A VR Teleoperation Suite with Manipulation Assist

Christian Barentine,[1] Andrew McNay,[1], Ryan Pfaffenbichler,[1] Addyson Smith,[1] Eric Rosen, [2], and Elizabeth Phillips[3]
[Christian.Barentine,C21Andrew.McNay,
C22Ryan.Pfaffenbichler,C21Addyson.Smith]@afacademy.af.edu,eric_rosen@brown.edu,ephill3@gmu.edu
[1]United States Air Force Academy, [2]Brown University, [3] George Mason University
Air Force Academy, Colorado

ABSTRACT

Advances in the capabilities of technologies like virtual reality (VR) and their rapid proliferation at consumer price points, have made it much easier to integrate them into existing robotic frameworks. VR interfaces are promising for robotics for several reasons, including that they may be suitable for resolving many of the human performance issues associated with traditional robot teleoperation interfaces used for robot manipulation. In this systems-focused paper, we introduce and document the development of a VR-based robot control paradigm with *manipulation assist* control algorithm, which allows human operators to specify larger manipulation goals while leaving the low-level details of positioning, manipulation and grasping to the robot itself. For the community, we also describe system design challenges to our progress thus far.

CCS CONCEPTS

• **Human-centered computing → Virtual reality**.

KEYWORDS

Virtual reality, robot, predictive control

ACM Reference Format:
Christian Barentine, Andrew McNay, Ryan Pfaffenbichler, Addyson Smith, Eric Rosen, & Elizabeth Phillips. 2021. A VR Teleoperation Suite with Manipulation Assist. In *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction (HRI'21 Companion), March 8-11, 2021, Boulder, CO, USA.* ACM, NY, NY, USA, 5 pages. https://doi.org/10.1145/3434074.3447210

1 INTRODUCTION AND PRIOR WORK

Robots are capable of fast, repetitive, and precise manipulation of objects both large and small, but are often limited by perception, planning, and control for manipulation tasks that require flexibility and re-planning. As a result, taking advantage of robot strengths while mitigating their drawbacks is a main appeal of teleoperation, or the direct remote control of robots by human operator(s). However, in order for humans to teleoperate robots well, they need high-fidelity control over the robot's actuators and an accurate and rich visualization of the robot's environment.

HRI '21 Companion, March 8–11, 2021, Boulder, CO, USA

ACM ISBN 978-1-4503-8290-8/21/03…$15.00
https://doi.org/10.1145/3434074.3447210

Previous teleoperation schemes have suffered from a disconnect between the presentation of the robot's environment to its operator, and the reality of the environment itself. Simply put, robots typically operate in 3-dimensional environments, while traditional teleoperation schemes present that environment to humans using 2-dimensional methods, generally relying on computer monitors to display sensor data, and joysticks or keyboards to actuate the robot. These 2-D teleoperation control schemes are cumbersome and workload-intensive for human operators [2, 10, 11]. As such, there is a strong need to investigate and develop new systems for interfacing with and teleoperating robotic systems. Previous research has shown that a Virtual Reality interface can be an effective means to teleoperate robots and more usable and less workload intensive than traditional keyboard and monitor interfaces [4, 6, 11].

Further, advances in the capabilities of virtual technologies, as well as their rapid proliferation at consumer price points, have made it much easier to integrate them into existing robotic frameworks. For instance, prior work showed that using a virtual reality (VR) interface which allowed users to teleoperate a Baxter robot's manipulators using waypoint-like control (called positional control), was faster and more accurate for both gross and fine motor manipulation tasks than a VR interface that mimicked directly "clicking and dragging" a Baxter's manipulators (called trajectory control) [3]. As a result, we are in the process of building upon the waypoint-like interface by creating a VR robot control paradigm with *manipulation assist*, a control assist algorithm, which allows a human operator to specify larger manipulation goals while leaving the details of positioning and manipulation to the robot itself. The purpose of this paper is to document the progress and report the system details of the manipulation assist VR system. We also describe for the community system design challenges to our progress thus far.

The intersection of robotics and virtual technologies like augmented and virtual reality (AR/VR) is a rapidly growing field. Recently, the human-robot interaction (HRI) community has begun to organize around research topics and applications that intersect these technologies [13, 14]. With the rapid proliferation of consumer-grade VR systems at accessible price points, smaller form factors, and with better on-board computing, there has been increasing interest in the use of these systems for many robotic applications. This proliferation has also allowed researchers to incorporate consumer-grade AR/VR in human-robot interactions in meaningful ways [5, 7, 8, 12].

