2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636005

Telemanipulation via Virtual Reality Interfaces with Enhanced Environment Models

Murphy Wonsick, Tarık Keleştemur, Stephen Alt, and Taşkın Padır[1]

Abstract— **Extreme environments, such as search and rescue missions, defusing bombs, or exploring extraterrestrial planets, are unsafe environments for humans to be in. Robots enable humans to explore and interact in these environments through remote presence and teleoperation and virtual reality provides a medium to create immersive and easy-to-use teleoperation interfaces. However, current virtual reality interfaces are still very limited in their capabilities. In this work, we aim to advance robot teleoperation virtual reality interfaces by developing an environment reconstruction methodology capable of recognizing objects in a robot's environment and rendering high fidelity models inside a virtual reality headset. We compare our proposed environment reconstruction method against traditional point cloud streaming by having operators plan waypoint trajectories to accomplish a pick-and-place task. Overall, our results show that users find our environment reconstruction method more usable and less cognitive work compared to raw point cloud streaming.**

I. Introduction

The capability of remote robot teleoperation is essential for robotic operation in extreme environments, such as disaster response, nuclear decommissioning, bomb disposal, or space exploration. By having the ability to teleoperate a robot from a distance, one can remove the inherent risk these environments pose to humans while keeping their knowledge and expertise in-the-loop. Most current interfaces for robot teleoperation though are 2D interfaces that utilize standard monitor, keyboards, and mice for interaction. These interfaces are often very complex and require operators to handle 3D data using 2D devices. Furthermore, they require comprehensive operator training [1]. Virtual reality (VR), as an emerging interaction technology, offers a potential solution to these problems by allowing for an immersive environment and 3D interaction abilities. Recent advancements in VR devices, making them more affordable and available, have allowed for an increase in VR robot teleoperation interfaces [2]. In general, these VR interfaces have shown to improve performance, lower operator workload, and are found to be considered more usable and likeable among users compared to traditional interfaces. However, there is still room for additional development in VR interfaces before they can be used in real-world applications, particularly in visualizing the robot's environment.

We aim to further the development of VR interfaces for robot teleoperation by designing a system architecture that utilizes object segmentation to reconstruct a modeled version of the robot's environment in VR. Using deep learning, we can detect known objects in a robot's environment and then display the corresponding models inside a VR headset. This allows for an increased fidelity of the real world in VR compared to point cloud streaming, as well as a reduction in the required bandwidth. It is also more flexible than pre-constructed VR environments as objects only need to be modeled once and the VR environment does not need to be altered anytime the real-world changes. The main contributions of this paper are: (1) a deep learning based environment reconstruction framework for VR, (2) development of a robot teleoperation VR interface, and (3) a user study design for the evaluation of the framework and VR interface with trained operators.

The remainder of the paper is organized as follows. We first discuss the prior related work in VR robot teleoperation interfaces in Section II. We present our system architecture and VR interface in Section III. An evaluation of our setup by comparing our proposed environment reconstruction technique to raw point cloud visualization is described in Section IV. The results and a discussion are presented in Sections V and VI, respectively.

II. Related Work

Due to virtual reality devices becoming commercially available, and therefore more accessible and affordable, there has been a rapid increase in VR interface development for robot operation [2]. Several works have demonstrated the utility of VR over traditional interfaces, such as keyboard, mouse & monitor interfaces and direct manipulation interfaces, where the user must physically move the robot. For fixed robotic manipulators, VR has been shown to be significantly better compared to keyboard & monitor interfaces with an improvement in task completion time, lower workload, higher usability, and higher likability score [3]. Similar comparisons have also been conducted using manipulators mounted on wheeled mobile bases with results showing an improvement in performance when using VR interfaces over a traditional monitor & controller interface in manipulation tasks, but not in driving tasks [4], [5]. However, there has been work that shows when utilizing VR to teleoperate a mobile robot to map an environment, users have less collisions and report a greater level of immersion and situational awareness compared to a traditional monitor view interface [6], [7]. Although it does take users longer to complete their exploration tasks when using VR.

*This material is based upon work supported by the National Science Foundation under Award Nos. 1928654, 1935337, 1944453, 1952032.

