Virtual Reality based Telerobotics Framework with Depth Cameras

Bukeikhan Omarali[1,2], Brice Denoun[1], Kaspar Althoefer[1], Lorenzo Jamone[1], Maurizio Valle[2], Ildar Farkhatdinov[1]

***Abstract*— This work describes a virtual reality (VR) based robot teleoperation framework which relies on scene visualization from depth cameras and implements human-robot and human-scene interaction gestures. We suggest that mounting a camera on a slave robot's end-effector (an in-hand camera) allows the operator to achieve better visualization of the remote scene and improve task performance. We compared experimentally the operator's ability to understand the remote environment in different visualization modes: single external static camera, in-hand camera, in-hand and external static camera, in-hand camera with OctoMap occupancy mapping. The latter option provided the operator with a better understanding of the remote environment whilst requiring relatively small communication bandwidth. Consequently, we propose suitable grasping methods compatible with the VR based teleoperation with the in-hand camera. Video demonstration: https://youtu.be/3vZaEykMS_E.**

I. Introduction

In comparison to conventional 2D display, keyboard and mouse teleoperation interface, Virtual Reality (VR) headset and handheld wireless controllers provide the operator with improved depth perception [1], more intuitive control and remote environment exploration [2], [3]. enabling challenging applications as disaster relief [4], surgery [5], exploration [6], and automation [7]. VR based teleoperation interfaces work efficiently when used with RGB-D cameras which provide a 3D point cloud visualization of the remote environment [8]. The majority of VR teleoperation systems use a single external static RGB-D camera directed at the robot's workspace [3], [7], [9]. In such systems, the task performance suffers from incomplete and imperfect visual reconstruction of the remote environment. Depending on the remote environment's reflectivity, geometry, and overall lighting conditions some objects may appear distorted in the point cloud [10]. Occlusion is another issue limiting visual feedback in VR when a single camera is used.

A possible approach to improve visual feedback in VR is to use an additional in-hand RGB camera [3] (a camera attached to the robot's end-effector (EE)) and render the video stream on a surface in VR. However, the grasp would have to be performed relying on the video stream rather than the point cloud which reduces the benefits of the VR. Alternatively, the remote environment can be represented with the help of multiple external static RGB-D cameras [11]. Although it greatly reduces potential occlusions it does not eliminate them, and deploying multiple cameras in a remote site is not always practically feasible.

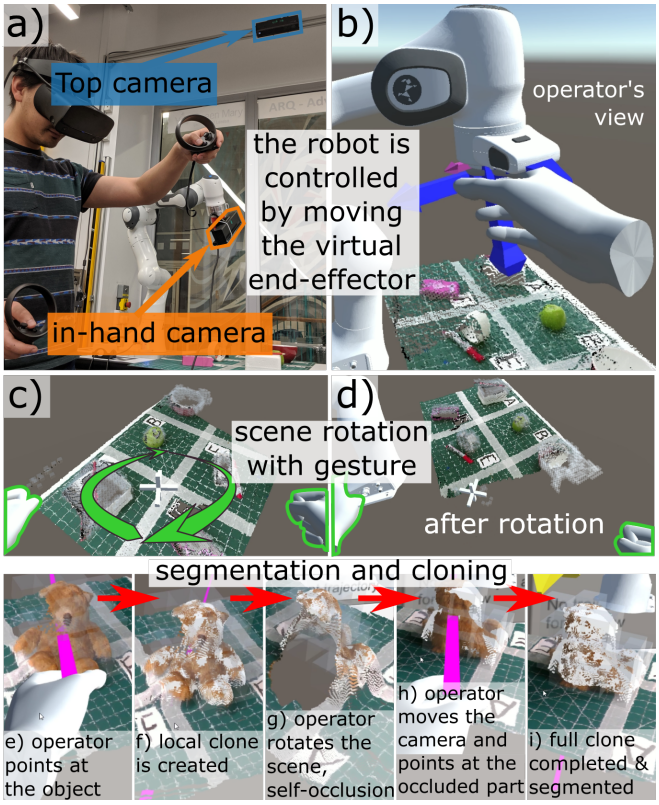

Fig. 1. Overview of the proposed teleoperation system: a) the experimental setup for the visualization study; b) the operator's view; c,d) the operator manipulates the virtual scene with gestures; e-i) steps for object segmentation in the proposed framework

