Design of Virtual Reality for Humanoid Robots with Inspiration from Video Games

Jordan Allspaw, Lilia Heinold, and Holly A. Yanco(✉)

Computer Science Department, University of Massachusetts Lowell, Lowell, MA 01854, USA
Jordan_Allspaw@uml.edu, Lilia_Heinold@student.uml.edu, holly@cs.uml.edu

Abstract. Advances in robotics have led to breakthroughs in several areas, including the development of humanoid robots. There are now several different models of humanoid robots available, but operating them remains a difficult challenge. Current operator control interfaces for humanoid robots often require very experienced operators and significant amounts of time for planning. A large amount of the planning and cognitive load is attributable to the operator attempting to gain adequate three-dimensional (3D) situation awareness and task awareness while viewing an interface on a flat, two-dimensional (2D) screen. Virtual reality (VR) has enormous potential to provide benefits to allow the operator to quickly and accurately understand the state of a robot in a scanned 3D environment and to issue accurate commands with less cognitive load. In the gaming sphere, VR headsets remain a new and promising interface for playing video games. In some cases, existing video games are being ported over to VR and, in others, brand new games are being designed with VR in mind. Control schemes and best practices for VR are emerging within the video game industry. This paper aims to leverage their lessons learned and to apply them to the teleoperation of humanoid robots.

Keywords: Virtual reality · Video games · Robotics · Humanoid robotics · Human-robot interaction

1 Introduction

Over the past decade, there has been an increase in the development and use of humanoid robots, just as virtual reality (VR) headsets have become more commercially available with improved capabilities. However, VR has not been used extensively with humanoid robots, although it seems that an immersive view could help with the operation of such systems.

Our team performed an analysis of the human-robot interaction (HRI) techniques used at the DARPA Robotics Challenge (DRC) Trials [19] and the DRC Finals [9], both of which had a large collection of humanoid robots used by the

J. Y. C. Chen and G. Fragomeni (Eds.): HCII 2019, LNCS 11575, pp. 3–18, 2019.
https://doi.org/10.1007/978-3-030-21565-1_1

participating teams. Both of these studies examined approaches taken by the teams for interaction design, such as the number of operators, types of control, how information was conveyed to the operator(s), and varying approaches to semi-autonomous task execution. There were a number of strategies employed by the various teams, but teams predominately relied on a combination of mouse, keyboard, and gamepad interfaces.

The only known uses of VR in the DRC Finals were two teams – of the 23 in the competition – utilizing the Oculus Rift Developer Kit (DK) [10] for viewing point clouds. This technique was used as a situation awareness complement to viewing point clouds through traditional means (i.e., on a computer monitor). No teams used VR as their primary interface for either visualization or control. A discussion with a member of one of the teams revealed that they had investigated using the Oculus Rift DK for control but found it to be of limited use and did not end up using it at the DRC Finals.

However, there is a requirement to visualize a great deal of sensor data when teleoperating, or even supervising, a humanoid robot in complex environments. For the visualization of sensor data in the DRC Finals [9], the teams in the study averaged 5.1 active screens, 6.2 camera views, and 2.5 different point cloud visualizations. There were an average of 1.6 active operators (using an input device for robot control) and 2.8 passive operators (watching over the shoulders of active operators, offering strategic advice), representing a large amount of manpower towards the goal of allowing the operator to gain proper situation and task awareness of the remote environment by interpreting sensor data from a 2D interface (the screen) and building a 3D mental model.

Another interesting and common trait was that teams frequently had different visualization and control paradigms depending on the task they were performing. The analysis concluded that this implied that each team spent a significant amount of effort creating a specialized interface for each task, in order to provide enough situation awareness to allow the operator(s) to complete the task within the allotted time.

Much of the work on both the interface design and the operator task load was centered around allowing the operator to gain a proper level of task and situation awareness by interpreting data from the robot, which allowed the operator to give the robot commands in an effective way. This need is why many interfaces had multiple camera views. However, with modern sensors, there is potential for a VR interface to greatly reduce the difficulty of this problem. For example, depth cameras such as the Kinect or Xtion Pro generate 3D point clouds that can be easily rendered inside a virtual world.

In previous work, we proposed an interface where the operator would use a VR headset to teleoperate a humanoid robot [1]. However, in the time since we initially designed our VR interface, a large number of video games have been released for play with VR headsets. Now that VR games are more commonplace, and there are many popular and well tested games available, we are interested in analyzing various VR control techniques used in video games to discover what lessons can be learned.

In this paper, we examine a sample of popular VR video games, analyzing their approach to controls and data visualization. We then categorize common control techniques of the VR games in our sample and discuss how those techniques could be adapted for controlling a humanoid robot. Some of the techniques used in video games will likely not be applicable to robot teleoperation due to the different assumptions and requirements of the domains. However, there should still be enough overlap for improvements to be made. We propose an altered interface from our original design in [1], as well as a series of studies to compare the two interfaces.