[1]Murphy Wonsick, Tarık Keleştemur, Stephen Alt, and Taşkın Padır are with the Institute for Experiential Robotics, Northeastern University, Boston, MA, 02115, USA. {wonsick.m, kelestemur.t, alt.s, t.padir}@northeastern.edu

Even with the success of VR interfaces over more traditional ones, there has been limited work in investigating the best interaction techniques or visualization modalities. Hetrick et al. [8] compared two different interaction control techniques for VR interfaces, a trajectory control, where operator movements are translated into robot movements, and positional control, where waypoints are placed for the robot to move through, for a fixed robot manipulator. Their results showed trends suggesting that a positional control paradigm may be more beneficial compared to trajectory control. Van de Merwe et al. [9] examined the effects of different levels of environmental information in VR teleoperation interfaces and found that performance time is decreased with the inclusion of both contextual and task-related information compared to only task-related information. Therefore, they recommend inclusion of contextual information, such as floors or recognizable objects, to improve robot teleoperation VR interfaces. Su et al. [10] ran a similar study comparing a pre-modeled VR environment to one with the addition of a point cloud, concluding that the inclusion of information from the real environment both increased success rates and was found more usable.

However, pre-modeling environments is not always feasible, especially for mobile robots where the environment is not fixed, nor is streaming point cloud data due to potential bandwidth limitations. Kohn et al. [11] aimed to address some of these issues by presenting a method capable of efficient processing and visualization of point cloud data by segmenting known objects from unknown objects to help reduce the bandwidth requirement. This method, however, is designed for a fixed robot manipulator in a semi-known environment.

III. System Overview

Our system can be broken down into three major components: *Robot & Planners*, *Object Detection*, and the *VR Interface*. Each component was developed separately and designed using principles of modularity for ease of future development. Communication between each component is accomplished through the Robot Operating System (ROS).

A. Robot & Planners

In this work, we use the Toyota Human Support Robot (HSR) [12] as our development platform. HSR is a mobile manipulation robot equipped with a 5 degrees-of-freedom (DoF) arm and an omnidirectional base. It has a 2 DoF head that holds the visual sensors including an RGB-D, stereo, and a fisheye camera. To reconstruct the robot's environment, send feedback about the state of the robot, and execute commands from the VR interface, we developed a Planning and Perception Server (PPS). This server has three responsibilities: (I) Plan motions for the desired end-effector poses and return the planned trajectories. (II) Execute robot actions commanded from the VR. (III) Send visual observations of the environment, such as point clouds or the IDs and poses of known objects.

(a) Object Segmentation (b) 6D Object Poses

Fig. 1: Object Detection. Objects are first segmented using a deep-learning model then their poses are calculated using point cloud data.

1) Motion Planning: Our motion planning module is based on the trajectory optimization framework that was developed in our previous work [13]. The framework is built on top of the TrajOpt [14] method where the motion planning is formulated as a non-convex constrained optimization problem and solved using sequential quadratic programming (SQP). The PPS implements a ROS service (*PlannerService*) using this framework in which the user can request multiple end-effector waypoints. Once the *PlannerService* receives desired waypoints, it plans a trajectory and sends it back to the VR interface for visualization.

2) Execution: To execute instructions from the VR interface, we built another service (*ExecutionService*) within the PPS. The *ExecutionService* takes three commands: (1) EXECUTE_TRAJECTORY: to send the latest planned trajectory to HSR's low-level controller, (2) GRASP_OR_RELEASE: to close or open the gripper for grasping or placing objects, and (3) HOME: to return HSR to its original configuration.

3) Perception: The goal of the perception module is to find object poses and return these poses to the VR interface along with the corresponding object IDs. To accomplish this, we developed a service (*PoseEstimationService*) that takes the pixel coordinates of each object in the scene, which is detected by our object segmentation model (explained in Section III-B), and calculates their 6D poses. Once the service receives a set of pixel coordinates, individual localized point clouds for each of the objects are extracted from the depth camera. Object positions are calculated by finding the geometric center and orientation is assigned by assuming the $z-axis$ to be perpendicular to the ground and the $y-axis$ as the longest side of an object.