For instance, a study by Whitney et al. [11] showed that a VR interface allowed non-expert users to teleoperate a robot to complete a number of dexterous manipulation tasks faster and with lower cognitive workload than than traditional 2-D keyboard and monitor

interfaces. The researchers also found that participants rated the VR interface more usable and assigned higher satisfaction scores to the VR interface than the keyboard and monitor interface. Lipton [4] conducted an informal user evaluation of a VR-robot teleoperation paradigm that employed a homunculus control interface. The homunculus model of the robot virtually embedded users in a "control room" inside the robot's "mind." The researchers asked users to control the robot to engage in a number of manufacturing and pick and place tasks with objects of different shapes and compliance. Via the homunculus VR model, users successfully picked up each item, transferred that item between robot hands, and finally placed each item in a bin. Because several researchers have demonstrated the promise of using consumer grade VR hardware for teleoperated control of robots, there is value in continuing to develop and test VR-based interfaces to improve human control paradigms of robots and ultimately human-robot interactions.

2 SYSTEM OVERVIEW

Due to the nature of teleoperation, our system has significant hardware and software components that need to operate in concert for successful execution documented in the following sections.

2.1 Hardware

The physical components of the system are:

- A Rethink Baxter Robot
- An Oculus Quest VR headset and controllers
- A WLAN router
- A network switch
- Multiple printed ArUco tags
- At least one Logitech USB Webcam
- External laptop running computationally intensive packages

Figure 1: The Baxter robot used for the TagUp system.

The Rethink Baxter robot is a two armed robot, commonly used for research and industrial tasks such as pick and place. In our system, we have one Baxter for the user to operate. The Baxter itself is connected to the network switch with an ethernet cable, which is connected to a laptop. The laptop serves as an access point for Baxter, and allows us to run additional software in a ROS environment, such as ArUco tag recognition. In addition to the ethernet connection from the switch, the laptop is connected wirelessly to the WLAN router. The router connects wirelessly to the Oculus Quest VR Headset, enabling wireless control of the robot. The printed ArUco tags are affixed to objects that Baxter should be able to manipulate. While Baxter comes with a camera in each wrist near its end effectors, a stationary external usb webcam is used to provide stable object tracking. This camera is placed such that it has a good view of the external workspace.

2.2 Software

The software components of the system are:

- ROS Indigo
- Ubuntu 14.04
- TagUp VR Control Program
- Unity Game Engine
- ZeroMQ
- ArUco Tag recognition package
- MoveIt Inverse Kinematics solver

Both Baxter and the laptop run ROS Indigo, with the laptop running Indigo on top of Ubuntu 14.04. The laptop is responsible for running additional ROS services, including a MoveIt Inverse Kinematics server, the ArUco Tag package, and a custom bridge between the headset and ROS. The Oculus Quest Headset runs on a proprietary version of the android operating system made by Oculus. It has an internal computer capable of running VR applications without an attached PC, and this is where the VR control program itself is run. We use the Unity Game engine to make the VR control program, and upload it to the headset. Several packages exist to allow other applications to interact with ROS nodes. However, our attempts to use the most common of these, ROS#, was unsuccessful. Instead, we use a lightweight messaging library called ZeroMQ to pass messages in real time between the ROS nodes running on the laptop and the VR control program on the headset. One of the most important streams of information coming from ROS is the pose of all identified ArUco Tags, as provided by the ArUco Tag ROS package. This package is capable of identifying tags from any number of cameras, which are then sent to the control program for rendering of the object the tag is attached to, and execution of the manipulation assist algorithms.

2.3 Object Pose Tracking

For the robot to assist with object manipulation, it must first be able to recognize what the object is and where in 3D space the object is relative to the robot itself. To meet this requirement, our system uses fiducial tags. Fiducial tags are high contrast patterns, similar in concept to a QR or bar code, though with more data redundancy built in. A tag recognition system is made up of printed tags affixed to objects or navigation points, and at least one camera connected to a computer running tag recognition software appropriate for the family of tag being detected. Each tag has an ID which the recognition software decodes in addition to the tag pose relative to the detecting camera. Our system uses a dictionary of tag IDs to map a tag's camera relative pose to its associated virtual object. To perform this mapping we use tag 0 as a special tag denoting the "origin" in both the physical and virtual environment. Cameras are positioned so they can see the origin tag, allowing us to work