We propose a VR teleoperation framework with dynamic field-of-view control, gestures-based manipulation of the virtual scene, and multiple grasping methods with in-hand RGB-D camera. We experimentally demonstrate that an in-hand RGB-D camera combined with OctoMap [12] occupancy mapping provides the operator with a superior overview of the remote environment compared to the conventional single static camera [3] and multiple camera setups [11]. Furthermore unlike [11] our framework is suitable for unknown and unstructured environments. The gestures-based manipulation of the virtual scene allows the operator to manipulate the 3D visualization of the remote environment in VR. Although the in-hand RGB-D camera improves the

*This work is supported by the EPSRC UK (NCNR EP/R02572X/1, and MAN[3], EP/S00453X/1) and by the QMUL-Genoa Univ. PhD Program.

[1]School of Electronic Engineering and Computer Science (Centre for Advanced Robotics @ Queen Mary, Centre for Intelligent Sensing), Queen Mary University of London, UK; [2]Cosmic Lab, The Electrical, Electronics and Telecommunication Engineering and Naval Architecture, University of Genoa, Italy; {b.omarali,i.farkhatdinov}@qmul.ac.uk

visualization of the remote site, the grasping and manipulation become more problematic as the point cloud of the grasp might not be registered if the grasp occurs very close to the camera. We propose multiple grasping methods that allow the operator to consistently perform grasps in our framework.

II. Proposed framework

A. Dynamic field-of-view control

We provide the operator with the ability to control the field-of-view of the remote camera to reduce visual occlusions and distortions and improve the operator's situational awareness. Although in-hand RGB visual feedback was used in various robotics applications [3], [13], [14], to our knowledge, there has been no research that use in-hand RGB-D cameras as a part of the VR teleoperation interface.

The major differences between the in-hand camera and external camera are the level of detail and the size of the captured area. External cameras are usually placed to maximize the overview of the remote site, resulting in a large but low detailed 3D reconstruction. By comparison the in-hand camera provides a more detailed view of a smaller area. We suggest that a mapping techniques such as OctoMap [12] can be used to continuously generate a lightweight occupancy map of the robot's workspace to compensate for the lack of overview. In VR occupied OctoMap nodes are displayed as transparent boxes, layered over the point cloud, see Fig. 1.c,d. The map is generated continuously as the operator explores the remote environment.

B. Grasping methods with in-hand RGB-D camera

The grasping with an in-hand RGB-D camera is more complicated compared to grasping with an external camera, since RGB-D cameras require a minimum distance to an object to register it as a point cloud. If the in-hand camera is too close to the gripper the operator will have to grasp nearly blind. We propose three solutions that vary in the operator's involvement in the process, illustrated in Table I and demonstrated in the video demo: https://youtu.be/3vZaEykMS_E.

Method 1: Grasp by direct control is designed for direct teleoperation in which the operator drags the virtual EE to control the robot in real-time. The core idea for this solution is to perform the grasp on a persistent segmented clone of the object of interest rather than on the "live" point cloud. We perform a naive segmentation by separate OctoMap chunks - all occupied contiguous OctoMap nodes - to isolate the object, illustrated in Fig. 1.e-i. The operator indicates the object of interest by pointing at it with a ray that extends from the operator's hand. All points contained in the isolated OctoMap chunk are then segmented and cloned to persistent local memory. If the partial clone of the object is insufficient the operator can move the camera and add more points to the existing clone. As the operator approaches for the grasp the "live" point cloud of the object disappears, since it is too close to the camera, but the clone persists and can be grasped, see Fig. 2.A.

TABLE I
Task delegation across grasping methods

	Grasp object	Grasp pose	Grasp trajectory
Direct grasp	Operator	Operator	Operator
Grasp on pose	Operator	Operator	Robot
Point and click	Operator	Robot	Robot

Method 2: Grasp on pose is designed for supervised teleoperation where the operator plans the grasp on the "live" point cloud. It is arguably less physically demanding as the operator does not have to manually drag the robot to the grasp pose. In Fig. 2.B the operator specifies the position of the grasp on the "live" point cloud. The desired grasp pose is then sent to the motion planner which proposes the grasp trajectory that can be previewed by the operator. Finally, the operator decides to reject or accept the trajectory.

