CS2951K Final Report

ROS-enabled Teleoperation with the Microsoft HoloLens

Leon Lei, Liam Walsh

Abstract

Teleoperation of robots is an important tool by which humans can interact with robots, and recent work with virtual reality (VR) teleoperation systems have enabled users to control robots in new and useful ways. While there currently exists a robust and ever-growing platform of tools for VR teleoperation, the corresponding tools for mixed reality (MR) teleoperation are scant. Our approach aims to adapt ongoing work in VR teleoperation systems to the corresponding MR systems, specifically the Microsoft HoloLens. Through Unity and our lab's ROS Reality package, we connected a user to a Baxter robot and allowed for teleoperation with an intuitive interface that was designed uniquely for the HoloLens. This involved using manipulation gestures that were native to the HoloLens as well as voice commands and overlaid holographic controls. With our tool, which can also generalize to other ROS-enabled robots, we were able to operate the arms of the Baxter using both gaze and hand position controls while opening and closing the grippers to pick up objects. Unlike its VR counterpart, MR teleoperation has the advantage of not completely removing the user from the environment and allowing them to see the actual robot's physical actions, providing a new level of intimacy in human-robot interaction.

Introduction

The advent of virtual and augmented reality technologies are changing the ways in which humans interact with robots. In particular, new capabilities in the space of robot teleoperation have moved us away from the need to arduously input commands and coordinates on a screen. Instead, a robot arm can now simply mimic the movements of a human arm, resulting in a more natural level of interaction that can also aid in robot learning. Such a system also enables users to operate a robot without being in its physical presence, and in some cases even if they do not have one for themselves. This opens the doors for crowdsourcing of robot training data, which have historically been quite cumbersome to obtain. Our project tries to tackle some of the challenges associated with using current MR technologies for robot teleoperation, with the hopes that it will make possible more advanced manipulation tasks and set the stage for future projects in this field.

Since VR and MR are still relatively new technologies, interfaces for robot teleoperation are mostly young and in their developmental stages. A number of labs including our own have already successfully used virtual reality systems to operate robots[1] and have them perform complex manipulation tasks. However, the use of mixed reality systems for the same purpose are few to none. Brown's robotics lab has developed the ROS Reality package for connection between ROS and Unity, but its primary usage is with the HTC Vive VR system. The few existing projects that related to MR had crude controls and made limited attempts to utilize the full capabilities of the Microsoft HoloLens. Moreover, they did little work to tackle the problem of user experience, which we believed was crucial for a new and developing system that has seen relatively limited public use to date.

Our technical approach involved refactoring our lab's pre-existing ROS Reality package to allow for communication with the HoloLens. Building on top of the original framework, we established communication with the Baxter through ROS topics, over which messages were sent back and forth using nodes that served as publishers and subscribers. We also wrote scripts in C# that would allow us to control the Baxter's arm movements. These scripts were attached to Unity objects that acted as controllers, which a HoloLens user could interact with and manipulate to send messages containing transform data whenever their positions updated. Through this, a user is able to operate Baxter by either using their gaze or their hand position, and they can select between these two settings when they first launch our program.

To achieve much of our program's functionality, we leveraged the Microsoft Mixed Reality Toolkit, a Unity library with a collection of scripts and components intended for development in the HoloLens. This library provided us with a number of helpful features that we were able to integrate into our system, including creating a cursor with feedback, adding interactable features to Unity objects, and recognizing and responding to voice commands. Voice commands were especially important, as they became the chief way by which users can open and close the grippers of the Baxter.

We evaluated our system by first trying to pick up several objects ourselves, using both the gaze and the hand position controls. This was a bit difficult initially, but more practice in the system made it easier to control the arm. We also tested out the other features of our system and captured our results using the built-in camera on the HoloLens. For further evaluation, we had a couple people completely unaware of our system and the Baxter attempt to pick up objects, to a fair degree of success.

Related Work

Our work was initially inspired by the recent paper *Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation*[2] by Tianhao Zhang et al., in which a PR2 robot was trained to perform a series of complex tasks using expert data collected from human trials conducted with a VR teleoperation system. The results were very good, and the most exciting part is that the authors claim only 30 minutes of recorded data is necessary for the robot to successfully learn the desired skills. At the onset of our own project, we had hoped to ultimately emulate the results of this paper but in MR, so the first component was to create an interface for MR teleoperation that was easy to use and effective in allowing users to perform the desired manipulation tasks.