B. Object Detection

At the core of the environment reconstruction is a deep learning model that was trained using the Detectron2 framework [15]. This model was trained on a custom dataset consisting of 10,000 images of 8 YCB [16] objects: chips can, cracker box, master chef can, mug, potted meat can, pudding box, tomato soup can, and sugar box. Each image contained a minimum of 1 object and a maximum of 8 objects and were annotated using the COCO JSON RLE format. A Mask RCNN instance segmentation model [17] with a ResNet + FPN backbone, which was trained on the COCO dataset [18], was pre-loaded and used as the model's

Category	mAP	Category	mAP
Chip Can	85.88	Potted Meat Can	77.67
Cracker Box	85.34	Pudding Box	71.42
Master Chef Can	86.54	Tomato Soup Can	72.15
Mug	79.77	Sugar Box	81.02

TABLE I: mAP Values

Hyper Parameter	Value
Iterations	25000
Base Learning Rate	0.0025
Learning Rate Decay	3x
Batch Size	512
Images/Batch	2
Training Time	2 hours

TABLE II: Detectron2 Hyper Parameter Values

starting point. We report the mean average precision (mAP) per object category in Table I to show the performance of our object segmentation model. The hyper-parameters for our training can be seen in Table II. The success of the model can be attributed to the variety of the data that it was trained on. The data set used was created by collecting video data of the 8 objects in the environment in which the experiment was conducted. We then apply data augmentation using the SBX Data Multiplier tool [19] to increase the diversity of the dataset by randomizing the objects positions and scales. An example of the object segmentation can be seen in Fig. 1a.

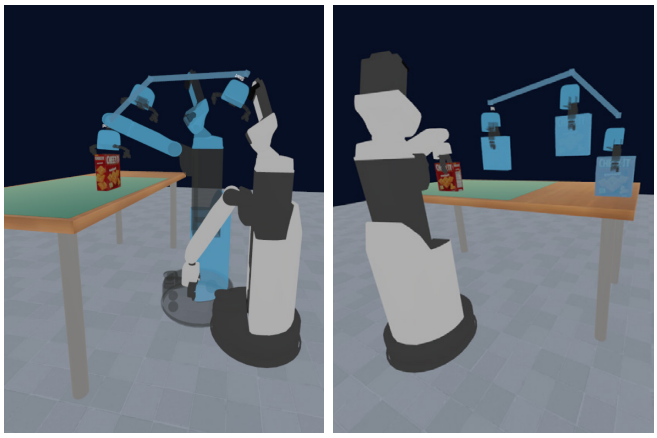
The object detection and image segmentation deep learning model is integrated into the system using a ROS service (*ObjectSegmentationService*). When the service is called, the model grabs the latest image from HSR's RGB-D head camera as input and outputs the bounding box, object class, detection score, and the pixel location data of each object detected in the image. This service is called by the *PoseEstimationService* in order to produce the 6D poses of the detected objects in the scene.

C. VR Interface

The VR interface is modeled after our supervisory-control VR interface for humanoid robots [20]. It is developed using the Unity game engine and uses an HTC Vive with a single Vive controller for the VR equipment. In addition, we utilize an open-source software library called ROS# [21] to assist in all the communication between the Unity project and ROS.

The interface is started with any known fixed objects in the scene, in our case, the floor and a table. Next, the rest of the environment is constructed by calling the *PoseEstimationService* presented earlier, that returns a list of identified objects and their poses in the robot's environment. Transformations are done to convert from the robot's environment to the VR environment and then high fidelity models are displayed in the scene. Fig. 4c shows an example of the reconstructed environment with Fig. 4a as the actual environment.

The VR controller can be in one of two modes, Teleport Mode or Planner Mode, and the controls on the controller change depending on the current mode. Teleport Mode



(a) Displaying Plan (b) Place Planning

Fig. 2: VR Interface. User can place waypoints, request a plan, and have the returned trajectory visualized before execution. Once an object has been grasped, the waypoints are displayed with the currently grasped object.