Method 3: Point and click takes advantage of automated grasp generation to further reduce the teleoperation task load. The idea is only to point at an object of interest and let the robot propose the grasp pose and the trajectory. We use the same pointing and segmentation method as in direct grasping, shown in Fig. 2.C except that the segmented point cloud is then fed to the grasp pose generator. The resulting grasp pose is then passed to the motion planner similar to the *Grasp on pose* method. The operator previews the proposed grasp pose and trajectory and rejects or accepts them.

C. Gestures-based manipulation of the virtual scene

We propose an intuitive set of gestures that allows the operator to manipulate the visualization of the remote environment. This helps the operator to navigate and inspect the remote environment, if the operator is uncomfortable with physically moving in VR (walking or crouching) or prefers to operate sitting (to reduce physical exertion).

Gestures are inspired by the 3D drawing tools like Tilt Brush [15]. To rotate the scene the operator should press both buttons of the left and right handheld wireless controllers and perform rotation as if rotating a physical steering wheel. In this case, the center of rotation is fixed to the middle point between the operators' left and right hands. Translating and scaling is implemented similarly. Pulling and pushing the center of rotation will pull and push all objects in the scene. Bringing arms closer/further apart will scale the virtual environment up/down. Translation, rotation, and scaling gestures can operate simultaneously. The gesture-based manipulation of the virtual scene is shown in Fig. 1.c,d.

III. Experimental setup

The experimental setup consisted of: the Franka Emika's Panda robot, two Microsoft Kinect2 RGB-D cameras (for the visualization study), Intel d415i RGB-D camera (for the grasping demonstration), Oculus Rift VR headset with Oculus Touch controllers, master PC, slave PC and a local Ethernet network. The setup for the visualization study is shown in Fig. 1.b and its schematic diagram is shown in Fig. 3. The first Kinect2 camera was placed two meters above the robot. The second Kinect2 or Intel d415i were attached

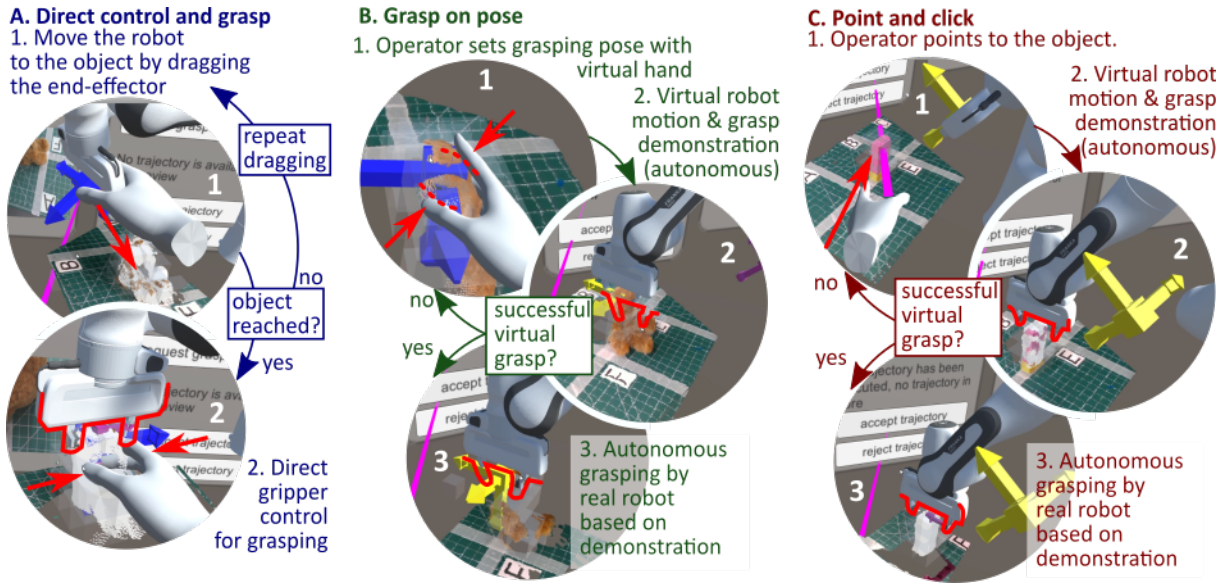

Fig. 2. Grasping methods with in-hand RGB-D camera in the proposed framework.

