Gary Chien
Code for my project can be found at https://github.com/h2r/Holobot/tree/holocontrol_movo.

Abstract

Mixed reality technology like the HoloLens is new enough that it is still relatively unexplored in the field of robotics research, so it is worthwhile to explore how it can be used in novel ways, such as teleoperating the mobile base of a robot. The main limitation of related works that use augmented reality to teleoperate robots is that the research is usually limited by the hardware from time before headsets like the HoloLens existed - for instance, augmented reality was typically constrained to a two-dimensional screen, instead of a 3D hologram overlaid over the real world. My project's goal was to use a HoloLens to place holographic waypoints on the ground to create a path for a Kinova Movo robot to navigate through. There were three main components in this project: a ROS component to perform localization and navigation, a Unity component to provide a user interface to place waypoints for the Movo to move through, and a networking component to enable communication between the ROS and Unity components. Waypoint navigation was achieved by first calibrating between the ROS and Unity coordinate spaces, placing spherical waypoints on the floor using spatial mapping, using ROS subscribers and publishers to pass waypoint coordinates from Unity to ROS, and then using gmapper to move the Movo's base to those coordinates. This HoloLens app contributes a novel way to teloperate the Movo base without using a traditional joystick or computer interface. This is also, to a lesser extent, a peek at the potential of using mixed reality to provide motion intent communication to the human operator of the Movo, in that the waypoints clearly mark where the Movo plans to move to.

Introduction

Mixed reality headsets like the HoloLens have the potential to contribute a lot to the field of robotics, similar to how virtual reality technology has inspired novel and creative robotics research in recent years. Unlike virtual reality, the HoloLens enables users to see the virtual and real world simultaneously. This is a desirable feature in robotics, where safety is important - with the HoloLens, the user can be aware of the movements of the real world robot at all times and get out of the way of potential danger. Because the HoloLens is still relatively new, not a lot of research has been conducted using this headset. Its powerful spatial mapping and voice/hand gesture interaction capabilities can become a powerful tool to help human interact with robots, and using it to teleoperate a mobile robot is a great application of this concept. The vast majority of robot teleoperation is done using joysticks, which can be tedious to use because it only supports a limited set of inputs: up, down, left, right, and twist. In order to fully control all moving parts of a robot, mode switching is often required, which, in the Movo's case, involves pressing various combinations of buttons to switch joystick control to different parts of the robot. This quickly becomes tedious when the robot must be teleoperated to perform complex tasks. With mixed reality, the dimensionality of input becomes much larger. Gaze tracking and hand gestures allow for quick tap-to-place actions well-suited to teleoperation around the surrounding environment, and voice commands allow for quick and intuitive mode switching. This project will be focused on holographic waypoint navigation of the Movo's mobile base. It's important to note

that this task by itself is rather simple and easy to perform using a traditional joystick. Thus, the biggest contribution of this project is not the specific feature of waypoint navigation, but the foundation of a ROS/Unity mixed-reality interface for Movo teleoperation that can be continually expanded and improved upon with additional features in the future.

Before the HoloLens, augmented reality research in robotics was typically constrained to computer displays, and lacked the freedom of movement that mixed reality headsets grant users. Thus, while there is precedence for using augmented reality to interact with robots, they are often limited in terms of hardware. There has been little research that takes full advantage of the HoloLens's capabilities to teleoperate robots, simply because the HoloLens is such a recent invention. One recent study similar to my own project was Rosen et al.'s study done at Brown, which investigated the use of the HoloLens to communicate motion intent using holograms. This study differed from many related works in that they focused on the specific strengths of the HoloLens compared to other state-of-the-art methods of robot interaction. This project, too, will take full advantage of the features of the HoloLens to create a mixed-reality app that is both powerful and intuitive.

Unlike previous approaches to mixed reality or augmented reality robot teleoperation, my approach to teleoperating the Movo is to tap-and-place holographic waypoints (small spheres) onto the floor using a HoloLens. This was achieved by 1) building a Unity app to act as the HoloLens interface allowing the user to interact with and send commands to the Movo, 2) writing ROS scripts to enable navigation in known maps, and 3) using ROS subscribers/publishers and rosbridge to enable communication between the Unity app and ROS.

