A Mixed Reality Interface for Robot Remote Control via Magic Leap Devices

Cecily Merkle, Eric Tüschenbönner, Robert Jomar Malate, Thorbjörn Höllwarth

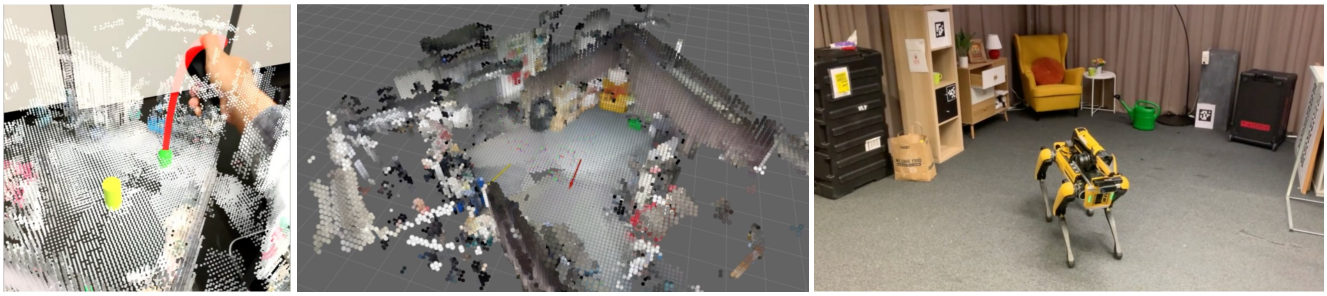

Figure 1. **Drag-and-drop demonstration**: Magic Leap interface (left), ROS visualization (middle), and robot in real life (right).

Abstract

This paper presents a system for remote control of robotic systems via mixed reality devices based on a similar work using the Microsoft HoloLens 2. The primary objective is to investigate the potential synergies between augmented reality and robotics, particularly focusing on intuitive control mechanisms. To visualize the environment of the robot, point cloud data is streamed and rendered on the Magic Leap 2 (ML2) AR glasses. In this map, the position of the robot is represented by a yellow game object. To change the position of the robot, a green game object can be dragged by using the Magic Leap controller and placed on the map. Additionally, a user study was conducted to test user preferences on the two different kinds of control types: "Drag-and-Drop" and "Velocity control". Finally, results from a real-world test show how the robot can be successfully controlled using the Magic Leap device.

1. Introduction

Augmented reality (AR) and robotics converge to redefine human-robot interactions, promising immersive and intuitive experiences. Imagine wearing AR glasses that not only visualize point clouds generated by a robot on a construction site but also enable you to guide its movements seamlessly within the AR environment to execute tasks, that are either difficult for humans to perform or tend to be routine tasks easily done by non-human devices and tools.

Technological improvements like these will also show off economically, as Shraft et al. [24] outline with three primary goals to harness robotics technology in small-scale production: minimizing training duration to a single day, streamlining modifications and reprogramming processes, and significantly decreasing overall programming time. The proposed system aims to make robot control more intuitive, accessible, and faster to learn, even for individuals with limited coding and robotics expertise.

In recent years, the integration of AR technology has revolutionized how we perceive and interact with our surroundings. Simultaneously, robotics, exemplified by the Boston Dynamics Spot robot, has advanced human-robot collaboration across diverse domains. This project explores the synergy of these technologies to enhance user experiences in controlling and understanding robotic systems.

However, challenges persist in seamlessly integrating point cloud visualizations from a robot into AR environments while enabling real-time control. Existing solutions are mostly based on 2D and often lack reliable 3D implementations when it comes to maneuvering robots in the real world. This is because the most used human-robot interfaces are 2D screens, where each adaptation to control the robot in 3D space comes with certain costs [3, 5, 18, 22, 23, 28]. Addressing these challenges is pivotal to unlocking the full potential of robotics applications.

The primary goal of this project is to develop a ROS plug-in for the Magic Leap 2 platform [19]. This plug-in not only facilitates the visualization of the environment by the

Boston Dynamics Spot robot (using its ROS-based interface) but also enables users to interact with the robot, intuitively guiding its movements through an immersive AR interface. Ultimately, the implementation within the scope of this project involves a user, wearing AR glasses, seamlessly setting desired goal poses for the robot through a straightforward drag-and-drop action. Once a goal pose is transmitted, the robot autonomously plans its trajectory and navigates to the specified position while avoiding obstacles along its path.