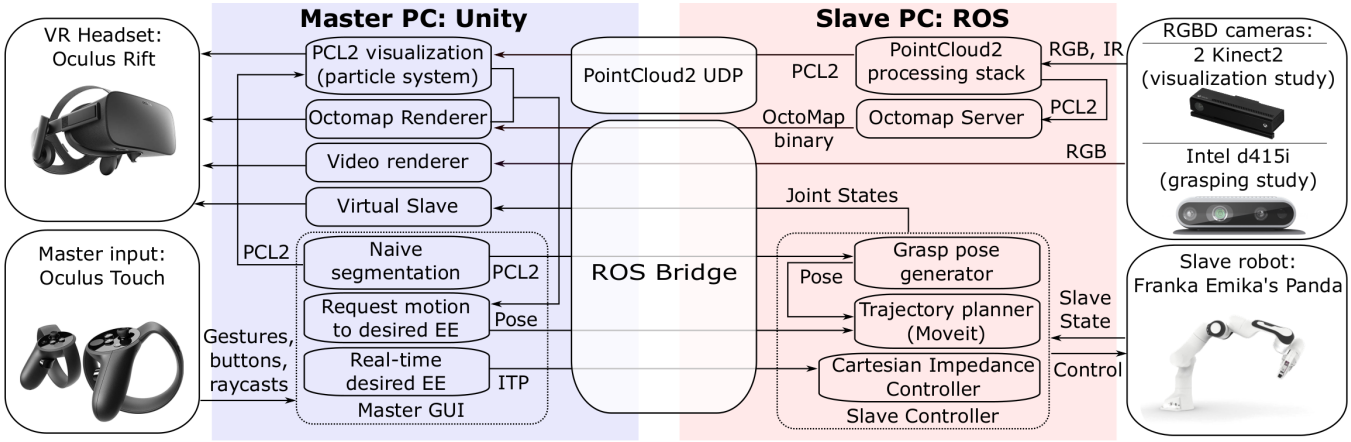

Fig. 3. The schematic diagram of the experimental setup.

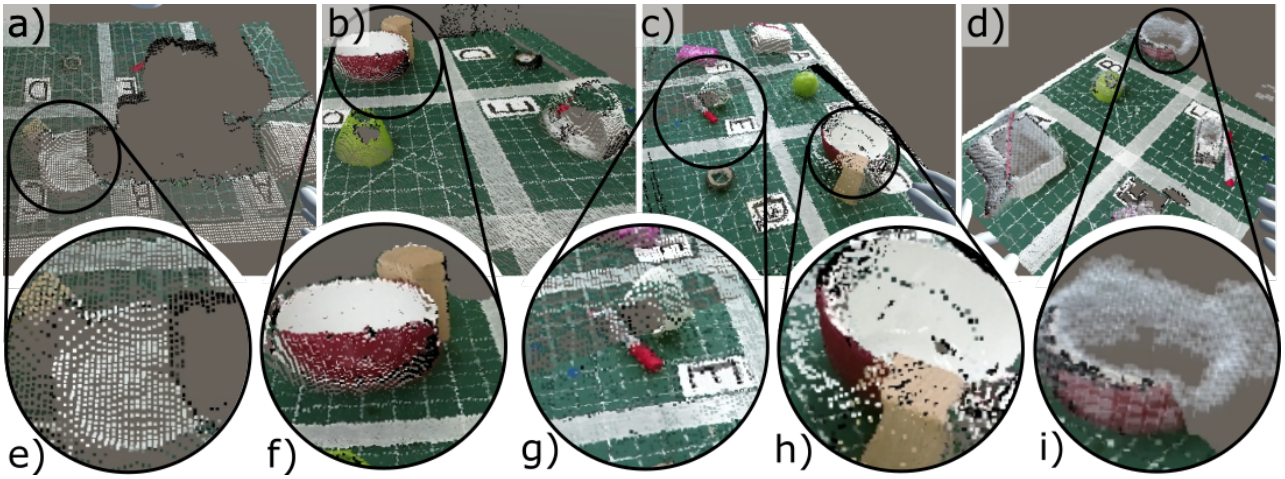

Fig. 4. Operator's view in different modes: a,e) top camera: the robot occludes a part of the view and the bowl's shape is deformed; b,f) in-hand: the bowl is more recognizable; c,h) double camera: note the difference in bowl registration g) the edge between in-hand and top cameras - notice the difference in resolution; d,i) in-hand camera with OctoMap: the bowl is not in the camera's view but the OctoMap maintains an accurate geometrical representation.

to the robot's EE and pointed along the EE. We ran the ROS-bridge on the slave PC to publish ROS messages to a WebSocket and ROS-sharp on the master to read them. We separated the point cloud (only the XYZRGB portion) into a dedicated UDP channel. In our tests the dedicated UDP channel proved to be faster than the ROS-bridge.

