Abstract—Robot teleoperation is a key requirement for enabling users to carry out tasks at a distance. Most existing teleoperation platforms rely on 2D interfaces. Several VR solutions exist, but none render a live 3D environment while operating a mobile manipulation robot. We propose GHOST, an approach to teleoperation that uses consumer VR hardware to enable a person to remotely operate a mobile Boston Dynamics Spot robot. We use Unity to render a 3D point cloud from Spot’s cameras in a virtual environment, enabling the person to control the robot as a “ghost” standing near it, along with a virtual gantry that allows the person to manipulate the robot’s end effector to carry out tasks. Our approach allows an experienced user to teleoperate the robot to perform 8 dexterous tasks such as YCB Cup stacking 42% more quickly than the state-of-the-art Spot tablet baseline. We also present results showing that new users strongly preferred VR to the tablet, and were over twice as successful at a manipulation task with GHOST.

I. INTRODUCTION

Robotic teleoperation is a rapidly emerging technological approach that enables human operators to complete tasks beyond their physical capabilities and away from dangerous environments. For a human operator to accurately control the robot and interact with the environment, physically capable hardware and sufficient situational awareness are necessary [17]. Virtual reality (VR) provides a mechanism to create this situational awareness by enabling a virtual human presence next to the robot that can see the robot’s environment in 3D as though they were physically present [16].

Existing research shows that VR teleoperation can be better than conventional 2D interfaces in success rate and usability ratings [24, 4, 17, 20]. Most of these approaches focus on tabletop robot manipulators, which are not capable of navigating environments. For a human operator to accurately control the robot and interact with the environment, physically capable hardware and sufficient situational awareness are necessary [17].

VR teleoperation for real-world mobile robots with manipulation capability is challenging for several reasons. The environment rendered around the user needs to be immersive, accurate, provide a sense of depth, and update quickly to show the world in real time. The user must be able to control the robot from a remote location to navigate and manipulate objects within its environment. The control scheme for this needs to be as intuitive as possible so that untrained users can learn quickly. Recent advancements in consumer-grade VR hardware and support from software tools like Unity have increased the feasibility of VR teleoperation [3]. The capabilities of mobile robots have been improving concurrently. The state-of-the-art Boston Dynamics Spot robot features advanced mobility, sensing, and manipulation [5].

We introduce GHOST (Ghost Human Operator for Spot Teleoperation)—a VR application in Unity for teleoperation that places the human operator next to Spot as a virtual partner in an immersive, live 3D scene depicting the robot’s surroundings. Users can remotely navigate the robot and manipulate the arm’s end effector by mirroring user hand movements using VR controllers (Fig. 1). The contribution of this paper is a VR system that intuitively allows users to teleoperate a mobile robot. Core criteria to our system are (1) remote control, (2) navigation, (3) manipulation, (4) 3D rendering, (5) live rendering, and (6) the use of VR.

Our evaluation shows that expert users demonstrate a higher success frequency with a decreased completion time on GHOST in comparison to Boston Dynamic’s 2D tablet interface when completing eight tasks including fruit collecting, pouring, and door opening. Novice users who were trained on both systems and given an inclined cup stacking challenge also performed better and conveyed a preference for GHOST over the tablet.

II. RELATED WORK

Teleoperation depends on the robot’s physical capabilities and the quality of visual features that the human operator receives. From these physical capabilities, we create three groups of teleoperation interfaces for robots: those that can only navigate, those that can only manipulate, and those that can do both. A sample of papers on robot teleoperation is included in Table 1. This table was created by identifying which of our core criteria each paper contained.

A. Navigation.

Teleoperation with VR that focuses on robot ground navigation has wide applications from space exploration...
<table>
<thead>
<tr>
<th>Category</th>
<th>Reference</th>
<th>Navigation</th>
<th>Manipulation</th>
<th>3D Rendering</th>
<th>Live Rendering</th>
<th>VR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation</td>
<td>Nguyen et al. [16], 2001</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Becerra et al. [1], 2002</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Nielsen et al. [17], 2007</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Lee et al. [12], 2016</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Elobaid et al. [7], 2019</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Manipulation</td>
<td>Whitney et al. [23], 2018</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Su et al. [20], 2019</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Tsokalo et al. [21], 2019</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Navigation &amp; Manipulation</td>
<td>Wyribek et al. [25], 2008</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Ciocarlie et al. [2], 2012</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>AVATAR [13], 2023</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>ALOHA [8], 2024</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Macias et al. [14], 2020</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Roldán et al. [19], 2019</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>GHOST, 2024</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