The ROS Reality package developed by our lab and the corresponding paper[3] by Whitney et al. were immensely helpful to our work, as they set up the grounds for the protocol we used for communicating between the Baxter robot and the HoloLens via Unity (Figure 1). A related package that we drew from which also built off ROS Reality was Holobot, which offered capabilities to visualize the URDF of a robot in the HoloLens. However, this project was lacking in its ability to allow users to control the robot in a meaningful way and also had numerous instances of deprecated code sources. Moreover, to the best of our knowledge, no previous project has focused as deeply on the problem of user experience while using the HoloLens for teleoperation.

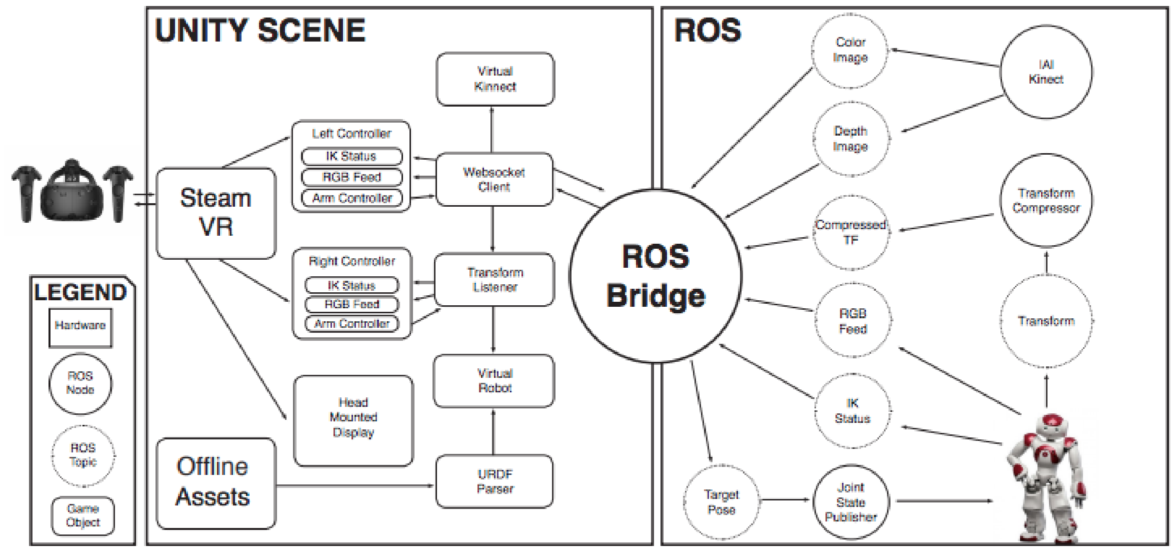

Figure 1: visual overview taken from ROS Reality paper by Whitney et al.

Since the start of this project, we have also been made aware of a few related projects in MR teleoperation, including ones that interact with robots such as the Baxter and the Movo. We have not had a chance to benchmark these projects in comparison to our own, but such a task would certainly be beneficial to conduct in the near future, along with possible integration of these various projects.

Technical Approach

The formal problem statement is this: How can we use the Microsoft Hololens to teleoperate a robot, specifically the Baxter, and allow a user to move its arms and pick up objects? And how can we do so in a way that will maximize the user experience and improve the efficiency of using the system?

Our technical approach started with ROS Reality, which established the groundwork for connecting a ROS-enabled robot to Unity by passing messages back and forth along a ROS topic. The package included a stencil for a publisher which sends the messages and a subscriber which listens to them. In this way, a Unity scene running on the HoloLens can communicate with a robot in the physical world such as the Baxter. The package also came with a C# URDF parser, which allows for the visualization of a 3D model of the Baxter that can update its proper pose in real-time. ROS Reality provided us with various commands which we could use to send messages across ROS, and its dependency ROS Reality Bridge provided us with the bulk of our networking capabilities. To avoid the overhead of needing to run multiple systems, we used the no-Ein branch of ROS Reality, which sends commands directly to the Baxter's IK Solver.

Nearly all of our work took place in Unity. We built our scenes on top of the previous ROS Reality and Holobot projects, and we performed extensive refactoring of code. In many cases, we switched out deprecated code sources with their supported counterparts. We also replaced the more basic scripts, many of which appeared to be remnants of Microsoft's HoloLens tutorials, with more robust and well-tested versions. Our approach involved a fair bit of new object creation in the Unity scenes, corresponding to the implementation of various new control features. For example, we created several buttons to perform various tasks, such as toggling the visibility of the Baxter model hologram. Each object was also associated with its own collection of C# scripts which we either wrote ourselves or imported from external libraries to define certain actions or provide desired functionality.