2 Virtual Reality in Video Game Design

We conducted a survey of current VR video games by reviewing online manuals and watching gameplay on YouTube in order to classify the methods for controlling different aspects of the game. If an online manual was not available, the game was omitted from our survey. Our survey originally included twenty-one VR games[1], of which we found game manuals for fourteen[2]. These fourteen games are included in the analysis discussed in this section, with a summary presented in Table 1.

2.1 Use of First vs Third Person

In all of the VR video games that we surveyed, first person was the primary method of interaction between the user and the world. In one game, Job Simulator, the user has the option to move the camera from a first person view to an over the shoulder third person view. However, even when in third person, the user controls the character as if they were still in a first person view, i.e., looking out from the character, not from its side. In all of the multiplayer games that we surveyed, the user could see other players in third person but their character stayed in a first person perspective throughout the game.

When in first person, the user could always see some indication of the location of their hands, usually as gloves on the screen, but sometimes the controllers themselves; however, they could not see the rest of their character.

[1] Job Simulator, Star Trek Bridge Crew, Fantastic Contraption, The Lab, Skyrim VR, Fallout 4 VR, Obduction, Subnautica, Rick and Morty: Virtual Rick-ality, The Talos Principle VR, Anshar Wars 2, Settlers of Catan, L.A. Noire: The VR Case Files, Budget Cuts, Arizona Sunshine, Onward, OrbusVR, Space Pirate Trainer, X-Plane 11, IL2 Battle of Stalingrad, and Eve Valkyrie.

[2] Job Simulator, Star Trek Bridge Crew, Fantastic Contraption, Skyrim VR, Fallout 4 VR, Obduction, Subnautica, Rick and Morty: Virtual Rick-ality, The Talos Principle VR, L.A. Noire: The VR Case Files, Arizona Sunshine, Onward, OrbusVR, and X-Plane 11.

Table 1. Characteristics of the VR games surveyed in this paper.

	1st Person	3rd person	Room-scale	Standing	Seated	Real Walking	Joystick	Teleport	Press to Grab	Hold	Button for Menu	Point At Buttons	Permanent HUD	Popup HUD	Wrist HUD
Job Simulator	•	•	•			•			•	•	•	•			
Star Trek Bridge Crew	•				•				•		•				
Fantastic Contraption	•		•	•	•	•			•	•		•			
Skyrim VR	•		•	•	•	•	•	•	•	•	•			•	
Fallout 4 VR	•		•	•		•	•	•	•		•		•	•	
Obudction	•			•	•		•	•	•		•	•			
Subnautica	•			•	•		•		•		•		•	•	
Rick and Morty: Virtual Rickality	•		•			•			•		•				•
The Talos Principle VR	•		•	•	•		•	•	•			•			
L.A. Noire: The VR Case Files	•		•			•	•	•	•	•	•			•	
Arizona Sunshine	•		•	•		•	•	•	•	•		•			•
Onward	•		•	•		•	•		•	•	•	•			
OrbusVR	•		•	•		•			•	•	•	•	•	•	
X-Plane 11	•		•	•	•			•	•	•	•				

2.2 Movement

The second area we examined to categorize the games is movement: how the game allows the player to navigate their character within the virtual world. This includes movement of the player's camera, as well as of the player's character if they are separate. On the gaming platform Steam, VR games are categorized in three ways: Room-scale, Standing, and Seated [7]. In most of the games that we surveyed (71%), the user could choose which method they used. This choice allows the user base to be as wide as possible, allowing each user to choose the method with which they are most comfortable [11]. The terms are defined as follows:

- **Room-scale:** The user moves within a dedicated area in order to move their character. A chaperone is used to let the user know that they have reached the boundaries of the tracked area in the real world. To leave the dedicated area in the game, the user must use an alternate method for movement, such as joystick or teleportation [3].
- **Standing:** The user is standing but stationary. Standing allows for some lateral movement, but the user must stay within a much smaller area than room-scale [15].
- **Sitting:** As with standing, the user is stationary. Because the user is seated, there is less room to move, but the motion controllers usually still track the user's hands [15].

Of the games we surveyed, 71% had the option of room-scale mode. In three of these (21%), the game was limited to a small area, such as a room, and thus did not have an open world that the user could roam freely. This limitation allowed the user to not worry about moving outside the room-scale area and the need for combining movement methods. The games with an open world (43%)

required that the user use a joystick or teleportation to get around outside of the small square in which they were standing. This method allowed the user to bend to pick up objects and to imitate using a weapon such as a gun or bow and arrow.