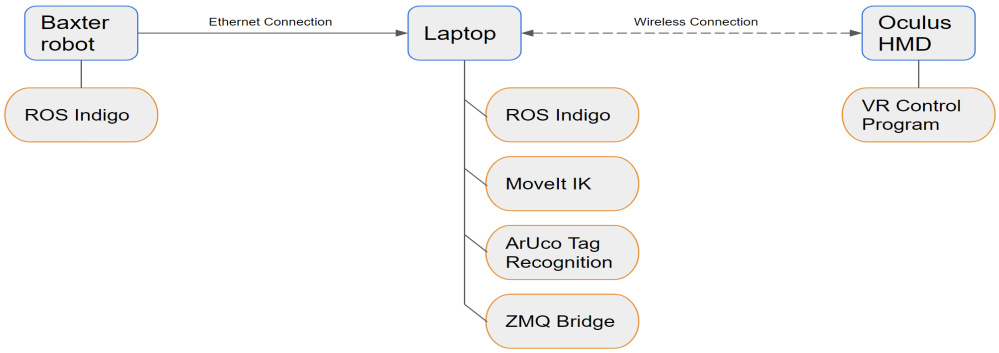

Figure 2: A high level overview of the system, with both hardware (square blue outline) and software (capsule orange outline) components shown.

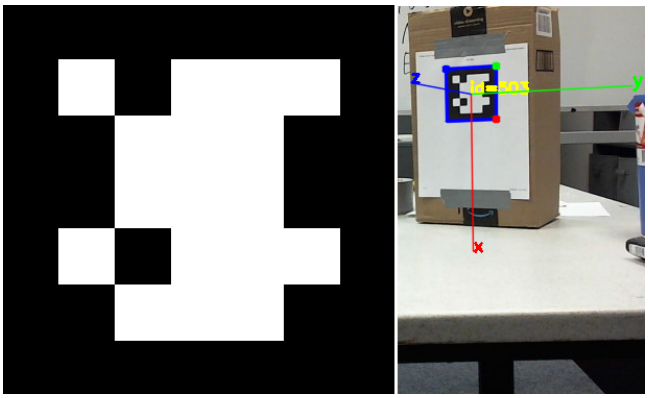

Figure 3: An example of ArUco tag number 503 (left) being identified via a usb webcam (right).

backwards to get the tag's position relative to the origin, and by extension place all tags into the same frame of reference, regardless of the camera that detected them.

2.4 Manipulation assist algorithm

The process of a manipulation assist is conceptually similar to following a set of waypoints. When a valid assist is requested, the robot loads a list of waypoints, as well as desired states, and performs them in sequence. Once complete, control is returned to the operator. The two key decisions that the algorithm must make are: (1) What assists are valid? and (2) Which of the valid assists does the operator desire the robot to perform?

Because an assist is dependent on the context of the robot, the assist controller uses a virtual context manager. The job of the context manager is to store a representation of the robot's state, to answer queries about the robot's state, and to update that state as appropriate. Furthermore, context is broken down into groups based on logical parts of the robot. Currently, the 4 available groups are LeftArm, RightArm, Robot, and Global. So, the context manager might record that the right arm is gripping an object, while the left arm is not.

First, an object must be identified as an assist target. This associates a list of possible assists, the waypoints to perform each assist, the conditions required for the assist to be valid, and the resulting changes in the context of the robot. When running the control program, there is an invisible volume designated around each of the operator's Oculus hand controllers. The volume keeps track of any assist targets within it, and keeps them as a list of potential targets for the appropriate context group. For example, if the object is within the volume attached to the right hand controller, it is stored as a potential target for the right arm context group.

Next, the user presses a button on the hand controller assigned to the assist control, indicating that they would like an assist. What happens next depends on both the context and the current permission paradigm. Two permission paradigms for manipulation assist are currently under development and will be the target of future user evaluation studies. They include *exception* and *permission* style of manipulation assist management. In a management by *exception* style, the robot will automatically apply the low level task it thinks the operator desires using the manipulation assist system, unless interrupted by the operator. This contrasts with the management by *permission* style, where the robot will never perform any low level tasks automatically unless explicitly directed to by the operator.