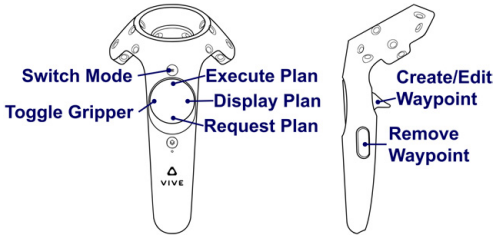
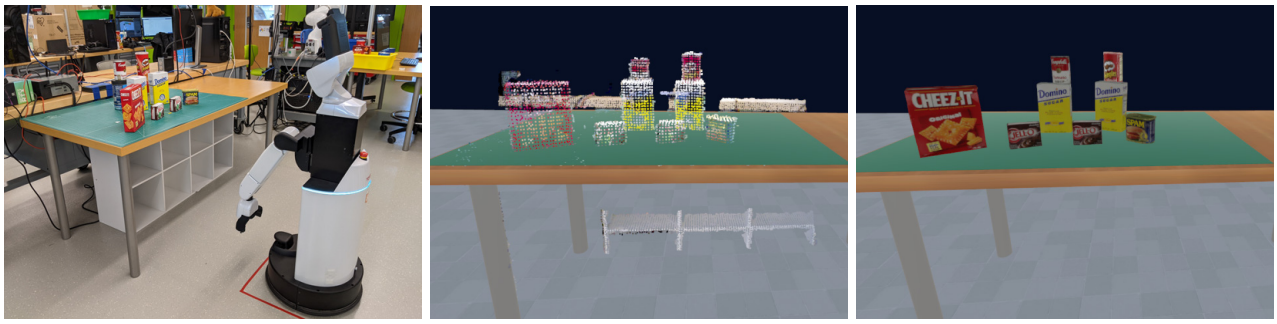


Fig. 3: Planner Mode Controls. Controls change depending on the current mode the user is in.

allows the user to use the controller to navigate around the scene without having to physically move their body around. While Planner Mode is where the user can interact with the robot to create, visualize, and execute trajectory plans, as well as control the robot's gripper. Users are able to switch between modes through a button click on the controller and the active mode appears as text at the top of the user's view.

Planner Mode interfaces with HSR and the planners presented in Section III-A. Fig. 3 shows the controls for this mode. For every button click in this mode, text is displayed at the top of the user's view informing the user that the request was sent and if applicable, text of the result is displayed (i.e., when a plan has been returned or if a grasp was successful or not). Users start by placing ordered waypoints to generate a path for the end-effector to traverse through. Waypoints can be edited at any time by hovering over the waypoint with the controller and using the trigger on the controller to do a click-and-drag operation to move the waypoint around. Once a user is satisfied with the waypoint placement, they can request a plan from our motion planning module. When a plan has been returned, it can be displayed to ensure it is both collision free and matches the desired intent. Plans can be displayed an unlimited number of times and can be discarded at any time by simply editing, adding, or removing a waypoint. Fig. 2a provides an example of placed waypoints



(a) Real World Environment (b) Point Cloud Visualization (c) Model Visualization

Fig. 4: Experiment Setup.

and the final configuration of the returned plan after the trajectory was displayed. After ensuring the returned plan is as intended, users can request the robot execute the plan. Following execution, users can then toggle the gripper to open/close to grasp/drop an object.

One additional feature of our interface using this system, is the ability to display a visualization of the grasped object in the waypoint visual. Fig. 2b shows an example of waypoints visualized with the grasped cracker box. The goal of this feature is to assist users in placing objects once grasped.

IV. Evaluation

To evaluate the effectiveness of our proposed system, we conducted a user study comparing our VR environment reconstruction method, hereby known as the *Model Visualization*, to the *Point Cloud Visualization* that streams and displays a raw point cloud inside the VR headset. We used a pick-and-place task to help evaluate each visualization. Fig. 4 shows our experiment setup.

A. User Study Design

We designed the pick-and-place task with the following considerations:

- Objects were chosen with a variety of different shapes and all objects had the ability to be grasped in more than one way to allow for multiple solutions.
- Objects were placed in close proximity to require a higher level of precision in waypoint placement. This also increased the difficulty in placing waypoints in a manner that resulted in a collision-free trajectory from other objects on the table.
- Similar shaped objects were used to increase the difficulty in object recognition. We selected two pairs of similar shaped objects: (1) a chips can and a tomato soup can and (2) a potted meat can and a pudding box.
- Some objects were partially occluded to increase the difficulty of waypoint precision, ensuring a collision free trajectory, and object recognition.