VR setups for navigation that display 2D camera information projected on the headset have shown usefulness in providing users a better sense of the environment. However, 2D projections require human operators to adapt to the lack of depth information. For example, Becerra et al. [1] developed a VR teleoperation interface that allows a user to navigate a robot that has a 360 camera attached to a moving base. The operator uses 2D images obtained from the 360 camera to decide where to move the robot. Elobaid et al. [7] use a humanoid robot, an omnidirectional treadmill, and VR to provide a 2D video feed for the user to steer the robot. Lee et al. [12] create a 2D surrounding environment for a human operator to navigate through a pre-made map using a mobile robot equipped with a camera and lidar sensor.

There have also been navigation systems that have incorporated 3D information. Nguyen et al. [16] developed a VR interface for teleoperation using stereo camera information to show a realistic 3D terrain environment around a planetary exploration robot. The limited frame rate created a 5 second delay in robot response time after sending a command. As seen in Table I, most Navigation papers do not use 3D renderings, likely because the value of 3D renderings is highlighted more by manipulation tasks, and 2D rendering methods can be simpler and have lower latency.

B. Manipulation.

Robots designed for manipulation allow them to interact with their environment through grasping, lifting, pushing, etc. VR setups designed for manipulation robots have used both 2D and 3D interfaces. Tsokalo et al. [21] uses a tabletop robot and streams a video from the robots perspective. The user can see the video feed and control the robot using the VR controllers. They use object detection to identify and generate digital models of objects and show that their system fulfills their requirements for remote robot control [21].

The use of 3D information in manipulation robots has been shown to aid users when controlling them remotely [23, 20]. As seen in Table I all of the Manipulation papers used VR, which highlights the advantage of depth information for manipulation tasks. Whitney et al. [23] creates a VR interface to teleoperate robots over the internet. Their interface was successful in controlling a dual-armed tabletop robot while displaying a 3D point cloud visualization. We use their public software to connect ROS to Unity, but address the new challenge of teleoperating a mobile robot. Su et al. [20] use a 3D reconstruction and control the robot through a wireless connection. Incorporating 3D information created more intuitive manipulation [20].

C. Navigation & Manipulation.

Robots with both navigation and manipulation hardware equipment have a larger range of capabilities. 2D interfaces have shown to be useful for these robots [13]. Macias et al. [14] used a stereo 2D video streamed to a headset to control a robot with a moving base and a gripper. They showed users performed better on manipulation tasks in stereo VR than in traditional 2D, but their approach fixes the user to one perspective and does not create a 3D rendering of the environment. Our system is different in that instead of a stereo 2D video, GHOST creates a 3D point cloud which allows the user to change their point of view by moving their head. Roldán et al. [19] also used a mobile manipulation robot and likewise found that VR was superior for manipulation, but not movement. Their approach relies on a pre-modeled environment instead of live rendering, while ours is created live from the robot cameras. Table I shows that most work done on Navigation and Manipulation robots do not have live 3D rendering. This requires users to adapt to the lack of depth information, which creates a cognitive demand that GHOST can decrease.

D. Teleoperation without Virtual Reality.

There has been success in teleoperating a robot without VR. For example, AVATAR [13] connects a tabletop manipulation robot to an omnidirectional mobile base and displays
a live video feed of the robots environment to a human operator. Nielsen et al. [17] developed a system that displays obstacles and free space in 3D with a live 2D video feed at the center of the screen. The use of 3D information allows for better visualization of the robots location and orientation [17]. They do not use VR and the combination of 2D and 3D information is described as mixed-reality. ALOHA [8] built a low-cost mobile manipulation robot capable of a variety of manipulation tasks. They use a bimanual mobile robot to perform these tasks through imitation learning. Teleoperating a robot while in the same room has allowed human operators to know the exact robot response when sending commands. Wyrobek et al. [25] developed a bimanual navigation robot that was teleoperated by a human in the room using a physical gantry system. Ciocarlie et al. [2] developed a system to teleoperate a robot using only a computer screen and mouse. They found that tasks that required depth perception were especially challenging. Remote teleoperation requires a way to mediate human sensory perception like sight and depth, which our system addresses.