One of the most helpful existing technologies we used was the Microsoft Mixed Reality Toolkit. It is comprised of a large collection of scripts and components written for Unity and is intended to accelerate development on the HoloLens. It provides various interfaces to define desired features on objects, like responding to a user clicking an object, or following the user's gaze. We chose to use these scripts because of the wide range of capabilities they provided, as well as for the extensibility they offered. Unlike some of the scripts used in previous projects, which appeared to be copied straight from the online Windows tutorials, these scripts have been well-tested and are far more robust for development.

We started with using gaze to move the Baxter, as tap to place was one of the most popular methods of manipulating and repositioning objects in the HoloLens. With this method, an object follows your gaze, generally staying in the center of your field of view, and can be picked up and placed down using the native HoloLens click gesture. We created two interactable control spheres, each corresponding to an arm on the Baxter. Users could use these spheres to control each respective arm, and the cursor would raycast onto these spheres and provide certainty to the user for which object they were clicking on. Then, the sphere would follow the user's gaze and send the relative coordinates to the Baxter's corresponding arm as an end effector pose. Since the gaze feature had a fixed depth, and we wanted to allow for flexibility in the z direction, we created two buttons with which users could move the control spheres closer or farther. This is then reflected in the position of the spheres when a user selects one.

We maintained some of the features of previous projects by rendering and updating in real-time the 3D hologram of the Baxter. This was helpful at times, but in some cases it could be rather wasteful. The HoloLens, while advanced for a first-generation model, still has a fairly limited range of view. The best that we could render the hologram without experiencing significant levels of latency is low quality. To provide the user freedom in case the model would obstruct their field of view, we created another button that would allow the user to toggle the model on and off in the HoloLens view (Figure 2). Our idea was that since you are using an MR device, you can see what's going on in the real world unlike in VR, so you might as well take advantage of that. By positioning themselves in front of the real Baxter, users can use the physical robot as feedback rather than a virtual hologram.

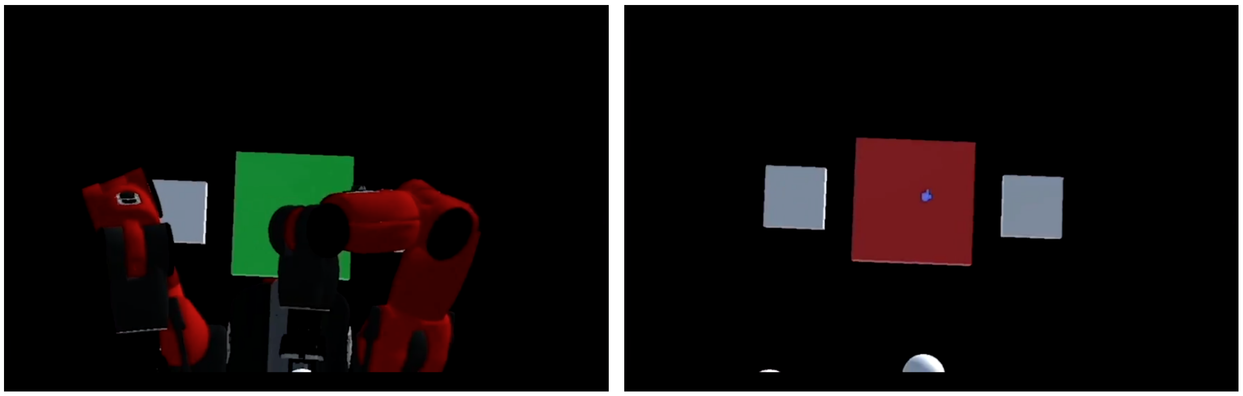
Figure 2: users can toggle the hologram of the Baxter on/off

Since we were interested in the user experience aspect of HoloLens teleoperation, we made a secondary mode of control so that we could provide versatility to the user as well as compare the two modes to see which one felt more natural (Figure 3). We called this second mode position control, in which a user uses their finger's position to control the Baxter's arm. The user clicks on a cube which corresponds to each of the Baxter's arms, and then it follows the user's finger. The cube's coordinates, as with the spheres in the gaze control, are sent to Baxter to update its position. This mode of control provided more usability in the form of depth control, since a user no longer had to click a button to increase/decrease the depth of the control object, they could just move their finger one way or another. A downside of this approach though was the fact that it was created nearly from scratch by us, and thus didn't have as nice movement animations as the other method (for instance, the position interpolation that came with the Mixed Reality Toolkit gaze control script).