The task involved picking three specific objects, a cracker box, a potted meat can, and a chips can, from the left side of a table and placing them on the right side of a table. The primary goal of the task was to place waypoints in a way that accomplished the pick-and-place task while avoiding collisions with the other objects on the table. Both the task and the underlining interface remained the same between the two visualizations, with the only difference being that there was no object visualization in the waypoint visual, as seen in Fig. 2b, for the *Point Cloud Visualization* since objects were considered "unknown" in this visualization. All the objects' poses were kept the same between participants, but the two pairs of similar shaped objects were randomized each trial.

B. Experimental Procedure

Participants were first brought in and given a high-level overview of the study and demographics were collected. Next, the general VR interface was demonstrated and afterwards, participants were placed in VR to grow accustomed to both the interface itself and VR in general. To help offset any potential learning biases, the first visualization was randomized for each participant. Participants were then provided a chance to practice with the visualization with a single cracker box placed in the scene for interaction. After participants were comfortable with the interface and visualization, we moved on to the pick-and-place task, where they were informed that their primary goal was accuracy, specifically to avoid collisions, and secondary goal was speed. Once complete, participants were given the questionnaires and then the process was repeated with the next visualization, starting with a chance to practice again. Finally, after completing the task in both visualizations, participants were asked to provide verbal feedback of their experience.

C. Measurements

There are four, two subjective and two objective, measurements we used to evaluate our *Model Visualization* to the more traditional *Point Cloud Visualization*.

1) Subjective: For the subjective measurements, we measured both the usability of the system and the perceived workload. For the usability measurement, we utilized the System Usability Scale (SUS) [22]. The SUS is a commonly used tool that measures the usability of a system by asking participants to rate 10 statements related to their perception of the system on a scale from 1-"Strongly Disagree" to 5-"Strongly Agree". SUS scores can range from 0-100 with any score above 68 is considered as "above average". For the

workload measurement, we used the NASA Task Load Index (NASA-TLX) [23]. The NASA-TLX measures workload by having participants rate on a scale from 0-Low to 100-High their overall perceived, mental demand, physical demand, temporal demand, performance, effort, and frustration.

2) Objective: For the objective measurements, we measured the time to complete each object pick-and-place and the total time to complete the task, as well as the overall accuracy. Completion times were collected through a logging system that recorded every time the gripper changed state. This allowed us to record each objects total pick and place time, as well as the total completion time. If a participant failed to pick-and-place an object, the time for that object was removed from the calculation and their total time became the average time of the other successful pick-and-places multiplied by 3. Accuracy was measured by counting the number of failures, with a failure being defined as any unsuccessful attempt to grasp an object, pick up the correct object, or major collision. For this given task, a perfect score is defined as 0 and the worst possible score is 3, where there was no successful pick or place operation.

D. Participants

Because of limitations caused by the COVID-19 pandemic, participants were recruited from Northeastern's robotics labs. We had a total of 14 participants (14 males, 0 females) complete our study with ages ranging from 19 to 33 ($M = 25.36, SD = 3.99$). All participants had prior experience with robotics. 11 of our participants had little to no previous experience with VR and the remaining 3 had only moderate VR gaming experience.

V. Results

Wilcoxon signed-rank tests were conducted to test for statistical significance for all the measured data.

A. Subjective

Table III presents the results from the SUS and NASA-TLX questionnaires for both visualizations. The usability for the *Model Visualization* scored much higher than the *Point Visualization* and the results were found to be statistically significant with $p = 0.000122$. Results for the workload were also found to be statistically significant with $p = 0.0336$.

Measure	Visualization	Mean	SD
SUS	*Point Cloud*	69.11	17.37
	Model	90.89	8.75
NASA-TLX	*Point Cloud*	31.31	13.44
	Model	26.25	13.08

TABLE III: SUS and NASA-TLX Results

B. Objective

Table IV displays the average time for each object and the total time for each visualization and Fig. 5 displays the box plot of the total time. The results show that participants were

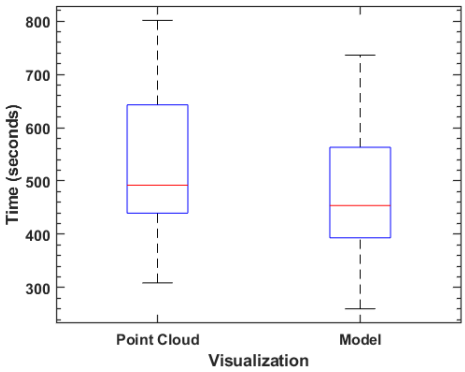


Fig. 5: Total Task Completion Times for Each Visualization.