Eight (57%) of the games used teleportation as one of their movement methods. The user would point one of their joysticks at a location and press a button, then a line and circle would appear indicating the exact location to which they would teleport. When the user would release the button on their joystick, the character would teleport to the specified location. This teleportation allows the user to move around quickly in the VR world without moving in real life. Two games (Rick and Morty, Star Trek Bridge Crew) that did not technically include point to teleport instead allowed the user to move instantly between designated locations, by either using portals or by pressing a button on a menu.

Nine (64%) of the games used joystick control (also known as smooth locomotion) for movement, where the user is stationary and uses a button on the joystick for walking, similar to how a user would use the arrow keys on a computer. Most VR controllers have a trackpad for this purpose. Six of these nine games also had the option to teleport. These games let the user choose their preferred method for movement, mostly because smooth locomotion causes nausea for many users, while teleportation has been reported to reduce nausea compared to smooth locomotion [3].

One of the games had no movement (Star Trek Bridge Crew). This game also only allowed the user to be seated, and the character would be sitting at a station. By moving their hands, the user could select various buttons for their character and switch stations. Similarly, another game (X-Plane 11) based on fighter planes allowed the user only to teleport outside of the plane, but, inside, the user would use their hands to press buttons to control the plane and would not move in real life.

2.3 Manipulation

All of the games used the joystick buttons to control manipulation. There were two primary methods for allowing a user to pick up an object. The first, similar to many PC video games, would pop up a menu on the screen when the user was in proximity of an object, indicating which button the user should press to use the object. The second was more VR specific, allowing the user to point their joystick at an object, using a laser pointer to show what object the user wanted to select. Upon release of a joystick button, the player would grab the object.

In 50% of the games, the user had to hold down a button to hold an object. When the user released the button, the character released the object as well. The Vive controller has a dedicated button for this, called the Grip button.

2.4 User Interface

Twelve (86%) of the games used a button to bring up a menu in the game. Seven of these allowed the user to point and click to select a menu button, where the

user points the controller at a button and a laser appears. The button would also change appearance (e.g., light up, change size, etc.) similar to how buttons change appearance on a computer when you hover over them with a mouse.

Just one game (Star Trek Bridge Crew) used button selection, both on menus and in the game, as one would in real life. The user moved their hand to hover over the correct button in VR, then used a button to select it. This game had the buttons laid out in front of the user like a console, instead of the more traditional menu hovering in front of the user, in order to simulate the look from Star Trek.

The games that did not include the point and click method had the user click buttons on the controller to select options. This method is more similar to a traditional console game where the interface would indicate which button corresponded to which option, and allowed the user to traverse the list with a joystick or arrow button.

2.5 Heads Up Display (HUD)

A Heads Up Display (HUD) is commonly used in video games to display stats such as health, amount of ammunition, inventory, etc. [18]. This method seems to be less common in VR, with only three games using a permanent HUD. In two, the HUD was simply a method for displaying information about other players and objects (e.g., a health bar above other players or information hovering above objects). The last of these had optional hints that would hover over objects.

A pop up HUD was marginally more common, used by five of the games. Status bars would pop up in certain instances and locations then disappear again. Sometimes the pop up was not in the user's control (e.g., a health bar when attacking an enemy, or a *-X* indicating decreasing health), and sometimes the user could choose to view their status whenever they wished.

A wrist accessory HUD (i.e., the user moves their wrist as if they were looking at a watch, allowing them to view their stats, which is sometimes combined with a menu) was as common as a permanent HUD, used by three games. This form of HUD is entirely up to the user as to when they wish to view it.

3 Potential Application of VR Game Design to Robotics

While VR game design has the advantage of many more hours of testing with many more users than the typical robot interface, the video game world is not always applicable to robotics. In this section, we discuss the VR game control paradigms from popular video games, described above, in the context of robotics, to determine which ones will be most applicable when designing a VR interface for a robot system.

3.1 How Video Games Differ from Robotics

The biggest difference that needs to be considered when comparing VR for robotics and for video games is that in robotics, the world that the user is

interacting with is real and not designed. In video games, the designer can bend the rules of physics and create game specific rules that help themselves or the user when it comes to control. For example, in a video game, sometimes the user is allowed to clip through objects that they accidentally run into, to fall from great heights, and to select objects that are not within grasping distance. Many of the surveyed games allowed the user to "summon" an object when they selected it; that is, when the user pressed the correct button, the object would fly into the player's hand.

Real world consequences also have to be considered when designing a VR control system for a robot. With direct control in video games (i.e., when the user is holding the motion controllers in their hands), it does not matter if the user accidentally moves their arm in the wrong direction or needs to make a strange motion to bring up a menu. While controlling a robot, a designer must account for the fact that if a certain motion will bring up a menu, for example, the robot will also attempt to make that motion. This design can be problematic when the motion is impossible for the robot or the robot must hold a certain position.