Kinect2 point clouds were generated using the standard definition - 512X424, the Intel d415i - 640X480. We ran the XYZRGB registration on the slave PC. This simplified the point cloud and OctoMap generation compared to registering on master [3], although it uses more bandwidth. To reduce the bandwidth consumption we cropped the point cloud to the area of interest - 0.9m X 0.6m X 0.3m and removed all points on which registration failed. We repacked original point clouds to 15 bytes per point (standard Kinect2 registration uses 32 bytes, Intel d415i - 24 bytes). We visualized the point cloud and the segmented clone using the Unity's particle system. We kept the segmented clone white to make it visible when layered over the "live" point cloud.

The widely used OctoMap representation of the environment was produced based on the in-hand camera's point cloud. We only detected objects that were located above the desk. For the visualization study the OctoMap resolution was set to 2 cm. We designed a custom parser and renderer for the binary OctoMap. The renderer visualized only the occupied OctoMap nodes with transparent boxes.

The RGB video was disabled in the visualization study as we were interested in the operator's ability to understand the remote scene from the point cloud only. The robot was controlled in direct teleoperation mode, similar to [2], [3], [16]. The operator moved the robot by dragging and rotating the virtual desired pose axes mesh, see Fig. 1b. Then, the desired change of pose was published in the Interoperable Teleoperation Protocol (ITP) format [17], [18]. The slave controller ran the Panda robot in the Cartesian impedance mode. This was also the control mode for the direct grasping.

We also added a supervised control mode (for "grasp on pose" and "point and click" grasping) that controlled the robot using the *MoveIt* with the RRTConnect planner. We defined three types of motion requests in the corresponding master GUI: "move to pose", "move to grasp pose", "propose grasp on point cloud". The "move to pose" generated a trajectory to a desired EE pose. The "move to grasp pose" command was similar to "move to pose" except it queued an additional pre-grasp pose that ensured a collision-free movement. The "propose grasp on point cloud" command used the segmented clone of the object of interest to generate a grasp pose using the principal component analysis [19]. The generated grasp pose was then processed as in "move to grasp pose" command.

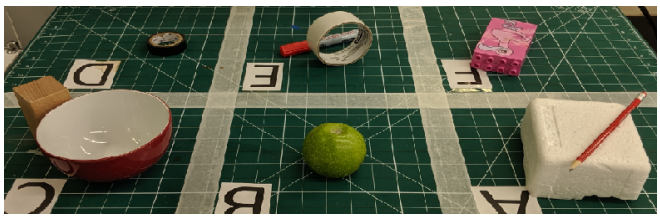

Fig. 5. An example of the area of interest used in the experimental study.

The robot mesh was animated using either the actual robot's joint angle values (default setting) or trajectory proposed by *MoveIt* command. The Master GUI was set such that whenever a trajectory message is received the virtual robot visualization is toggled into preview mode. The virtual robot animation was switched back to the real robot when the operator rejected or accepted/executed the trajectory.

IV. The visualization study

The goal of the study was to investigate the effect of the RGB-D camera placement on the operator's ability to understand the remote environment. For this study we implemented four visualization modes used as experimental conditions (see table II): **M1)** *top camera*; **M2)** *in-hand camera*; **M3)** *double camera* (top and in-hand); **M4)** *in-hand camera with OctoMap*. The experimental task was to explore the remote environment in order to visually recognize and locate objects placed randomly next to the robot. The performance of each task was characterised by completion time, number of correctly recognised objects and NASA-Task Load Index (TLX) [20].

A. Experimental Protocol

The robot's workspace was divided into a six-segments grid as shown in Fig. 5. Thirty six different objects (varied in size, shape, color, and reflectivity) were used for the visualisation study. Nine different objects were selected and placed in each of the grids for each trial (their locations and combination were randomized in for each trial). Each object appeared for each participant once. In certain trials some segments of the grid were left empty and some objects could overlap to create partial occlusions. The robot was placed into a random pose before each trial. Ten participants were recruited from for the experimental study (all healthy adults; one female; age 25-30). All participants had some experience with VR but did not have prior experience with robot teleoperation. Each participant was given a 10 minute training time during which participants got accustomed to the VR teleoperation interface and the testing procedure, using a separate training set of objects.

