- Wifi capabilities, allowing video chatting application to operate wirelessly
- Camera that is compatible with video chatting application
- Microphone that is compatible with video chatting application
- Speaker that is able to play audio from video chatting software at a reasonable volume
- Cost should be reduced as much as possible

Luckily, for our project, most tablet computers on the market fulfil most of these needs, and reducing the cost of the tablet results in lowering the quality of these capabilities, but does not remove them. Given these specific needs, there are two tablet options for this project. The ideal option is the Amazon Fire 7 Tablet, which is enabled with wifi access and is compatible with all up to date applications available through Amazon, including Skype [3]. The second option for a tablet is the DigiLand 7” Tablet, also equipped with wifi access, is a slightly outdated model that is no longer compatible with Skype, meaning that Google Hangouts would be used as the video chatting software [4]. The major trade off between these two tablets, other than their compatibility with Skype, is their price, as the Amazon Fire costs $49.99 and the DigiLand 7” Tablet costs $24.99. Once more information is provided about the budget of this project, a single tablet will be chosen to mount on the robot.

Navigation by Remote User

Although there are a lot of approaches one can take to accomplish the goal of communicating with the robot over the Internet, the concept is very similar. The laptop connected to the turtlebot will serve as a server, and the laptop that is in charged of controlling the robot will be client and subscribe to such server. Data can be sent asynchronously between the server and client, as the client will send data regarding to the controlling commands, while the server will update subscribed client with information such as odometry and topological location.

From a server-side perspective, we expect to use the rosbridge package, a package offered by ROS that allows interaction between ROS and external softwares through JSON interface. While rosbridge and the turtlebot packages are running, the rosbridge package is also able to open up a WebSocket connection, which can be accessed from a browser through the Internet. This connection will serve as a means for the server and the client to communicate to one another.

From a client-side perspective, we expect to build a GUI system that will communicate with the server created by the onboard laptop connected to the turtlebot. This GUI platform will be run on a browser and will be subscribed to the server through WebSocket protocol. While the GUI will be coded mainly using HTML and JavaScript, additional libraries, such as jQuery and roslibjs, a library offered by ROS to parse and send APIs to the rosbridge package, will be used to make the communicating protocol easier.

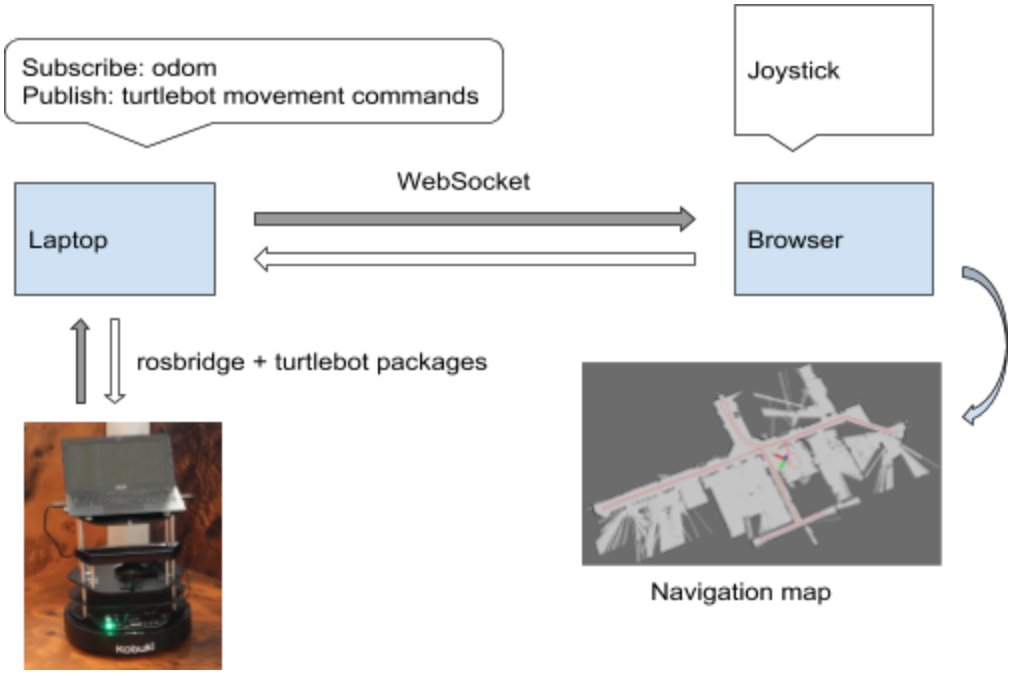


Figure 5: Communication between browser and turtlebot

Although the GUI system will provide basic controlling options, such as forward, backward, left, right, stop, etc. we seek to implement some sort of autonomous maneuvering command. Using the pre-scanned map of Halligan, we seek to build an animation of that map in the browser and update the location of the robot accordingly based on odometry data. We also seek to create a “click-and-drag” option, in which a coordinate will be generated once one clicks on the map animation, and such coordinate will be sent to the server through WebSocket protocol. That coordinate information will pass through rosbridge, will be parsed, and will be actuated on the robot as lower level command.

Evaluation Plan and Criteria for Success

Our project must be able to accurately represent the remote user in the robot’s environment and reflect the robot’s environment to the remote user. In order to do this, our system must have specific core functionalities:

1. Video communication between the tablet and the remote interface
2. Audio communication between the tablet and the remote interface
3. Remote navigation of the robot via manual (left, right, forward, back) commands
4. Remote navigation of the robot by specifying a goal location on a map of the robot’s environment

For video communication, a person in the robot’s environment should be able to view the remote user in near-real time. Likewise, the remote user should be able to view a video feed of the robot’s environment taken from the tablet’s camera in near-real time. We will evaluate this functionality by starting our video calling protocol with a remote user in a different room as the robot and verifying that the video feed to both ends is not significantly lagging.

The audio communication component will be similar to the video communication but instead evaluate the microphone and speaker systems of both the tablet and the remote user interface. The audio communication must not be significantly lagging and must sync relatively well with the video feed.

Manual remote navigation must function so that the remote user can turn the robot and move it forwards (and perhaps backwards, depending on safety mechanisms) with relative ease. In particular, the lag time between the command being issued and the movement being performed must be short enough that the remote user can respond to events in a reasonable amount of time. We will test this first by controlling the robot from the same room so that the remote user can see the robot’s environment for safety reasons. If that is successful, we will test by dividing the robot and remote user into separate rooms and performing the same task.

Finally, goal-based remote navigation must allow the remote user to direct the robot to a specific location on a map and the robot to navigate there autonomously. In order to test this, we will first use goal locations only a couple feet away from the robot in a relatively uncrowded

environment. If these tests are successful, we will eventually extend the testing to more complicated environments or more lengthy routes.

References

1. Dragan, A. D., Srinivasa, S. S., & Lee, K. C. (2013). Teleoperation with Intelligent and Customizable Interfaces. *Journal of Human-Robot Interaction, 02*(02), 33-57. doi:10.5898/JHRI.2.2.Dragan
2. Do, Ha M. Mouser, Craig J. Sheng, Weihua. (2012). Building a Telepresence Robot Based on an open source Robot Operating System and Android, Theoretical and Applied Computer Science (TACS 12)
3. https://www.bestbuy.com/site/amazon-fire-7-tablet-8gb-7th-generation-2017-release-black/5822948.p?skuId=5822948
4. https://www.bestbuy.com/site/digiland-7-tablet-16gb-black/5246601.p?skuId=5246601