It is also worth noting that, in a video game, any object or obstacle the user comes across is there by design. The user will not come across anything unexpected (that is, not expected by the designer), so in video games it is much easier to account for all possibilities that could occur while the user is playing. This is definitely not true in the real world where robots must operate.

3.2 Applying Common Video Game VR Techniques to Robotics

While the VR control used in video games is not directly applicable to robotics, design guidance can still be drawn from it, especially when it comes to what is most comfortable and intuitive for a user. For example, no game surveyed was exclusively in third person and very few games had the option of third person view. This indicates that in VR, the most intuitive method of control is first person – that is, you are the character. It follows that when designing VR control for a humanoid robot, the user should be able to "be" the robot and see through its eyes. We do believe, however, that third person views will also have relevance to human-robot interaction.

Methods for movement in the games were relatively evenly distributed. Most games offered more than one method so that users could choose a method with which they were comfortable. This provision of multiple movement methods also accounts for players with different gaming setups (e.g., some users cannot afford a full room-scale VR setup or do not have the room for it) and expands the possible user base of the game. Games with room-scale also often combine one-to-one motion, where the character moves exactly with the user, along with locomotion with a joystick or teleportation. This design was usually to account for the fact that even in room-scale, the player has a limited space in which to move. A similar problem is encountered in robotics: even if the operator has a room-scale setup, the robot may have more space to move than the operator. One way to tackle this problem is to allow the user to directly control the robot

for smaller motions (e.g., grabbing, reaching, and limited lateral motion) and leave walking to a point and click method. Another way of tackling the limited space in which the operator has to move is to use an omnidirectional treadmill [3], but none of the video games surveyed used this method. This design choice could be attributed to the fact such a treadmill is not accessible to most gamers at home and thus is primarily used in arcade setups.

In the video games that we surveyed, it was extremely common that while the user was holding motion controllers, the game showed a pair of hands instead. When the user pressed a button to grab, the hand would perform a grabbing motion. Since all of the games surveyed used motion controllers to track the users' movements, the users could not physically move, grab, and point with their hands and had to correlate certain buttons to hand movements. A possible application to robotics is that a user could control the robot's hands by using motion controllers, and the buttons could be mapped to preprogrammed hand motions performed by the robot.

In terms of manipulating objects, every video game surveyed which allowed objects to be grabbed used a button on the controller to select the object of interest. It was more evenly distributed which games chose to have the user hold down the "grip button" to keep holding an object or to release the object by tapping a button. Similar to movement, many games also offered settings for object manipulation. The user could choose which method was most intuitive to them, allowing the game to access a larger user base.

Menu access seemed to be tackled in very similar ways across all of the games. The user would press a designated button on the controller to see the menu, then would either use a point and click method of selection or use more buttons on their controller to select. This design can be applied directly to a VR robotics setup, where an operator could use a designated button to access an in-VR menu. This design would account for the fact that current 2D interfaces have many settings and buttons around the view of the robot that would need to be accessed in VR through a menu. A point and click method of selection could also be used, since direct movement could be disabled during menu access.

Most of the games that we surveyed did not include a HUD, but this can be attributed to the fact that many did not need one. Some of the methods of HUD used (e.g., hovering hints and information above other players) would not be applicable to robotics. Since HUDs in video games are usually used to display stats (e.g., health, amount of ammunition, and inventory), it could be used for something similar in robotics. Some applicable statistics to display for robots could include battery power, joint states, and other state of health information.

4 Initial Virtual Reality Interface

Our initial interface [1] was designed before VR video games were the norm and while VR best practices were still being developed. Therefore, when designing our interface, we looked to existing 2D interfaces for inspiration. We particularly looked at interfaces used during the DRC, due to the wealth of information available and the fact that the majority of competing robots were humanoid.

While fully autonomous robots are popular in research and have had success in very specific domains, for many complex tasks having a human operator in the loop is still often preferred. For the DRC specifically, many of the teams utilized very little autonomy. The teams that did have some autonomy typically used it in very specific and limited cases. For example, IHMC started with scripts that could perform tasks on their own and over time broke up those scripts into smaller pieces. The end result was a series of steps where the robot would plan a smaller action, and the operator would either confirm the plan or, more often, make adjustments to the robot's state or plan before approving it [6]. Team IHMC allowed their robot operator to "command the robot via interaction with the three-dimensional (3D) graphic tool rather than specifying robot motion or joint positions directly" [6].

With these factors in mind, we designed an interface where the operator retains control of most of the robot functions. Similar strategies were common among several of the teams, albeit all displayed on 2D screen interfaces, with interaction usually through a mouse and keyboard.