In each trial a participant was asked to identify an object type, its location (all grids were labeled), and communicate verbally the object name or color and shape as well as its location. Participants had no prior knowledge of what object might appear in the grid. The number of incorrect object recognitions was counted. A full point was given for each

TABLE II

Visualization modes used in the study

mode	top Kinect2	in-hand Kinect2	OctoMap
M1	yes	no	no
M2	no	yes	no
M3	yes	yes	no
M4	no	yes	yes

incorrect answer. A half-point was given when the participant was able to locate the object but failed to recognize the object or the shape of the object; for example ”green cone” instead of ”green sphere” or ”apple”, see Fig. 4.b (an apple is located in the scene but the point cloud appears to look like a cone). We did not specify the total amount of objects in the scene to the participants, nor did we inform them if any objects were missed. After each trial the participant filled in the NASA-TLX questionnaire. We also benchmarked the average and peak communications bandwidth usage by recording the number of points in the point cloud and the size of the OctoMap binary message.

B. Results and discussion

Visualisation and object recognition. The results of the study are presented in Fig. 6. The top plot of Fig. 6 demonstrates that the visualization based on the top camera was characterized by a high number of incorrect recognitions. Participants often failed to recognize the shape of an object if objects were non-convex or when looking at an object along its axis of symmetry, i.e. looking straight down at the bowl (see Fig. 4.e). Other objects were missed due to partial occlusions. In all other modes participants were able to correctly recognize the same objects thanks to the ability to direct the camera sideways as shown in Fig. 4{f,h}. In double camera mode (M3), the mean number of misses was slightly higher compared to the in-hand camera only (M2). This is surprising given that more information can be accessed with the latter. It is likely caused by the imperfect overlap between the point clouds, see Fig. 4h. If one of the cameras gives a false detection on the object shape - the operator would not know which camera to trust. According to ANOVA test visualization modes affect the number of missed objects ($F(3,37) = 15.83$; $p < 0.01$). The in-hand camera with the OctoMap (M4) mode had the lowest number of incorrect object recognitions. That means that a combination of dynamic field-of-view adjustment and OctoMap representation of the scene is an efficient human-robot interface for VR based remote inspection tasks.

Completion time and workload. Second and third plots of Fig. 6 shows the results for task completion time and NASA-TLX index across all participants per visualisation condition, respectively. The differences in completion times and NASA-TLX are insignificant: $F(3,37) = 0.32$; $p = 0.81$ and $F(3,37) = 0.13$; $p = 0.93$ (ANOVA test). However, we can outline some interesting behaviours observed during the experiment. Although completion times are comparable across modes, the distribution of the time per task is different. In the in-hand camera mode, the participants spent more time re-positioning the camera; on the other hand, in the top camera mode the operators spent more time manipulating the view using gesture-based control. This is reflected by the shift from the mental demand to physical demand on the TLX breakdown, see bottom Fig. 6 which presents the average NASA-TLX breakdown across all participants.

The double camera mode has the lowest effort value, which is to be expected given that it provided the most