III. TECHNICAL APPROACH

The goal of our research is to test the hypothesis that VR can enhance user experience and capability in mobile manipulator robot teleoperation with live rendering. We specifically seek to outperform Boston Dynamics’s own 2D tablet application to control Spot. To accomplish our goal, we created a VR application that allows a user to view a live-updated 3D representation of the environment rendered from data streamed from the Spot robot, and allows the user to teleoperate the robot to move the base and control the arm to manipulate the robot in a variety of dextrous tasks.

A. System Overview

We chose Spot as the robot for mobile teleoperation due to its wide range of capabilities. The cameras that surround Spot’s body are equipped with depth sensors, allowing depth estimation crucial to 3D rendering. Spot can walk around its environment and adjust to tricky terrain, and its arm with attached gripper allows it to grasp and manipulate objects. Boston Dynamics offers an extensive SDK to allow developers to connect to and control Spot in Python [6]. Nothing in our overall approach is specific to Spot; the GHOST interface paradigm can be used with any mobile-manipulation platform.

In our implementation, a computer running ROS connects to Spot using that SDK. The ROS computer sends information from Spot’s sensors to another computer running a Unity VR application [22] connected to a Meta Quest Pro headset [15], and routes commands from that application to the robot. Spot and the computers are connected to a shared network via Ethernet cables.

B. Visual Perception

1) Point Cloud: The primary advantage of virtual reality over 2D interfaces like the tablet is the user’s ability to perceive depth in a VR headset. GHOST renders a live 3D point cloud oriented around the robot, and a 3D model of Spot reflects the real world pose of the robot. A Unity scene demonstrating this environment is shown in Fig. 1. The point cloud is drawn using Unity shaders.

Displaying a point cloud requires color and depth information relative to a location, which can be obtained from Spot’s camera and depth sensors. The point clouds are aligned using each camera’s intrinsics and extrinsics, retrieved from Spot. The Unity scene is robocentric with the robot fixed as the center point of the scene, and as the robot moves, the point cloud moves relative to it.

2) Network Limitations: Relying on three components sending large amounts of information to each other over the network introduced major network challenges. Spot supports on-board compression of the color images to JPEG format, but the corresponding depth is sent as full arrays with two bytes per pixel. The size of those arrays became a major burden on network bandwidth, which decreased the frame rate.

To address the bandwidth issue, we made several design decisions. We wired the whole system with Ethernet to ensure a fast connection, which limits Spot’s mobile abilities, but was necessary in our implementation given the bandwidth and latency limitations of wireless networks available to us. 802.11g wireless protocol limits bandwidth to at most 54 mbps [18], and our system requires approximately 600 mbps to run at the desired frame rate.

Additionally, we chose to only send the depth data and render the point clouds for the two front cameras because they provide the most relevant information for navigation and manipulation in teleoperation. We also used RangeLinear compression [9] on the ROS machine to reduce the size of the depth information from two bytes per pixel to one byte per pixel before the array is sent to the Windows machine through ROS. These changes allowed us to receive depth frames at 20-30 frames per second. The scene itself runs at approximately 40-60 frames per second.

3) Depth Sparsity: Another challenge we faced was the sparse information retrieved by Spot’s depth sensors. Each frame failed to detect a range value for approximately 80% of pixels, and would return a value 0 millimeters away for those pixels. This resulted in an unstable point cloud that had pixels constantly relocating as they came in and out of each frame, creating an unpleasant experience for the user.

To mitigate this sensor issue, we kept a rolling average of each pixel’s values over the last 100 frames while the robot was stationary. This resulted in a more densely populated and denoised point cloud as shown in Fig. 2. We turn this aggregation off while the robot is moving because the information in the previous frames is no longer accurate relative to the robot’s new position—the scene only displays one frame at a time while the robot is moving.

4) Time Synchronization: Finally, because color is processed more quickly than depth due to JPEG compression and the two arrays are received asynchronously, color is always received before depth. Rendering color and depth as they were received led to an unsettling effect where new
C. Controlling the Robot

Our rendering system enhances the user’s view by including 3D perception. However, to outperform the 2D interface for teleoperation, our application must also provide intuitive control over the robot. Our design uses the idea of a “ghost” operator to situate the user in the scene. To accommodate the wide range of inputs necessary for control, we split the interface into two modes — one for driving Spot, and one for manipulating the arm. When the scene starts, the user is placed directly above the rendering of Spot’s body looking forward. Moving the user’s head in the VR headset allows the user to look around the scene. A video of the scene while a user completes tasks is attached.