After deciding on the level of autonomy that we wished to used in our interface, we decided that building our interface from the lessons learned during the DRC was the best course of action. While the teams competing in the DRC had a specific goal – namely completing a series of structured tasks in a controlled competition – it still represents one of the most significant events in HRI for humanoid robots. We found that the teams used a combination of 2D and 3D imaging. For instance, teams with greater success at performing tasks relied on a fused display of the 3D robot avatar, point cloud, and camera images, in a common reference frame [9]. However, 2D interaction methods, like a mouse and keyboard, were used to maneuver the displays and issue commands. HRI for humanoid robots consists inherently of 3D data, for which VR offers a unified solution for both 3D display and control. Our goal was to create an interface that would leverage VR in order to increase operator situation and task awareness when visualizing a remote location, while also providing adequate control methods to compete with traditional 2D screen with mouse and keyboard interfaces.

4.1 Robot Platform

While our interface has been designed to be applicable to any humanoid robot with similar features, we developed and tested it using NASA's Valkyrie R5 [13]. Valkyrie is a 6 foot tall, 300 pound humanoid robot with a pair of four fingered hands attached to 7DOF arms. The robot has several sensors built in, including a Carnegie Robotics Multisense [14], located in its head, capable of generating both high and low density point clouds, depending on bandwidth requirements, as well as a stereo camera view. The robot also has an additional pair of stereo cameras in the lower torso. Finally, in addition to accurate joint state and torque tracking on each joint, the robot has embedded tactile sensors in the fingers and palm to detect grasping success.

The types of actions that the robot performs are to move a foot to a location in 3D space and hold it without taking a step; to move a foot to a location in 3D

space and take a step; to command the torso, pelvis, or neck to a commanded position; and to command one of the arms to a location in 3D space.

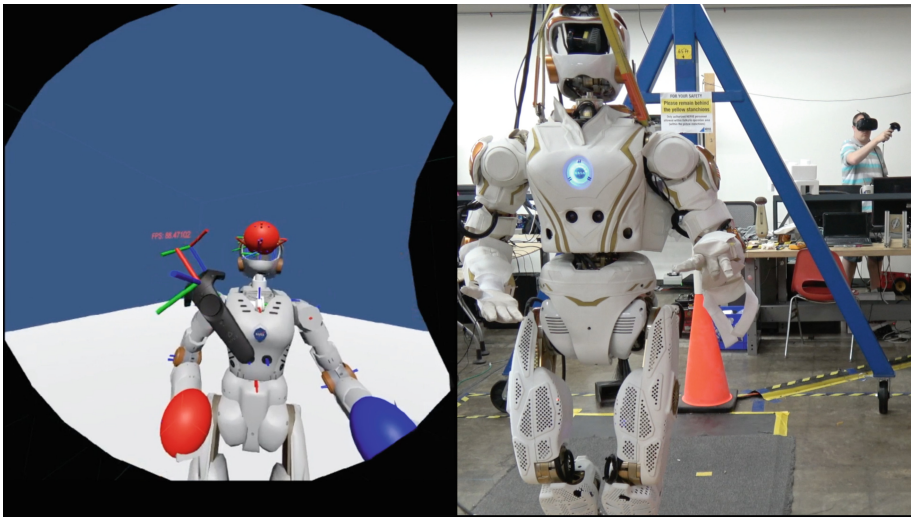

Fig. 1. Screenshot of an operator controlling a robot in VR. On the left is the virtual view that the operator sees; on the right is the real robot with the operator standing behind it.

4.2 Software Platform

Valkyrie's sensors and controller interface communicate using ROS [12], a communication middle-ware commonly used in robotics. When this project was started, the Linux driver for the HTC Vive was lacking many important features so we built our application within the Windows operating system. In order to communicate with the Robot we utilized a Unity [17] plugin of ROS.NET [16] which handled all of the communication and conversion between standard ROS topics and types into things that could be uses within Unity. With the infrastructure settled, we were able to use the most supported and feature complete SDK for the HTC Vive, with the added bonus of being able to leverage several other features within Unity.

4.3 Design of the Initial Interface

The interface utilizes the HTC Vive [5], a commercially available VR headset combined optionally with the Manus VR [8] gloves, a pair of gloves that allows accurate finger tracking, as well as tracking the wrists' location with the HTC Vive Trackers [5]. As an alternative to the glove, we also use the standard controllers that come with the headset. We also utilized a treadmill so that the operator could walk without leaving the tracking area.