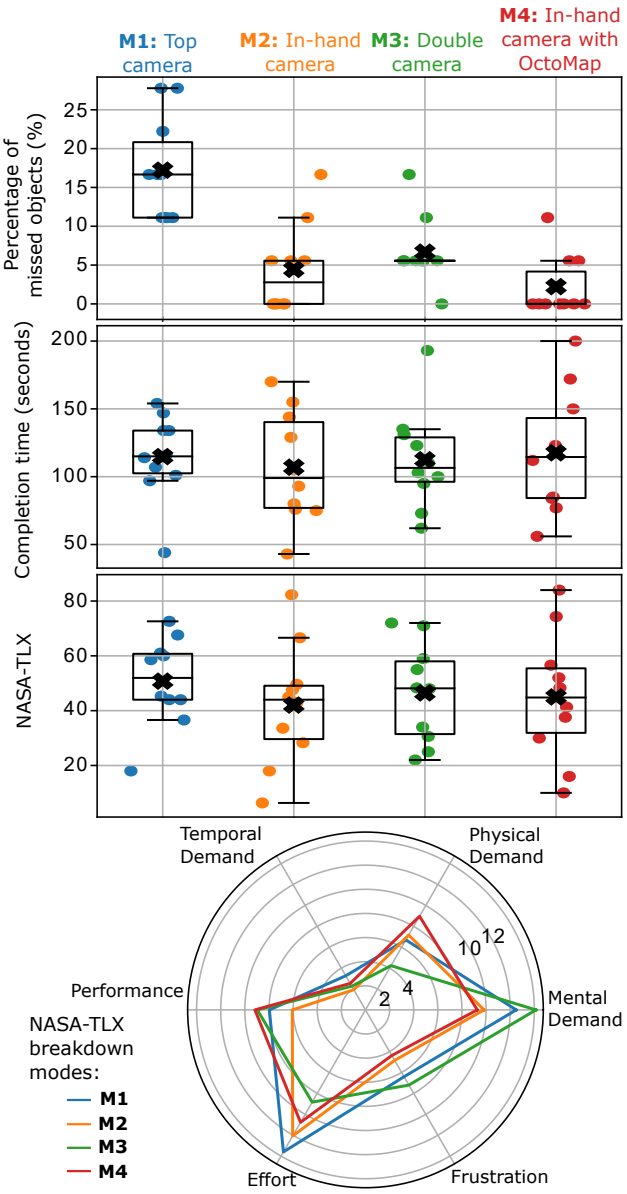

Fig. 6. Experimental results averaged across all participants. From top to bottom: percentage missed objects; completion time; NASA-TLX index; averaged NASA-TLX breakdown. Boxes span test sets from the upper to the lower quartile with a line at the median, whiskers extend from boxes until the last datum within 1.5 interquartile range. Crosses mark mean values. Scatter points represent individual participants.

full representation of the robot's environment. The relative increase in the frustration and mental load was likely caused by the aforementioned imperfect overlapping between the cameras' views and reduced point cloud refresh rate, caused by asynchronous merging of top and in-hand point clouds.

Another interesting behavior observed during the study is the fact that operators preferred to change their view in VR through gestures rather than by displacing their whole body. Although it did not have a significant impact on our study, the gesture-based re-positioning in VR teleoperation can be useful for telerobotics systems that require the operator to remain physically stationary, for example seated.

TABLE III

COMMUNICATION BANDWIDTH COMPARISON

Mode	Mean (MB/s)	Std.dev. (MB/s)	Max (MB/s)
Full PointCloud2	97.69	0	97.69
Top camera	7.78	0.78	8.82
In-hand camera	41.48	10.51	63.73
OctoMap	0.75	0.15	1.05

Data transmission rate. The proposed VR teleoperation interface and visualization modes rely on sending a significant amount of data from the slave site to the master site. Therefore, we have also investigated how much communication network bandwidth was required in each of the visualization modes. The results are summarised in Table III. Since the top camera is placed much farther from the area of the interest the portion of the point cloud that represents the area of interest is considerably smaller than the in-hand camera. As a result the cropped point cloud of the area of interest is much lighter for the top camera compared to the in-hand camera. Naturally bringing the top camera closer to the area of interest would increase the point cloud resolution but it is arguable if it would affect the underlying problems of point cloud self-occlusion and distortion. The in-hand camera with OctoMap provides a comparable overview of the area of interest to double camera mode, but the OctoMap requires much less communication bandwidth than the top camera.

Grasping and camera location. The grasping with an in-hand RGB-D camera has proven to be more difficult compared to grasping with an external camera, since RGB-D cameras a limited to have a minimal distance to an object for it to register a point cloud. In situations when the in-hand camera is too close to the gripper and the object the operator has to grasp the object without or with limited visual feedback. We proposed three solutions for grasping: grasp by direct control, grasp on pose, point and click. We plan to perform a comparative study of the operator's task load across these grasping methods in future work.

V. Conclusion

We proposed a VR teleoperation framework with a dynamic field of view control, gesture-based navigation and control, and multiple grasping methods suitable for robotic manipulation systems with the in-hand RGB-D camera. The key results of our study are: 1) the in-hand camera combined with OctoMap visualization improves scene understanding; 2) users prefer to navigate the scene through proposed gesture rather than moving physically; 3) OctoMap provides an overview of the remote scene comparable to an extra camera but at much lower communication bandwidth cost. In future work we plan to replace the current naive segmentation with a semantic segmentation pipeline to allow the operator to request grasps on objects in a clutter.

References

[1] W-K Fung, W-T Lo, Y-H Liu, and N Xi, "A case study of 3D stereoscopic vs. 2D monoscopic tele-reality in real-time dexterous teleoperation," in *2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005, pp. 181–186.