1) Ghost Partner Interface: The user exists in the VR scene as a “ghost” partner. The person controlling the robot acts as an invisible partner standing near it, helping it carry out tasks. The Unity virtual camera system allows the user to change their perspective through natural head movements.

2) Drive Mode: Navigation controls make up the contents of drive mode. In this mode, moving the Quest’s joysticks sends vector commands to the robot. The left joystick commands the robot to move forward, back, left, and right. The right joystick commands the robot to rotate relative to its current position. These controls are similar to driving the robot on the Spot tablet. The user can also command the robot to stand higher or lower, allowing it to manipulate objects from floor level up to a low table.

In this mode, we set the maximum render distance to 6 meters from the robot. Compression of depth for the point cloud causes a loss of depth precision proportional to the furthest point rendered. To decrease this loss, we limit the distance of the furthest point rendered. The distance of 6 meters was determined by prioritizing vision information necessary for navigation.

3) Arm Mode: Manipulation and viewing controls make up the contents of Arm mode. While in this mode, the robot’s body and arm become translucent to give the user maximum view of the robot’s environment. Moving the joysticks sends vector information that changes the user’s position in the scene. This allows the user to fly around to any desired perspective or location within the point cloud. In the event of an occlusion, the user can freeze and unfreeze the point cloud. This allows the user to grab objects that would otherwise be blocked by the robot’s arm. The maximum render distance in this mode was set to 2 meters to improve the precision of the point cloud for objects within reach.

In this mode, a green copy of the robot’s end effector, which we call the “ghost” hand, appears in the same location as the robots real end effector. To move the robot arm, the user must hold the right trigger and change the position and rotation of the right controller. While the user holds the right trigger, the interface detects the change in the controller’s orientation, which it uses to update the orientation of the ghost hand. The system then sends a command to move Spot’s end effector to the position and rotation of the ghost hand, effectually moving the arm in real time to mimic the motion of the user’s right hand. This feature was made possible by motion tracking on the Quest’s controllers.

In contrast, the tablet sends relative velocity commands in a similar way to our drive mode, so that three dimensional positional and rotational movements are each mapped to different combinations of pressing joysticks and triggers. We argue that sending goal positions based only on the VR operator’s hand movement is a more intuitive way to control the robot arm. Human hands naturally move in the same dimensions as Spot’s gripper. In contrast, individually controlling three positional and three rotational directions requires complex spatial reasoning, and is slower due to the need to individually adjust each axis in turn.

4) Common Controls: Several controls are available in both modes, including the option to switch between 2D and 3D views. Two dimensional depth is a crucial option to supplement the point cloud for two reasons. First, the point cloud can be sparse and limited to at most 6 meters. Second, the side and back cameras are disabled in 3D mode to receive depth data as quickly as possible, but in 2D mode they are enabled to allow the user to look around. The user can also open and close Spot’s gripper, stow the arm, and send an emergency stop command in both modes.

IV. ROBOT TELEOPERATION EXPERIMENTS

The aim of our evaluation is to test our hypothesis that the GHOST interface increases the speed and accuracy of mobile robot teleoperation compared to the tablet baseline. We conducted experiments with expert users as well as
novice users carrying out a variety of manipulation tasks of varying difficulty. Human operators controlled the robot from a different room 20 feet away and performed 8 real-world tasks. We provide videos of the VR interface and robot movement for each task in the attached video.

A. Tasks

We identified 5 categories of tasks that human operators should be able to perform using GHOST: stacking, inserting, opening, grabbing, searching. We derived a set of 8 tasks by choosing common robot tasks that address these categories and highlight the advantages of GHOST.

Detailed descriptions and images of each task can be found in Fig. 7. They are ordered in terms of expert user difficulty from easiest to hardest where 1 is easiest. There is overlap in the categories that each task falls under. Stacking: 4, 7; Inserting: 3, 5, 6; 8; Opening: 2, 6; Grabbing: 1, 2, 3, 4, 5, 6, 7, 8; Searching: 5, 8.

B. Data Collection

Two of the three authors of this paper acted as the expert teleoperators when completing each task. For each task, each expert user was given three consecutive recorded attempts on both the tablet baseline and GHOST. For each recorded attempt, the human operator controlled the robot from a different room without being able to see the robot, and after completion, the elapsed time and success count was documented.