	Point Cloud		*Model*	
Item	**Mean**	**SD**	**Mean**	**SD**
Cracker Box	3:08	1:02	3:01	1:17
Potted Meat Can	2:24	0:42	2:08	0:26
Chips Can	3:17	1:37	2:54	1:28
Total	9:01	2:33	7:59	2:22

TABLE IV: Completion Time (mm:ss) Results

faster on average for each object individually and overall while using the *Model Visualization* compared to the *Point Cloud Visualization*. However, the overall completion time results were not found to be statistically significant with $p = 0.135$. Furthermore, completion time results showed that on average participants were 2 minutes and 59 seconds faster when going from the *Point Cloud Visualization* to the *Model Visualization*, but only 49 seconds faster on average when going from the *Model Visualization* to the *Point Cloud Visualization*. Additionally, participants were approximately equally as accurate for each visualization with a slight advantage to the *Model Visualization*. There were a total of 4 failures for all the *Point Cloud Visualization* trials, with all four failures being object picking failures, and only 2 failures for all of the *Model Visualization* trials, with only one failure being an object picking failure and the other a major collision with the table.

VI. Discussion

Overall, our subjective results show, with statistical significance, that participants found our developed environment reconstruction, *Model Visualization*, more usable and less taxing than the more traditional visualization, *Point Cloud Visualization*. We also observed this with participant feedback, which was overwhelmingly positive. Almost every single participant used the word "fun" to describe the interface and several remarked how intuitive the interface was after a small learning curve that was generally caused by the unfamiliarity of VR from many of our participants. Additionally, many participants noted that they appreciated the object visualization inside a waypoint when an object was currently in the robot's gripper and stated that it helped in placing the object on the table.

Unfortunately, the objective measures were not found to be significant. However, there are trends that suggest the *Model Visualization* reduces overall completion time. It was observed that most participants appeared to feel more comfortable and confident with the interface during their second visualization and a few verbally shared this observation as well. This generally resulted in lower, or at least similar times, for the second visualization compared to the first one, regardless of which visualization was first or second. This speed up was much larger on average though for participants that started with the *Point Cloud Visualization* and then moved to the *Model Visualization*, compared to the reverse, suggesting that the *Model Visualization* decreases completion times. A larger sample set could help solidify these trends. In addition, an alternative task that is more dependent on accurate visualization, such as a stacking task, could potentially show greater differences in time completion between the two visualizations, as participants appreciated having the grasped object displayed in the waypoint claiming it made object placement easier. An alternative task could also potentially alter the accuracy results as well. During the study, several participants asked for help identifying the objects in the scene when using the *Point Cloud Visualization*. Participants were informed to do their best and ultimately no mistakes were made in the object recognition, but several participants admitted to taking a "best guess" when identifying objects and took more time identifying them. Less familiar or recognizable objects would likely decrease the accuracy for the *Point Cloud Visualization*.

VII. Conclusion

This paper presented our environment reconstruction method that creates high fidelity VR environments to represent the robot's environment. Our work revealed that by using our proposed system, the usability of our VR interface is increased and the overall workload is reduced. However, there is still additional work for further improvement, such as evaluating the proposed system with other user groups and investigating a combination visualization that can utilize the environment reconstruction for known objects, but standard visualization techniques for unknown ones. Overall, this work helps demonstrate the feasibility of using VR to create more natural and easy-to-use robot teleoperation interfaces.

References

[1] M. DeDonato, F. Polido, K. Knoedler, B. P. Babu, N. Banerjee, C. P. Bove, X. Cui, R. Du, P. Franklin, J. P. Graff *et al.*, "Team wpi-cmu: Achieving reliable humanoid behavior in the darpa robotics challenge," *Journal of Field Robotics*, vol. 34, no. 2, pp. 381–399, 2017.