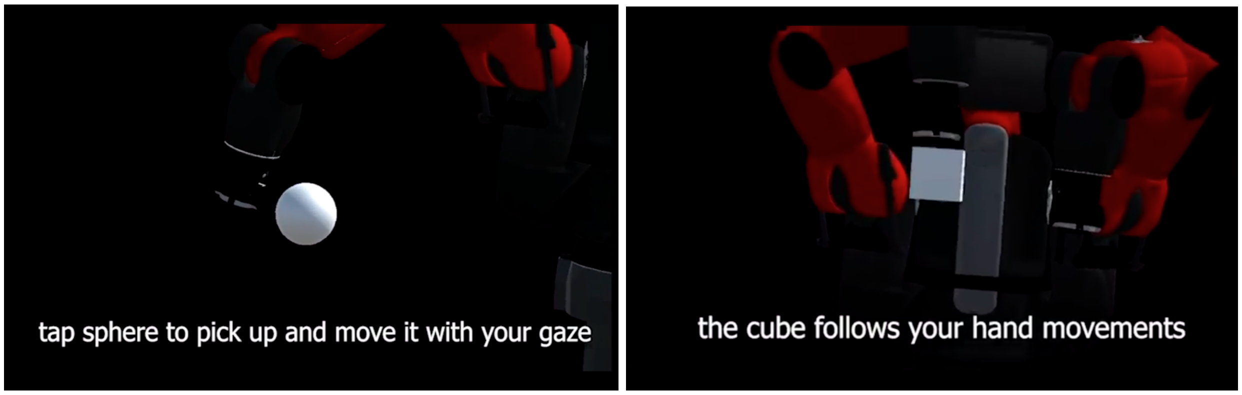
tap sphere to pick up and move it with your gaze
the cube follows your hand movements

Figure 3: demonstration of gaze v. position control modes

At any time, regardless of the means of control, a user can give the HoloLens voice commands to operate the grippers. They can say ‘open’ to open the grippers, ‘close’ to close them, and ‘grab’ to open and close them with a short delay in between. This allows the user to pick up an object and move it around. Since we created two modes of control, we also created a basic start menu which prompts a user to select the means of control that they would like to use, based on which button they selected - ‘gaze’ or ‘hand’.

Throughout the process, our technical approach adhered to standard software engineering principles. Prior to starting with our own project, we conducted some research and surveyed some pre-existing ones for examples of good and bad experiences. We also made iterative improvements to our product using feedback that we obtained from a small set of volunteers.

Evaluation

Our goal in building our technical approach was to ultimately provide users with a tool that they could use to control the robot in a reliable and intuitive way. We know that we achieved our goal if our product allows users to interact with the Baxter and perform manipulation tasks such as picking up a cup in a comfortable and efficient manner. Since our product is essentially only in its first version, we weren’t expecting to see incredibly high success rates or time benchmarks. What we were looking for was a solid baseline from which we could continue to improve our interface.

We evaluated first by performing several trials ourselves using the gaze control mode. The goal was to pick up the cup placed at the corner edge of a table in as little time as possible. A trial was marked as unsuccessful if the robot’s actions made completing the task impossible - for instance, if the cup were knocked off the table. We could have neglected success rate if we chose a more central location on the table for the cup, but since we were interested in the reliability of our system, we wanted to make sure that mistakes such as these were penalized. In a series of 7 trials, we were unsuccessful for 3 of them. These mostly happened towards the beginning, when we were acquainting ourselves with the manipulation process. Successful trials ranged with lower and upper bounds of 1 and 12 minutes, with shorter times generally corresponding to the later, more experienced trials. We also tried the position control mode, but did not like it as much. We have not yet performed any formal trials with it, but predicted that it would not be as successful as the gaze control.

We recorded a video of one of our successful trials picking up a cup with Baxter (Figure 4). We also recorded a video demoing some of the interface features of our product, which can be found here: https://www.youtube.com/watch?v=b5J6WJLmWiM.