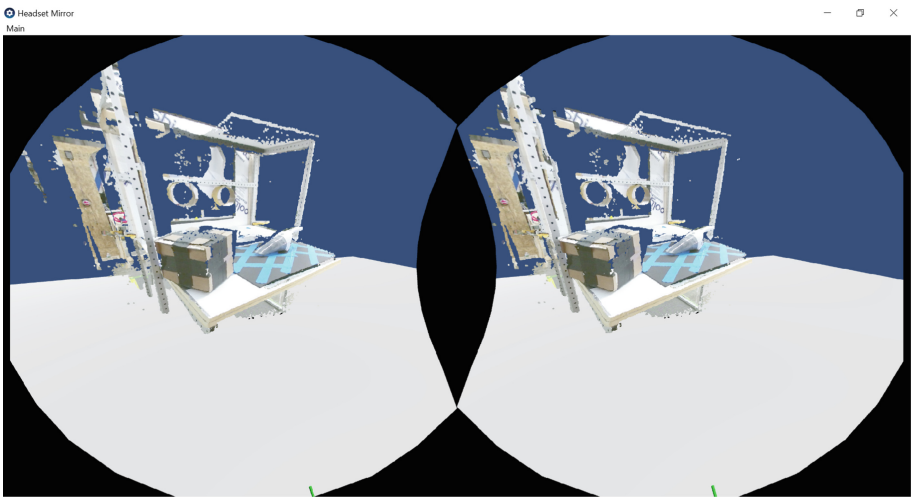

Fig. 2. Visualization of the high density point cloud reconstruction.

The operator sees the robot from a third person perspective, and when walking in real life walks around inside the robot's virtual world, allowing them to see the robot from different angles. An example of an operator viewing the robot in VR can be seen in Fig. 1. The operator can also teleport to different locations in the world if they do not wish to walk.

In terms of movement, the way the operator moves the robot is very similar to a 2D interface. While in VR, the operator reaches out and pulls the robot's limbs in a direction, and then the robot uses inverse kinematics to move that part of the robot in the correct direction. The operator can also plan footstep paths that are then visualized in 3D and can view various point cloud visualizations of the world, as seen in Fig. 2. Finally, they can view what is seen by the robot's front camera by movable interactive windows, as seen in Fig. 3. More information on this interface design can be found in Allspaw et al. [2].

5 Proposed Redesign Based on VR Video Game Analysis

Currently, our VR interface is very different from the interfaces used in video games. As previously discussed, our interface was designed with the operator existing as a virtual avatar alongside the robot. The operator can then see and interact with the virtual robot from a third person view, sending commands only when desired. In the video games surveyed, the player typically exists in the first person, where there movements directly cause the virtual agent to move. In this case, the user exists in the virtual world as the virtual agent, rather than merely alongside it.

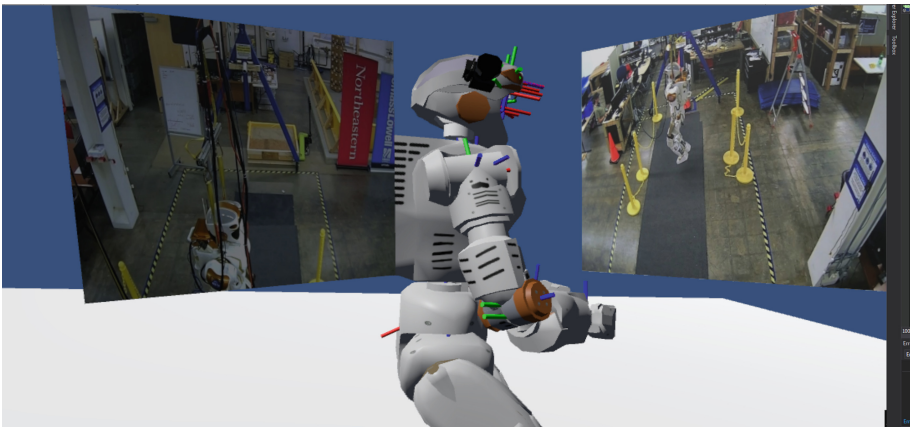

Fig. 3. Virtual world with interactable windows, both displaying different external cameras near the robot.

5.1 Perspective

Our initial interface was designed for a third person perspective, in contrast to the majority of VR video games, which are first person. One possible reason that first person is so ubiquitous is that many first person video games were easily designed for or ported over to VR. It also has the added benefit of possibly increasing the immersion of the player. We are interested in examining in what situations first person versus third person viewpoints during robot teleoperation would be beneficial.

Our initial decision on the perspective was based upon what was being done at the time in robotics, that is, using VR to get a better idea of the robot's position in space rather than control. In a 2D interface, it can be difficult to tell if the robot is close enough to an object to interact with it, or if the robot is oriented correctly to complete a task. An operator can use VR to "look around" a robot to give them a better idea if the robot is in the correct position for a task. In contrast, the main purpose of perspective in VR is to give the user the best immersion possible. However, this does not mean that a VR interface for robotics should not also have immersion as a goal. With this in mind, we propose a possible change to our interface so that the operator can toggle between first and third person based on their current needs.