The controller looks through all of the possible assists for each potential target, and checks their requirements against the context. For example, one of the requirements for picking up an object is that the gripper is currently empty. All assists that have all of their requirements satisfied are added to a list of valid assists. After compiling the valid assists, the controller checks to see what it's permission mode is. If it is by exception, the first valid assist is selected. If it is by permission, then a menu containing all valid assists is displayed to the operator, and the controller waits for a choice to be made. Once an assist has been selected, the controller stops listening for user input, and instead sends the waypoints/gripper state to the robot as goal states, waiting for the robot to signal that it has reached the given waypoint before sending the next one. Once all waypoints have been reached, control is returned to the operator.

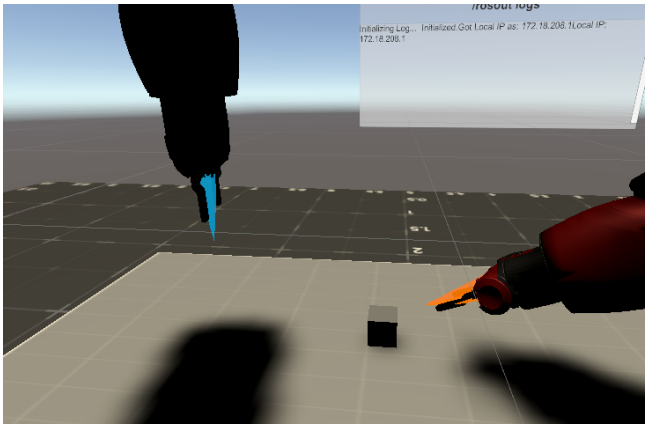

Figure 4: What the user sees as they control the Baxter. Note that the user has taken an egocentric position.

2.5 VR environment

Currently, most of the VR environment is hard coded. A model of Baxter sits at the center of the space, with a grey box representing the real Baxter's work table. Assist targets are stored in a hard coded list, with an ID assigned to each. When one of the cameras in the work space detects a tag, the ID is looked up, and the corresponding object is placed in the scene, relative to the camera that detected it. If multiple cameras detect the same object, then an average of the positions is assigned as the object's position. These objects are visually represented by low polygon models of the corresponding physical objects. If a tag has not been detected after a certain timeout value, the virtual object is disabled until tracking is re-established.

3 SYSTEM DESIGN CONSIDERATIONS AND CHALLENGES

During implementation of this system, we encountered several roadblocks and limitations. Primary among them was difficulty in presenting the robot's environment to the user. Point clouds are a common technique, but we felt that their high bandwidth cost and low fidelity made it worthwhile to explore other options. Using a tag based system proved to be a challenge in its own right, because of issues with integration and the tags themselves. First, there has been much research directed at making fiducial tags that are small, have large dictionaries, and can still be read reliably from great distances. Two such implementations that we attempted to use for our system were LFTag[9] and Stable Tag[1]. However, after much effort we were unable to get the tags to work with ROS-Indigo, the version of ROS required by the Baxter robot. We found that ArUco tags would work with our version of ROS, but that the tags themselves had severe drawbacks.

ArUco tags required large sizes to be tracked from distances of more than 3 feet, limiting the placement of the tags to larger objects in the YCB set. Furthermore, ArUco tags are not robust to even partial occlusion. Tags are lost immediately upon an object obscuring any part of the tag, and can only be rotated by small amounts relative to the camera. The challenge this presents for our system is that assists often obscure tags, either directly through the grippers, or indirectly by manipulating the object into a position where the tag is at an angle. The result of this is that the virtual display is frequently out of sync with the objects real positions.

4 FUTURE DEVELOPMENT

As we work to prepare this system to run HRI Experiments, our focus is on ensuring a robust, comfortable experience for the operator. Towards this goal, we are working to add the camera feeds, improve our object tracking ability, and add tutorials for controlling the robot using the VR system to improve the performance of all levels of operators. One possible approach to improving our object tracking is to use AI object pose estimation. We initially wanted to use tags to track objects because we were not interested in researching AI per say, but instead wanted to focus on how we could improve the operating efficiency of teleoperated robotic systems. Our experience working with tags thus far has been challenging. If we can find a way to introduce more robust object tracking, regardless of method, we will look to implement it.