In the subsequent sections, we delve into the Magic Leap 2 platform, the architecture of the developed plug-in, and the functionalities it introduces. By combining cutting-edge AR technology with the dynamic capabilities of a robot, this project strives to contribute to the development of more interactable robotics experiences.

2. Related Works

This paper presents a re-imagined mixed-reality human-robot teaming system, inspired by the work of Chen et al. [8] about integrating robot control and visualization. While our work doesn't significantly extend the system, we undertake a redesign specifically tailored for the Magic Leap 2 [19], deviating from the original implementation with HoloLens 2 [1]. Our adaptation ensures compatibility with the new platform, demonstrating the system's flexibility and applicability in diverse mixed-reality environments.

The system designed by Chen et al. contains two major components. The first component is a real-time, multi-agent visual SLAM pipeline that includes gravity alignment and reference frame averaging procedures. The second component visualizes the cumulative map in the mixed reality device, correctly aligned to the real world. The system allows operators to visualize spaces beyond obstacles and control multiple agents in 6 DoF within the HoloLens.

The mapping and co-localization system constructs a map based on camera images, IMU, and self-odometry signals for each agent. Co-localization aligns and merges local maps into a unified global map using landmarks to maximize geometric consistency and a centralized server node maintaining the unified global map.

Reference frame management ensures accurate global positioning, incorporating sliding-window averaging for stability. Dense reconstruction integrates RGB-D sensor data using Voxblox [20], yielding an SDF representation of the environment for planning. The map overlay is visualized in HoloLens, providing an accurate holographic representation of the 3D point cloud map in the real world.

The drag-and-drop robot control interface allows users to manipulate holograms of robots in a 'mini-map' mode. The interface represents each robot with two holograms, one tracking its current position and another manipulable by the user. Viability checks and path planning ensure that goal poses are feasible, considering robot-specific constraints.

The user study evaluates the proposed drag-and-drop interface against the RViz interface through user tasks and NASA TLX [14] scores, showing that the HoloLens is more effective for complex tasks with multiple robots.

Real-world tests validate the system's functionality with a quadrupedal robot and showcase successful robot control through the drag-and-drop interface.

Likewise, the work of M. Ostanin and A. Klimchik [21] employs HoloLens and hand gestures for controlling the goal poses of both a six-axis and a seven-axis robot arm, catering to human assistance in industrial production settings. In this setup, operators use AR glasses to directly set virtual goal positions and orientations. Once defined, the system simulates the robotic arm's movement within the HoloLens. Upon operator satisfaction and confirmation, the real-world robotic arm executes the specified action. The operator can select from various path options, including point-to-point, line, arc, and custom paths, with the choice of collision avoidance. Additionally, tools attached to the robot can be activated either during the pose transition or at the final position.

Efforts to attain similar objectives have been made using AR glasses, predominantly HoloLens, to enhance human-robot interaction, as demonstrated in the works of [9, 10, 12, 13, 27].

3. Methodology

This section outlines the method used to build the system, including the robot setup via a ROS environment, two-way communication between Unity and ROS, and the setup of the Magic Leap interface via a Unity app.

Figure 2 shows an overview of the system and its interactions. The robot operates in the real world (top left), where the sensor data and actuator commands are represented in the ROS environment (bottom left). Sensor data from the robot in the real world in the form of 3D point clouds and robot poses are communicated to Unity to be displayed in the virtual world (top right) that is represented in the ML2 headset (bottom right). Interactions by the user in the virtual world via the ML2 controller are communicated to the ROS environment to be translated into robot commands in

the real world. This two-way communication between ROS and the ML2 device (through a Unity app) is facilitated by the TCP Connector [25].

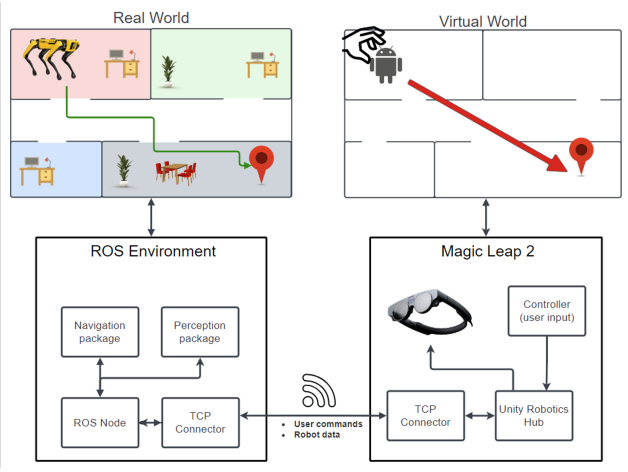

Figure 2. Overview of the system

3.1. ROS Environment Setup

A realistic robot setup is essential to demonstrate the functionality of the system. Using a ROS environment, navigation and perception can be managed via a range of configurable topics, actions, and services.