It should be noted that in our interface, there is nothing preventing the operator from moving their virtual avatar to the exact position and orientation of the robot, allowing them to gain a first person point of view, which was often used in our pilot testing. However, since the controls were designed for a third person viewpoint, trying to maintain a strictly first person viewpoint would be cumbersome. A better solution might be to allow the operator to switch between a proper first and third person viewpoint, possibly adjusting controls and visualizations to match.

5.2 Controls

The common tasks that an operator encounters while controlling our humanoid robot fit into two categories:

1. **Movement:** Tasks that involve moving the robot, whether arms, legs, or the head/sensor systems
   - Instructing the robot to walk forward or backward
   - Positioning arms, hands or fingers in preparation for a task
   - Picking up an object
   - Manipulating an object (such as a handle, a lever or a wheel)
2. **Evaluation:** Tasks that involve ensuring the robot will not be harmed in the next task
   - Evaluating a planned footstep path
   - Evaluating the robot's position before executing a task
   - Evaluating the terrain

Tasks in the evaluation category are not typically relevant in video games. A player usually does not have to worry in a video game if terrain is passable or if their character is too far away to manipulate an object. Typically, the player's virtual character could be considered a near-perfect highly autonomous robot, where the risk of failure is low. If a player commands their character to walk forward, there is a low risk of a catastrophic failure, such as it tripping, except in intentional video game mechanics.

In robotics, evaluation before executing tasks is much more important. If a task is executed improperly, a robot could fall down or break. Evaluation tasks are more easily performed in third person because the user can view the robot and the surrounding from multiple angles and get information that would not otherwise be visible if they could only see in first person.

The tasks in the movement category are much more similar to tasks that also need to be performed in video games. It stands to reason that these tasks could be more easily accomplished from a first person perspective. In our initial interface, the operator would move the robot's limbs in a similar way to a 2D interface: dragging them to an appropriate position in space or using sliders to tune each joint value. However, first person control would allow for greater immersion in VR and potentially more effective control of the robot.

By combining first and third person views into our interface, we hope not only to allow easier movement for the operator (e.g., by allow the robot to move with the operator as in a room-scale video game), but also to allow the operator to evaluate the robot's position and fine tune if needed.

5.3 Movement: Walking

Currently, our interface allows for both room-scale and teleportation. The operator can walk around their space to navigate the virtual world for smaller movements, and, if greater distance is needed, they can use the controller to teleport their play area to a new location. For sending movement commands to the robot,

the operator points to a location in the virtual world, which prompts the robot to plan, and execute if approved, the footsteps required to reach that location. When considering a first person perspective to the interface, it makes sense to use the similar methods for movement. In first person mode, the robot would imitate the operator moving in real life. As the operator moves around their play area, rather than just move a virtual avatar, the robot would attempt to walk to keep up. In this master slave system, the robot obeys any commands sent by the operator. The alternative method, whereby the operator points to a location and the robot plans and walks on their own, would be problematic due to concerns of motion sickness. As the robot walks to the location, the operator would need to be a slave to the robot's perspective as it walks, which would increase the chances of motion sickness. A method for hiding this movement from the user might be necessary to prevent motion sickness.

Given that a humanoid robot could fall if an incorrect step were taken, we will need to conduct a study to determine the best method for walking control, both for the operator and for the robot.

5.4 Movement: Manipulation

Unfortunately, the manipulation techniques used in the surveyed video games are not easily applicable to robotics. Press to grab was overwhelmingly the most popular method of object manipulation, but such a method would require a very high level of consistent autonomy, which is not always available. In video games, objects typically snap into the player's hand when prompted, in a way that is not realistic for the real world. Except in very controlled environments, it is difficult to have a robot capable of grabbing a randomly, and possibly unmodeled, selected object; this is currently an active research problem. However, one important takeaway from the manipulation analysis is that direct manipulation of objects is a large aspect of VR games and thus not something that our interface should ignore.

With this in mind, we propose an alternate design for our manipulation strategy. Instead of using controllers to manipulate the robot's arms and hands, and then watching the robot move to the commanded position, the operator will wear the Manus VR Gloves to provide direct control. The Manus VR Gloves allow for accurate finger tracking for all five fingers, and, when combined with the SteamVR Trackers, we can also track the position of the operator's hands without them holding the controllers. Using this, we can allow control of the robot's hands with inverse kinematics and control the fingers directly. The head can similarly be tied to the operator's headset in a master-slave configuration; as the operator looks in a direction, the robot will do the same. This design goes hand in hand with adding a first person perspective to the interface, as it may be more intuitive to use direct motion to control the robot while in first person.