The uses of this platform are also highly restricted due to hard-coding. To address these issues, our approach is to have TagUp look at specific folders when starting up, to search for object and robot definitions. Robots already have the widely used URDF format that could be provided, though we will probably have to make our own object information format, so that all relevant information, such as assist waypoints, are captured for each object. Once these formats are in place, all that TagUp would need to work with any robot or object would be for the appropriate information files to be placed in their respective folders. Our eventual goal is to have TagUp be as plug-and-play as possible, and dynamically loading robots and objects would be a huge step in this direction.

Due to restrictions to conducting in-person studies as a result of the global spread of COVID-19, we have placed emphasis on incorporating a tutorial in the VR environment which would demonstrate basic operation of the VR system for controlling Baxter, such that no researchers would need to be present or co-located when a participant in a research study is using the system. We gathered this idea from completing the VR Oculus headset tutorial provided with the headset. The tutorial had users perform many different types of tasks such as picking up objects, throwing objects, shooting guns, and moving the head and body around the "safety circle." We anticipate the tutorial will support future user evaluations of the system.

5 CONCLUSIONS

VR interfaces are promising for allowing users to more effectively teleoperate robots, especially for manipulation tasks. In this paper, we presented system details for a VR-based robot control system with *manipulation assist*, which allows users to specify high level manipulation goals while leaving the low-level details of position and manipulating to the robot itself. For the community, we detail the individual components of the system (hardware, software, and algorithm details) and progress and challenges to its development to date, as well as planned future development details.

6 ACKNOWLEDGMENTS

This research was funded by AFOSR Grant 16RT0881.

REFERENCES

[1] Burak Benligiray, Cihan Topal, and Cuneyt Akinlar. 2017. STag: A Stable Fiducial Marker System. *CoRR* abs/1707.06292 (2017). arXiv:1707.06292 http://arxiv.org/abs/1707.06292

[2] Jennifer L Burke, Robin R Murphy, Michael D Coovert, and Dawn L Riddle. 2004. Moonlight in Miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise. *Human–Computer Interaction* 19, 1-2 (2004), 85–116.

[3] Rebecca Hetrick, Nicholas Amerson, Boyoung Kim, Eric Rosen, Ewart J de Visser, and Elizabeth Phillips. 2020. Comparing Virtual Reality Interfaces for the Teleoperation of Robots. In *2020 Systems and Information Engineering Design Symposium (SIEDS)*. IEEE, 1–7.

[4] Jeffrey I Lipton, Aidan J Fay, and Daniela Rus. 2018. Baxter's Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing. *IEEE Robotics and Automation Letters* 3, 1 (2018), 179–186.

[5] E. Rosen, D. Whitney, E. Phillips, G. Chien, J. Tompkin, G. Konidaris, and S. Tellex. 2017. Communicating robot arm motion intent through mixed reality head-mounted displays. In *International Symposium On Robotics Research*.

[6] Eric Rosen, David Whitney, Elizabeth Phillips, Daniel Ullman, and Stefanie Tellex. 2018. Testing robot teleoperation using a virtual reality interface with ROS reality. In *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*.

[7] Daniel Szafir, Bilge Mutlu, and Terrence Fong. 2014. Communication of intent in assistive free flyers. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. ACM, 358–365.

[8] Daniel Szafir, Bilge Mutlu, and Terry Fong. 2015. Communicating directionality in flying robots. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*. 19–26.

[9] Ben Wang. 2020. LFTag: A Scalable Visual Fiducial System with Low Spatial Frequency. arXiv:2006.00842 [cs.CV]

[10] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex. 2017. Comparing Robot Grasping Teleoperation across Desktop and Virtual Reality with ROS Reality. In *International Symposium on Robotics Research*.

[11] David Whitney, Eric Rosen, Elizabeth Phillips, George Konidaris, and Stefanie Tellex. 2020. Comparing robot grasping teleoperation across desktop and virtual reality with ROS reality. In *Robotics Research*. Springer, 335–350.

[12] Tom Williams, Daniel Szafir, and Tathagata Chakraborti. 2019. The reality-virtuality interaction cube. *VAM-HRI* (2019).

[13] Tom Williams, Daniel Szafir, Tathagata Chakraborti, and Heni Ben Amor. 2018. Virtual, augmented, and mixed reality for human-robot interaction. In *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. 403–404.

[14] Tom Williams, Daniel Szafir, Tathagata Chakraborti, and Elizabeth Phillips. 2019. Virtual, augmented, and mixed reality for human-robot interaction (vam-hri). In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 671–672.

,