The HoloLens interface is simple to use. On startup, the HoloLens uses spatial mapping to build a mesh of the user's surroundings. Once this is accomplished, the user can use a combination of gaze and hand taps to place a 3D hologram of the Movo on the floor, such that it aligns with the real-world Movo. Unity then requests the Movo's pose from ROS and calculates the transform between Unity's coordinate space and the ROS coordinate space. The user then places waypoints on the floor using tap gestures. The coordinates of these waypoints are sent to ROS, which then instructs the Movo to navigate to these waypoints via gmapping. As the Movo moves, the hologram of the Movo updates its position to follow the real Movo. Once this navigation is complete, the user can place another set of waypoints on the floor, and the process repeats.

The HoloLens app was able to successfully teleoperate the Movo to specified waypoints placed on the floor, with one caveat: calibration of the Unity and ROS coordinate spaces only work when the orientation between those two spaces are aligned. In other words, the user must be standing directly facing the Movo before starting the HoloLens app. Besides this limitation, the interface successfully allows users to instruct the Movo to move to various locations with simple taps.

Related Work

The integration of augmented reality into robotics is not a novel idea. For instance, Pessaux et al. used AR to project a visualization of the interior of a patient's body onto the patient's skin, in order to aid robotic-assisted surgery (Pessaux et al., 2015). Kuriya et al. used AR to enable robot navigation. However, they relied on using AR tags to give the robot

commands, and AR interface could only be viewed on a computer screen (Kuriya et al., 2015). In 2016, Michalos et al. provided evidence that AR can be used to enhance safety and efficiency of shared human-robot workspaces. Their study, however, required the use of an Android tablet to interact with holograms (such as enabling or disabling specific holograms), instead of running entirely on an AR headset (Michalos et al., 2016). Rosen et al. conducted a study that investigated the use of a HoloLens to communicate motion intent of a robot arm (Rosen et al., 2018). They provided evidence that the HoloLens is much better at visualizing the planned motion of a robot than other state-of-the-art methods. With their study, they showed that the HoloLens, when used to its full extent, is a powerful tool that can be employed in robotics. I hope to reinforce this claim with my own project. Unlike past studies that rely on AR interfaces using 2D displays or more primitive controls to interact with robots, my project will take advantage of the features of the HoloLens, like spatial mapping and hand gesture recognition.

Technical Approach

The goal of this project was to create a mixed-reality interface to teleoperate the Movo base by placing holographic waypoints on the floor to define a path. Achieving this goal involved solving a series of smaller problems: 1) Calibrate between the ROS and Unity coordinate spaces; 2) Place holographic waypoints on the floor and send their coordinates to ROS; 3) Instruct the Movo to move to these waypoints; and 4) Continually update the pose of the holographic Movo to match the pose of the real Movo. Each of these steps will be described in their own paragraph.

Calibrating between the ROS and Unity coordinate spaces is simple in concept: simply find the cartesian offset between the Unity origin and the ROS origin, and then rotate the Unity coordinate system to match the Movo's initial rotation. When the Unity app starts up, the user uses a combination of spatial mapping and gaze to align a Movo hologram with the real Movo, and then uses a tap gesture to fix the hologram in place. Meanwhile, the TFListener script sets up a series of publishers ("unity_waypoint_pub", "movo_state_request", "movo_pose_request") and subscribes to the following topics published by ROS: "ros_movo_state_pub" and "ros_movo_pose_pub". The "movo_pose_request" topic enables Unity to request from ROS the current pose (x, y, z, theta) of the Movo. With the Movo's pose known, the ROS coordinates of each waypoint is computed as follows:

1. Get the local space position of the waypoint relative to the Movo GameObject (for example, if it is 1m in front of the Movo and 1m to the left, its local space position would be (x,y) = (1, 1)).
2. Get the rotation of the Movo in ROS space, and get the rotational offset of the Movo hologram's pose relative to Unity space. Add these two angles together to get some θ, which represents how much the Unity coordinate space needs to be rotated to match the ROS coordinate space.
3. Set $x' = x\cos(\theta) - y\sin(\theta)$, $y' = x\sin(\theta) + y\cos(\theta)$. This simply multiplies the waypoint's Unity coordinates by the standard 2D rotation matrix.
4. Increment x' and y' by the Movo's initial position in ROS. This theoretically yields the coordinate of the waypoint in the ROS coordinate space.