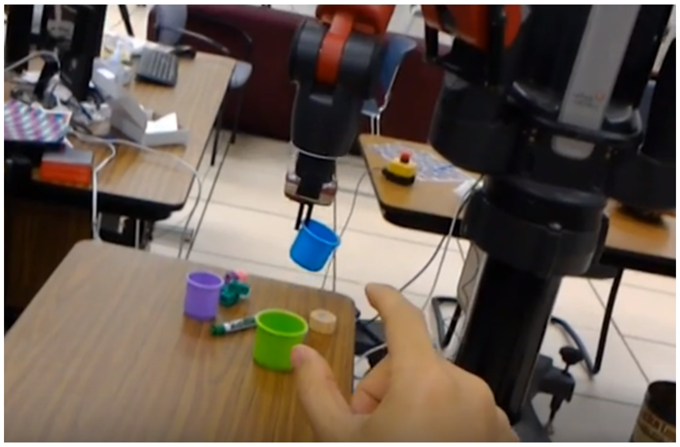

Figure 4: picking up objects with Baxter: https://www.youtube.com/watch?v=t5XodnVVjDs

We had two volunteers who had never before used our interface nor the Baxter robot try to perform the same task as described above. Because of time constraints, we did not put them through a formal series of trials, although both subjects were ultimately successful at picking up the cup. We mainly wanted to see what they thought of our product and hear their feedback.

The main concern which we also experienced in our own trials was that the high levels of latency occasionally made it difficult to reliably move the arm to a desired location. This latency would generally occur when the ROS connection was left on for too long. We tried to mitigate this by throttling when messages were sent and making sure they were only sent if the controller positions actually changed. However, we were unable to avoid it completely and suspect the root problem is a networking one.

Both volunteers preferred the gaze control mode and felt the position control mode was a bit clunky. They played around with the depth control prior to manipulation, but neither subject used the depth controls after starting the manipulation attempt - if necessary, they instead just moved their heads forward and backward to control movement in the z-direction. Both users also chose to leave the holographic model on during their interactions. Some of the feedback we received was that the project was really cool and that overall it was pretty intuitive to use. However, one of the volunteers expressed the wish to be able to calibrate the holographic robot to the physical one to make it easier to align the holographic control objects with the real world item of interest. Both agreed that the most frustrating part was the latency and the robot's occasional unresponsiveness.

Overall, we believe our project is a success. Our volunteers generally responded favorably to our interface, and everyone was ultimately successful at using the interface to manipulate an object. While the experience is still a bit clunky in parts, the baselines we collected seem quite reasonable, and we would be interested in benchmarking them with other similar products if they exist. We hope to continue improving the product with user feedback and new features.

Conclusion

The goal of this project was to allow for MR teleoperation of a robot, in our case the Baxter. We leveraged existing technologies including ROS, Unity, and the HoloLens to achieve this goal, and we produced an interface that was intuitive, easy to use, and highly extensible. It should be noted that although we focused our design on the Baxter, our code can also easily generalize to other ROS-enabled robots. While there have been many recent developments in using VR for teleoperation, the corresponding MR technologies are rather undeveloped. This project makes many attempts to improve the existing user experience of MR teleoperation with the goal of making possible future work in this area of research.

Some of the most pressing concerns include the networking and latency issue, which if fixed could significantly improve the user experience and increase success rate while reducing the required time needed for performing manipulation tasks. Other tasks to work on include incorporating calibration of the holographic model viewed in the HoloLens with the real-world Baxter. We suspect that the implementation of such a feature would also yield large improvements in the efficacy of the system.

The problem of how to provide the optimal experience for users performing MR teleoperation tasks is still up in the air. Future work in this project would involve conducting more extensive user feedback and making incremental improvements. This would likely require the introduction of yet another (or several) additional control modes, and would culminate in a user study that compares the various modes for effectiveness and overall experience using both subjective and objective measurements.

Lastly, there exist several other projects in this area with a host of different features, and it would be nice if we could integrate the best parts of each project. An example would be the Movo MR teleoperation project, which has features catered towards the robot's mobile nature (in contrast to a stationary robot such as the Baxter) such as setting waypoints and planning paths. We envision a navigation bar with a series of tools for interacting with all sorts of robots in meaningful ways. As augmented reality technologies become more pervasive in our lives, the challenge of designing good interfaces for these tasks will only increase in importance.

References

[1]
David Whitney, Eric Rosen, Elizabeth Phillips, George D. Konidaris, and Stefanie Tellex. Comparing Robot Grasping Teleoperation Across Desktop and Virtual Reality with ROS Reality. In *Robotics Research: the 17th Annual Symposium*, December 2017.

[2]
Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. arXiv preprint arXiv:1710.04615, 2017.

[3]
David Whitney, Eric Rosen, Daniel Ullman, Elizabeth Phillips, Stefanie Tellex. ROS Reality: A Virtual Reality Framework Using Consumer-Grade Hardware for ROS-Enabled Robots.