[2] D. Rakita, "Methods for Effective Mimicry-based Teleoperation of Robot Arms," *Proc. of the Companion of the 2017 ACM/IEEE Int. Conf. on Human-Robot Interaction*, pp. 371–372, 2017.

[3] D. Whitney, E. Rosen, D. Ullman, E. Phillips, and S. Tellex, "ROS Reality: A Virtual Reality Framework Using Consumer-Grade Hardware for ROS-Enabled Robots," in *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, no. July, oct 2018, pp. 1–9.

[4] T. Rodehutskors, M. Schwarz, and S. Behnke, "Intuitive bimanual telemanipulation under communication restrictions by immersive 3D visualization and motion tracking," *IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 276–283, 2015.

[5] M. Draelos, B. Keller, C. Toth, A. Kuo, K. Hauser, and J. Izatt, "Teleoperating robots from arbitrary viewpoints in surgical contexts," *IEEE Int. Conf. on Intel. Robots and Systems*, pp. 2549–2555, 2017.

[6] J. Thomason, P. Ratsamee, J. Orlosky, K. Kiyokawa, T. Mashita, Y. Uranishi, and H. Takemura, "A Comparison of Adaptive View Techniques for Exploratory 3D Drone Teleoperation," *ACM Trans. on Interactive Intelligent Systems*, vol. 9, no. 2-3, pp. 1–19, 2019.

[7] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation," in *2018 IEEE Int Conf on Robotics and Automation*, 2018, pp. 1–8.

[8] L. Pérez, E. Diez, R. Usamentiaga, and D. F. García, "Industrial robot control and operator training using virtual reality interfaces," *Computers in Industry*, vol. 109, pp. 114–120, 2019.

[9] D. Lee and Y. S. Park, "Implementation of Augmented Teleoperation System Based on Robot Operating System (ROS)," in *2018 IEEE/RSJ Int. Conf. on Intel. Robots and Systems*, 2018, pp. 5497–5502.

[10] Y. He and S. Chen, "Recent Advances in 3D Data Acquisition and Processing by Time-of-Flight Camera," *IEEE Access*, vol. 7, pp. 12 495–12 510, 2019.

[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 Int. Conf. on Intel. Robots and Systems*, oct 2018, pp. 1–9.

[12] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, vol. 34(3), pp. 189–206, 2013.

[13] J. I. Lipton, A. J. Fay, and D. Rus, "Baxter's Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 179–186, jan 2018.

[14] D. Rakita, B. Mutlu, and M. Gleicher, "An Autonomous Dynamic Camera Method for Effective Remote Teleoperation," in *Proceedings of the 2018 ACM/IEEE Int. Conf. on Human-Robot Interaction*, USA, 2018, pp. 325–333.

[15] Google, "Tilt Brush." [Online]. Available: https://www.tiltbrush.com/

[16] I. Farkhatdinov, V. Balashov, J.-H. Ryu, and J. Poduraev, "A comparative study of indirect and direct workspace representation in human-robot interaction," *IFAC Proc. Volumes*, vol. 42(4), p. 1730, 2009.

[17] H. H. King, B. Hannaford, Ka-Wai Kwok, Guang-Zhong Yang, P. Griffiths, A. Okamura, I. Farkhatdinov, Jee-Hwan Ryu, G. Sankaranarayanan, V. Arikatla, K. Tadano, K. Kawashima, A. Peer, T. Schauss, M. Buss, L. Miller, D. Glozman, J. Rosen, and T. Low, "Plugfest 2009: Global interoperability in Telerobotics and telemedicine," in *IEEE Int Conf on Robotics and Automation*, 2010, pp. 1733–1738.

[18] B. Omarali, F. Palermo, M. Valle, S. Poslad, K. Althoefer, and I. Farkhatdinov, "Position and velocity control for telemanipulation with interoperability protocol," in *Annual Conference Towards Autonomous Robotic Systems*. Springer, 2019, pp. 316–324.

[19] T. Suzuki and T. Oka, "Grasping of unknown objects on a planar surface using a single depth image," *IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, pp. 572–577, 2016.

[20] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Human Mental Workload*, ser. Advances in Psychology, P. A. Hancock and N. Meshkati, Eds. North-Holland, 1988, vol. 52, pp. 139 – 183.