Unfortunately, there appears to be a bug in rotational calibration that has yet to be solved: starting the HoloLens app from arbitrary positions often results in the Movo moving to incorrect destinations. For the project, rotational calibration was commented out from the code (see the Waypoint class in Utils.cs), which causes the slight inconvenience of forcing the user to stand directly in front of the Movo upon starting up the HoloLens app, but enables waypoint navigation to work properly.

Waypoints are placed in much the same way as the Movo's hologram was placed at startup: the waypoint spheres are raycasted onto the spatial mapping mesh of the floor, following the user's gaze until a tap gesture is detected. Upon tapping, the waypoint is fixed to the floor, and its ROS coordinates are calculated and visually displayed over the waypoint. At this point, the waypoints' coordinates are published to "unity_waypoint_pub" in the form of a string of pairs of numbers separated by semicolons. Note that waypoint placement is only enabled when the Movo is in "standby" mode, which Unity verifies by querying ROS for the Movo's current state. For the project, exactly two waypoints can be placed a time, before switching to navigation mode.

Instructing the Movo to move to the waypoint coordinates is done with gmapping, with an actionlib client (see ROS_scripts/move2goal.py). Upon starting up the Movo, gmapping is launched, with a pre-scanned map of the room loaded. The Movo is then aligned to the map via the RViz interface. The move2goal.py script is subscribed to "unity_waypoint_pub"; as soon as it detects a published set of waypoint coordinates, it parses the string of numbers and uses actionlib to send navigation goals to the "movo_move_base" topic. Gmapping takes over from there, planning and executing a path for the Movo to move to each waypoint goal.

Making the Movo's hologram follow the real Movo is a simple process: Unity continually queries ROS for the Movo's current pose (x, y, z, theta), adjusts it to Unity space, and updates the Movo hologram's position within Unity. Thus, after the calibration process, the Movo's hologram will continually follow the real Movo.

Evaluation

My goal of creating an intuitive mixed-reality interface to teleoperate the Movo's mobile base was largely successful. With just gaze and tap gestures, the user is able to use waypoints to instruct the Movo to move anywhere it's capable of going within its internal map of the room. However, this only works if the ROS and Unity coordinate spaces are not rotated relative to one another. Accommodating for this rotational offset in the calibration stage contains yet-unsolved bugs that causes the waypoints to have incorrect ROS coordinates; there is likely something wrong in the math used to calculate the rotational transform from Unity coordinates to ROS coordinates. However, removing rotation from the equation entirely and running the HoloLens app while directly facing the Movo on startup has temporarily solved this issue. It is difficult to give a quantitative evaluation of waypoint navigation, except to say that the Movo has a 100% success rate in navigating to waypoints, provided that each waypoint is placed in a legal location in the Movo's map (in other words, it's physically possible for the Movo to move to the waypoint). One drawback is that navigation is slow. Upon reaching each waypoint, the Movo will pause and not move for several seconds, before moving on to the next waypoint. This is because it takes time for gmapping to generate a new plan for the Movo to navigate to the next

goal pose. One potential future fix for this would be to find a faster alternative to gmapping to navigate to waypoints - perhaps some form of odometry-based rather than map-based navigation. Another interesting behavior is that, upon reaching each waypoint, the Movo will rotate itself until it is aligned with the x-axis of its internal map. This happens because, currently, the Movo is only sent (x, y) coordinates as its goal, and its goal orientation is set to a predefined value of 0. A demo of mixed-reality waypoint navigation can be found here: https://drive.google.com/open?id=1qg8nl2i1fgtA5Tbeez4DQPXr-fBm2k2U

Conclusion

My goal of building a mixed-reality app to teleoperate a Movo base using waypoint navigation was achieved, although it has much room for improvement. This project explored the potential of using the HoloLens to interact with robots, unlike past research in which AR was limited by the technology of the time. The app was designed to be simple and intuitive: drag a hologram of the Movo over the real Movo to calibrate ROS/Unity coordinate spaces, place waypoints on the ground using spatial mapping and tap gestures, and instruct the Movo to move to these waypoints. Although the performance is somewhat inefficient due to long pauses between waypoints and unnecessary rotations upon reaching waypoints, waypoint navigation using the HoloLens works.