We included five users unfamiliar with either system, but who had previous experience with VR, to collect information on the difficulty difference. Each new user was given 25 minutes of training time in one interface followed by 10 minutes to complete the Inclined YCB Cup Stacking task with the interface they just trained on. This process was then repeated with the opposite interface. Three of the novice users started with the tablet, and the other two started in VR. Training took place in the same room as the robot with the same set of objects to practice grabbing.

C. Quantitative Experiment Results

The timed results for experienced users are shown in Fig. 4. In all 7 tasks after the trivial task 1, the experienced users averaged at least slightly better times on VR tasks than on the tablet. The advantage of VR became more pronounced the longer the tasks ran. Experts rarely failed tasks — there were only two failures across all 32 expert trials, both of which were using the tablet. Across all tasks, experienced users performed 42% faster in VR compared to the tablet. Expert users typically spent at least 60% of the time in 3D mode, and the rest in 2D.

Novice user experiments show even more of an advantage for VR. Fig. 5 shows that all five users new to the system performed better on the cup stacking task, both in speed in and accuracy, regardless of which mode they were trained on first. None of the novice users were able to finish stacking the cups within the ten minute time limit using the tablet, while three of the five stacked all four. The two users who did not stack all four cups in VR spent 6% and 24% of the time in 3D mode, while the three who did spent 45%, 70%, and 100% in 3D mode, suggesting that users who relied too heavily on 2D information performed worse.

D. User Feedback

Figure 6 shows feedback from the novice users taken after task completion, and highlights the all around more positive experience users had in VR when compared to the Tablet interface. The only category the tablet scored better in was in physical demand, since the VR interface has users moving their arm, head, and even bodies to interact with the scene. When asked about their experiences with the two systems, one user said "The tablet felt more familiar, but had a ton of controls to remember..., while with VR you could just move your arm. VR was just significantly easier". Another participant suggested that "With access to VR, there shouldn’t be any need to use the tablet".

E. Discussion

These results support our hypothesis that GHOST is an improvement over the 2D tablet interface in two ways. First, it increased speed and accuracy to accomplish tasks, and
second, it was shown to be a much more pleasant experience for unfamiliar users. All of the tasks required grabbing and manipulating objects (fruit, doors, etc.).

VR users were able to identify the correct location for the end effector based on the point cloud, and using the gesture controls were able to make minute adjustments to the rotation and position of the gripper at the same time. In the cup stacking task, for example, this enabled users to adjust the angle of insertion of the cup while lowering it in, allowing them to quickly make improvements on the fly. The tablet, on the other hand, required users to try to reach the goal location by determining the relative movements and rotations necessary based on the hand’s current location in the 2D image, and then adjust each component of rotation or translation separately. This discrepancy meant that the more grasping and manipulation a task required, the more GHOST’s advantage was highlighted.

While the 3D view is crucial for moving the robot’s hand to the correct location, our results show that 2D information was still leaned upon by nearly all VR users because of the sparsity of the point cloud. These experiments demonstrate that in situations where depth information is sparse, teleoperators still rely heavily on 2D views, making it crucial to provide users the option to switch between 2D and the point cloud. With more dense depth data we predict that the need for a 2D mode might vanish entirely.

Other factors aside from the view type also gave our VR system an advantage. We placed the **YCB Block Stacking** and **Balls in Cup** tasks on the ground instead of on a table because the tablet interface did not allow users to keep Spot standing high enough to see the objects it was trying to manipulate, making the tasks functionally impossible.

Our approach has its limitations. Though the feedback forms showed that GHOST was significantly better than the tablet, it was still not easy to learn or operate. Additionally, we only tested with a relatively small group of users who had some experience with VR, and all were under the age of 35. We expect systems like ours to be less intuitive for people less familiar with VR technology.

Self-occlusion is also a major obstacle in both control methods. Since all sensors to detect the environment are on the robot itself, the user’s view can easily be blocked when the arm moves in front of a camera to manipulate an object. The challenge of self-occlusion can usually be mitigated in VR by raising or lowering the body to the desired point, and positioning the arm to grab an object from the side, so the body camera can still view the object. On the tablet this can be more difficult, because the body automatically adjusts its pose as the hand moves, making it difficult to keep it in the ideal pose to view the object.