[2] M. Wonsick and T. Padır, "A systematic review of virtual reality interfaces for controlling and interacting with robots," *Applied Sciences*, vol. 10, no. 24, p. 9051, 2020.

[3] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex, "Comparing Robot Grasping Teleoperation Across Desktop and Virtual Reality with ROS Reality," in *Robotics Research*, N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, Eds. Cham: Springer International Publishing, 2020, pp. 335–350.

[4] J. J. Roldán, E. Peña-Tapia, P. Garcia-Aunon, J. Del Cerro, and A. Barrientos, "Bringing Adaptive and Immersive Interfaces to Real-World Multi-Robot Scenarios: Application to Surveillance and Intervention in Infrastructures," *IEEE Access*, vol. 7, pp. 86 319–86 335, 2019.

[5] M. Maciaś, A. Dąbrowski, J. Fraś, S. Karczewski Michałand Puchalski, S. Tabaka, and P. Jaroszek, "Measuring Performance in Robotic Teleoperation Tasks with Virtual Reality Headgear," in *Automation 2019*, R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds. Cham: Springer International Publishing, 2020, pp. 408–417.

[6] P. Stotko, S. Krumpen, M. Schwarz, C. Lenz, S. Behnke, R. Klein, and M. Weinmann, "A VR System for Immersive Teleoperation and Live Exploration with a Mobile Robot," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, nov 2019, pp. 3630–3637.

[7] J. Du, W. Sheng, and M. Liu, "Human-guided robot 3D mapping using Virtual Reality Technology," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2016, pp. 4624–4629.

[8] R. Hetrick, N. Amerson, B. Kim, E. Rosen, E. J. de Visser, and E. Phillips, "Comparing Virtual Reality Interfaces for the Teleoperation of Robots," in *2020 Systems and Information Engineering Design Symposium (SIEDS)*. IEEE, apr 2020, pp. 1–7.

[9] D. B. Van de Merwe, L. Van Maanen, F. B. Ter Haar, R. J. E. Van Dijk, N. Hoeba, and N. der Stap, "Human-Robot Interaction During Virtual Reality Mediated Teleoperation: How Environment Information Affects Spatial Task Performance and Operator Situation Awareness," in *Virtual, Augmented and Mixed Reality. Applications and Case Studies*, J. Y. C. Chen and G. Fragomeni, Eds. Cham: Springer International Publishing, 2019, pp. 163–177.

[10] Y. H. Su, Y. Q. Xu, S. L. Cheng, C. H. Ko, and K. Y. Young, "Development of an Effective 3D VR-Based Manipulation System for Industrial Robot Manipulators," in *2019 12th Asian Control Conference (ASCC)*, 2019, pp. 1–6.

[11] S. Kohn, A. Blank, D. Puljiz, L. Zenkel, O. Bieber, B. Hein, and J. Franke, "Towards a Real-Time Environment Reconstruction for VR-Based Teleoperation Through Model Segmentation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2018, pp. 1–9.

[12] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, "Development of human support robot as the research platform of a domestic mobile manipulator," *ROBOMECH Journal*, vol. 6, no. 1, pp. 1–15, 2019.

[13] T. Keleştemur, N. Yokoyama, J. Truong, A. A. Allaban, and T. Padir, "System architecture for autonomous mobile manipulation of everyday objects in domestic environments," in *Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, 2019, pp. 264–269.

[14] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[15] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," 2019. [Online]. Available: https://github.com/facebookresearch/detectron2

[16] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[19] I. Dewancker, J. Kuntz, and A. Avdacev, "SBX Robotics Data multipler," https://sbxrobotics.com, 2021. [Online]. Available: https://sbxrobotics.com

[20] M. Wonsick and T. Padır, "Human-humanoid robot interaction through virtual reality interfaces," in *2021 IEEE Aerospace Conference*. IEEE, 2021.

[21] M. Bischoff, "ROS#," Jan. 2021. [Online]. Available: https://github.com/siemens/ros-sharp/releases/tag/v1.7

[22] J. Brooke, "Sus: a "quick and dirty'usability," *Usability evaluation in industry*, vol. 189, 1996.

[23] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Advances in psychology*. Elsevier, 1988, vol. 52, pp. 139–183.