There are, however, small issues that need to be fixed in the near future. The calibration process needs to be reworked so that the transform between ROS and Unity coordinate spaces can be properly defined if they are rotated relative to each other. Fixing this should simply be a matter of taking another look at the equations used to transform Unity coordinates to ROS coordinates and ensuring that the coordinate rotation code works properly. Another issue that should be fixed is preventing the Movo from unnecessarily aligning itself to the x-axis of its internal map whenever it reaches a waypoint. This occurs because its goal pose's rotation is always set to a constant 0. This issue can be fixed by finding the directional vector between the Movo's start position and the goal position, and using this vector to define the goal pose's rotation. Because the Movo moves to each subsequent waypoint in straight lines (no detours), it should already be at its goal rotation by the time it reaches the waypoint, and will not have to rotate much to reach its goal pose. This issue could actually be addressed in a different way: by getting rid of gmapper entirely and replacing it with a simpler navigation process that does not rely on planning paths within an internal map. Doing so would reduce the pauses that the Movo takes between waypoints to plan new paths. One final modification to improve the interface even more is to give the user the ability to specify the number of waypoints they want to place at a time, perhaps by using voice commands to switch between modes.

These fixes would greatly streamline the waypoint teleoperation process. However, there is potential to expand beyond waypoints. For example, the robot's planned path could be visualized by a trail of multiple robot holograms (similar to the Baxter arm in Rosen et al.'s study) that can represent velocity and rotation as well as position over time. With this interface, the user could have full control of position, rotation, and speed of the Movo's mobile base through time, by manipulating the poses of individual holographic Movos in the motion trail, as well as the density of the trail itself (where a higher density of Movo holograms indicates slower speed at that point in time, and a lower density indicates higher speed). It would also be nice to

integrate Ein into the project, which would enable the user to control all moving parts of the Movo (such as its tilting head, its telescoping base, and even its arms) by interacting with various holographic controls.

In the long term, I can see this project becoming the foundation for a general-use mixed-reality robot teleoperation interface, where any robot can be controlled by this app as long as the robot has a URDF file and is capable of tracking its joint states. After all, all the HoloLens does is communicate with ROS by subscribing and publishing to ROS topics: if anything is achievable with ROS, it is achievable through a HoloLens interface. Perhaps multiple HoloLenses could be used together in a shared augmented reality space, allowing multiple people to teleoperate multiple unique robots however they see fit. This could have real-world impacts as well. For example, robot arms have been shown to immensely improve the lives of wheelchair-confined disabled people. However, these arms can only be controlled with a joystick, so they are not accessible to people who are incapable of using joysticks (there has recently been investigations into the user of a brain-computer interface to control robot arms with the brain, but they are either highly invasive or not accurate enough for practical use). A mixed-reality robot teleoperation interface could potentially allow even quadriplegics to control a robot arm, with a combination of gaze tracking and voice commands. The HoloLens is simply too powerful a tool to not be taken advantage of by the robotics community, and it will likely become increasingly integrated into robotics as the technology continues to improve.

References

Michalos, G., Karagiannis, P., Makris, S., Tokçalar, Ö. and Chryssolouris, G. (2016). Augmented Reality (AR) Applications for Supporting Human-robot Interactive Cooperation. *Procedia CIRP*, 41, pp.370-375.

P. Pessaux, M. Diana, L. Soler, T. Piardi, D. Mutter, and J. Marescaux, “Towards cybernetic surgery: robotic and augmented reality-assisted liver segmentectomy,” *Langenbecks Archives of Surgery*, vol. 400, no. 3, pp. 381–385, 2014.

R. Kuriya, T. Tsujimura, and K. Izumi, “Augmented reality robot navigation using infrared marker,” *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2015.

Rosen, E., Whitney, D., Phillips, E., Chien, G., Tompkin, J., Konidaris, G. and Tellex, S. (2018). *Communicating Robot Arm Motion Intent Through Mixed Reality Head-mounted Displays*. [online] Arxiv.org. Available at: https://arxiv.org/abs/1708.03655 [Accessed 16 May 2018].