Navigation means converting control inputs to send appropriate actuator commands to the robot, to facilitate position and velocity control, the former requiring path-planning. Perception means using sensor data to visualize and gather information about the environment, allowing for better decision-making. Once integrated, both of these packages will be communicating directly with the Unity app; navigation will receive control inputs from Unity, while perception will send visualization data to Unity.

These features were first tested in a simulation using the CHAMP open-source development framework [15] with the Boston Dynamics Spot configuration [16]. Next to simple velocity control via a ROS message, this allows for position control via calling a ROS action, using an internal path-planning algorithm to let the robot navigate to the desired goal pose. Using various sensors, the simulation also collects relevant perception data. The simulation environment in Gazebo and sensor data visualization in RViz are shown in Figure 8 and Figure 9, respectively. As can be seen, obstacles in the simulated world are detected by the laser scanner and visualized, helping the robot self-localize within the environment. This 2D scenario is a useful unit test case before expanding it to 3D point clouds and integrating it with the other components.

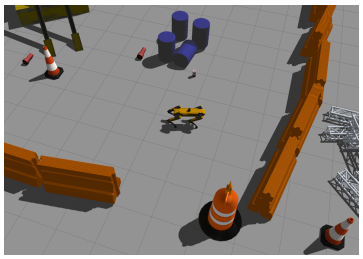

Figure 3. Simulation environment in Gazebo

Figure 4. ROS sensor data in RViz

Following simulation tests, to properly demonstrate the system and test reliability on real hardware, it was deployed on a real Boston Dynamics Spot [11] robot. Here, the included perception packages of the robot were used to retrieve a real-time 3D point cloud of the environment. For position control, a lightweight 3D RRT [17] path-planner has been re-used from a previous project.

3.2. Unity-ROS Communication

A robust translation layer is required to convert ML2 controller commands into formats compatible with the ROS system as well as to convert ROS sensor data into formats compatible with Unity to be displayed in the ML2 headset. To facilitate this communication between the Unity application and the ROS master, the *ROS-TCP-Endpoint* [26] and *ROS-TCP Connector* [7] (version used is a fork from ROS-TCP Connector repo made by Unity Technologies [25]) packages were utilized. These two packages work in tandem to bridge the communication between the two.

On the ROS side, the *ROS-TCP-Endpoint* package was imported into the catkin workspace. This sets up a server that picks up messages and data being sent from Unity. It is set with an IP address and port number and calling *roslaunch*.

In the Unity application, the *ROS-TCP Connector* package was imported. A *ROSConnectionPrefab* object was added to the scene to set up the same IP Address and port number as on the ROS side.

3.3. Magic Leap Interface

An intuitive user interface on the Magic Leap 2 headset is needed to show the user all relevant information at a glance and enable them to control robotic applications. With the headset on, the user observes a 3D point cloud representation of the environment along with the robot's pose, presented from a third-person perspective. To control the robot, the user either sends goal poses via drag-and-drop using the

controller or uses velocity control to maneuver the robot.

The ML2 interface is built using Unity, which enables many possibilities for developing virtual environments and provides many features for various VR/AR - devices. Using the provided packages for Mixed Reality headsets allows an easier tracking system and in-time visualization while moving around in a virtual space. Including a Magic Leap asset in the application allowed for simple management of the Magic Leap Controller and testing via an additional simulator from the Magic Leap Hub.

3.3.1 Robot Environment Visualization

To help the user operate the robot, a visualization of the robot's environment is needed. Although the system provides many different sensor visualization data, we opted to render the point cloud data for this project. The point cloud data was selected because it provides a detailed, 360°3D RGB-D visualization of the robot's environment. This reduces the computational burden on the ML2 device and the ROS-TCP connector since more information can be extracted from a single data type.

This feature was implemented by subscribing to the point cloud data topic from the robot within ROS and rendering it on the device. To render and display the point cloud data that is coming from the ROS-TCP connector, a modified version of the *Unity Robotics Visualizations* package was used [7]. Two Unity game objects were created (both of which are provided by the package). The first game object is the *DefaultVisualizationSuite*, which enables the Unity application to subscribe ROS data visualization types and their respective topics. The second game object is the *PointCloud2DefaultVizualizer*. This enables us to directly interact with the ROS PointCloud2 data and change how it is rendered in the application. To ensure that the point cloud data is rendered when the application starts, a point cloud renderer script was created that automatically enables the rendering of a specific ROS PointCloud2 topic. This script was based on a version that was used for the HoloLens [6].