5.5 Planned Studies and Future Work

Despite the wealth of information that we have gained from our analysis, there are many questions that remain unanswered about the best way to implement a VR interface for a humanoid robot. Our next action is to perform a comparison between the first person and third person interface designs. We plan to have participants use both interfaces to perform a series of tasks using a simulated Valkyrie humanoid robot, using simulated environments from previous challenges such as the DRC and the Space Robotics Challenge [4]. We will then analyze the data to determine in what situations the various controls and visualizations were superior. Metrics to be considered include task success, task time, and operator comfort.

While it is possible that one interface would be unanimously superior to the other in every metric, we suspect that the superior interface will be a hybrid of the two, where the operator is able to seamlessly switch between the two modes. After the study, we will adjust our interface accordingly, and run a new study with a similar design; however, this time we will compare the VR interface against a more traditional 2D interface modeled directly on successful interfaces used in the DRC.

6 Conclusions

With the advances in both humanoid robots and available VR devices, we believe that there is potential for designing a new type of robot interface that will increase operator success when performing tasks that require situation awareness in a remote 3D environment. In this paper, we have discussed our existing work in designing a VR humanoid robot control interface, as well as discussed possible changes to be made. However, more work is needed to determine what types of controls and visualizations will be the most useful for remote teleoperation, as well as determining when a VR interface versus the standard 2D interface may be desired.

Acknowledgements. The work in this paper has been supported in part by NASA (NNX16AC48A) and the Department of Energy's Office of Environmental Management (DE-EM0004482).

References

1. Allspaw, J., Roche, J., Lemiesz, N., Yannuzzi, M., Yanco, H.A.: Remotely teleoperating a humanoid robot to perform fine motor tasks with virtual reality-. In: Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI) (2018)
2. Allspaw, J., Roche, J., Norton, A., Yanco, H.: Teleoperating a humanoid robot with virtual reality. In: VAM-HRI 2018 Proceedings, pp. 7–13 (2018)
3. Langbehn, E., Lubos, P., Steinicke, F.: Evaluation of locomotion techniques for room-scale VR: joystick, teleportation, and redirected walking. In: Proceedings of ACM Virtual Reality International Conference, vol. 4, p. 9. ACM (2018)

4. Hambuchen, K.A., et al.: NASA's space robotics challenge: advancing robotics for future exploration missions. In: AIAA SPACE and Astronautics Forum and Exposition, p. 5120 (2017)
5. HTC: Discover virtual reality beyond imagination. https://www.vive.com/us/
6. Johnson, M., et al.: Team IHMC's lessons learned from the DARPA robotics challenge trials: finding data in the rubble. J. Field Robot. **32**(2), 192–208 (2015)
7. Lang, B.: Steam expands VR game listings with supported headsets, play area, and more. Road to VR, February 2016. https://www.roadtovr.com/steam-expands-vr-game-listings-with-supported-headsets-play-area-and-more/
8. ManusVR: Manus VR: the pinnacle of virtual reality controllers. https://manus-vr.com/
9. Norton, A., et al.: Analysis of human-robot interaction at the DARPA Robotics Challenge Finals. Int. J. Robot. Res. **36**(5–7), 483–513 (2017)
10. Oculus Rift, January 2018. https://en.wikipedia.org/wiki/Oculus_Rift
11. Parrish, K.: Stand up or sit down? Many don't take advantage of VR's room-scale experience. Digital Trends, June 2018. https://www.digitaltrends.com/computing/oculus-rift-owners-want-to-sit-for-vr-experiences/
12. Quigley, M., et al.: ROS: an open-source Robot Operating System. In: ICRA Workshop on Open Source Software, Kobe, Japan, vol. 3, no. 2, p. 5 (2009)
13. Radford, N.A., et al.: Valkyrie: NASA's first bipedal humanoid robot. J. Field Robot. **32**(3), 397–419 (2015)
14. Carnegie Robotics: Multisense SL. https://carnegierobotics.com/multisense-sl/
15. Shanklin, W.: Room-scale or standing VR? Why all that walking around may be overrated. https://newatlas.com/room-scale-vs-standing-vr-oculus-rift-htc-vive/48219/
16. UML-robotics: UML-robotics/ros.netunity. https://github.com/uml-robotics/ROS.NET_Unity
17. Unity: Unity. https://unity3d.com/
18. Wilson, G.: Off with their HUDs! Rethinking the heads-up display in console game design. https://www.gamasutra.com/view/feature/130948/off_with_their_huds_rethinking_.php
19. Yanco, H.A., Norton, A., Ober, W., Shane, D., Skinner, A., Vice, J.: Analysis of human-robot interaction at the DARPA Robotics Challenge Trials. J. Field Robot. **32**(3), 420–444 (2015)