### V. CONCLUSIONS

GHOST provides the first VR teleoperation platform for a mobile manipulator robot in a live 3D environment, thanks to Spot’s highly flexible design and accessible API. The experiments outlined in this paper demonstrate that our system is highly capable and has critical advantages over the 2D tablet baseline for teleoperation in speed, accuracy, and user experience.

Our system provides an excellent VR teleoperation interface, but much work remains to improve upon it. Section 5 of Wonsick and Padir [24] list challenges of VR teleoperation applications, most of which still apply to our work. The first is rendering, which could be significantly improved over our relatively sparse point cloud. Our group has explored techniques such as Depth Completion [11] and Efficient Neural Radiance Fields [10].

Network challenges also make rendering difficult, and new solutions need to be developed that allow depth and color streaming with low latency without requiring Ethernet. Wonsick and Padir [24] also note that best practices in VR have yet to be defined, and that user studies need to draw from a larger and more diverse group of people, and need to be more applicable to the real world. Our study is no exception, as we have not conducted enough user trials to make definitive claims about the improvements of VR.

Controls for teleoperation could not only benefit from broadly accepted best practices, but also from the advanced eye and hand tracking offered by newer headsets. Our work makes hand movement more natural by tracking the location of a controller, but hand tracking without controllers, combined with thoughtful user experience design, could lead to a far more natural user experience than anything to date.

Finally, our research could be greatly improved by incorporating multiple robots working in unison. If users could operate two or more Spot robots at the same time, a more cohesive scene could be rendered, significantly mitigating the occlusion problem and increasing immersion. They could also collaborate for multi-handed tasks, compounding their real-world utility.
1. **Grab Soft Object**: Grab and lift a plush toy from the ground. The plush toy is located 4 feet in front of the robot within the robot’s line of sight. This task is meant to be an easy baseline.

2. **Open Door**: Open an unweighted door wide enough for a robot to walk through. This task requires the operator to identify the handle, place the gripper on the handle, close the robot hand on the handle, twist the gripper, and pull it towards the robot. Precise movements are necessary in controlling the robot’s gripper; Otherwise, the door will not open.

3. **Pour Jelly Beans**: Pour jelly beans from a cup into a bowl. The cup and bowl are located next to each other within the robot’s sightline. This task requires users to account for the contents of the cup when moving, if movements are too jerky or imprecise the beans can spill out, necessitating precision when changing the orientation of the robot’s end effector.

4. **Inclined YCB Cup Stacking**: Stack 5 YCB stacking cups. The cups are placed in order from largest to smallest, and every other cup is placed on an inclined plane, which requires the human operator to change the orientation of the robot’s end effector each time they grab a cup.

5. **Balls in Cup**: Place two balls inside of a cup. The two balls are located on opposite ends of a mat and a cup is located between the cups and the robot. The items are placed such that it is not possible for the robot to reach the balls and the cup from the same location. The robot must move to each ball and back to the cup to place it.

6. **Place Lemon in Oven**: Open the oven door of a toy kitchen, place a lemon inside of it, and close the door. The robot cannot reach both the lemon and the oven door from the same location, so the human operator has to move to each location. Placing the lemon inside the oven door requires the operator to have a good sense of depth to know when to let go of the lemon.

7. **YCB Block Stacking**: Stack 6 rectangular YCB blocks in a pyramid formation where there are three blocks on the bottom, two in the middle, and one on the top. The blocks are placed in staggering diagonals, which require rotation of the robot’s end effector to be picked up. Depth information is valuable when balancing the blocks on top of each other since miscalculations result in it toppling over.

8. **Collect Fruit**: Search for, pickup and place 6 plastic fruits into a cooking pan and then pick up the pan, balancing all of the fruit within it, and carry and place the full cooking pan on top of a table. The human operator has to move around the room to search for each of the fruits and balance the pan with the fruit inside it when placing it on top of the table.

---

Fig. 7. The starting state, goal state, and description for each task, ordered by difficulty for expert users.
ACKNOWLEDGMENTS

This work is supported by NSF CNS-2038897, NSF REU Site 2150184, by Amazon Robotics.

We would also like to thank Eric Rosen, Ethan Asis, Autonne Petitjean, Ezra Odole, Daniel Cai, Daniel Hu, Jean Yoo, Jennifer Lu, Michael Fu, Ziyan Liu, Nicholas Vadasz, and Chengfan Li for contributing to this work.

REFERENCES