Refer to Figure 5 and Figure 6 for an example of the output of this system.

3.3.2 Drag-and-Drop control

In this control mode, the user interacts with the robot by dragging and dropping a game object based on the current position of the robot. To represent the current position of the robot and the target position, they are represented by a yellow wolf and a green wolf game objects, respectively. The yellow wolf tracks the current position of the robot based on the odometry data reported by the ROS node. The green wolf is an object that the user can modify by drag-and-drop. After being picked up and dropped off, the green wolf publishes a command to the ROS topic that sets the new goal pose for the robot.

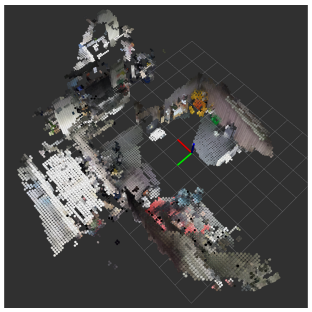

Figure 5. ROS point cloud in RViz.

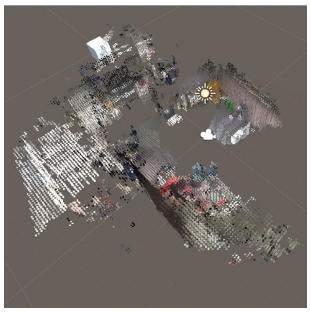

Figure 6. Point cloud visualized in Unity.

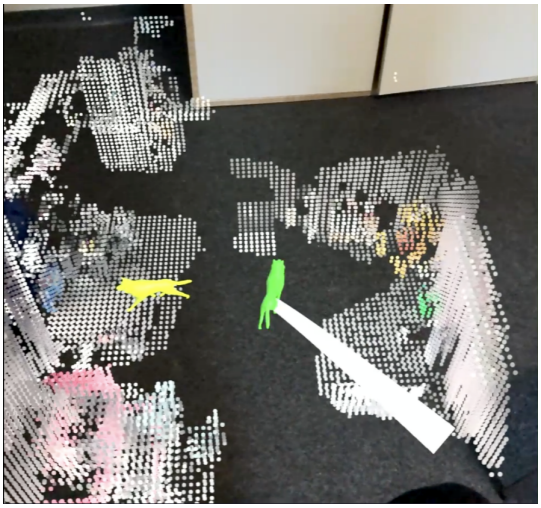

Figure 7. Demo of drag-and-drop interface. The yellow wolf represents the current position of the robot and the green wolf is the interactable object

3.3.3 Velocity Control

For this interaction, the user controls the robot by using the hand-held Magic Leap controller. By pulling the trigger on the controller and moving their finger on the touchpad, the user controls the speed and direction that the robot is moving. The trigger activates the mode and signals to the system that commands are being received, preventing unwanted commands from being broadcasted. To rotate the robot, the bumper is pressed, which rotates the robot in a

counter-clockwise direction.

4. Results

This section describes the user study that was performed to evaluate different kinds of user interactions for controlling the robot, the real-world tests to maneuver the robot, and finally, the outcome of the study and the test.

4.1. User Study

A user study was conducted to evaluate different methods of how a person can interact with the robot. As described in Section 3.3, two interaction types were created: ”Drag-and-Drop” and ”Velocity Control”. The ”Drag-and-Drop” interaction was inspired by a similar type of feature in the work of Chen et al. [8]. However, because the ML2 device comes with a remote controller and does not have onboard hand-detection features like the HoloLens, this interaction was modified to fit this. The ”Velocity Control” interaction was developed because the ML2 comes with a controller. This uses the controller's touchpad, where the robot can be directly controlled similar to using a remote controller. We believe this could provide another way to more easily control and interact with ROS-based robotic systems, therefore we wanted to investigate this by including it in our user study.

4.1.1 User Study Setup

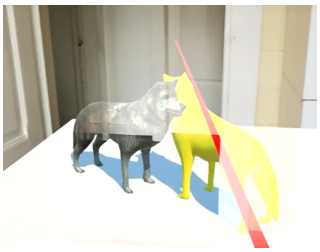

Figure 8. User study game with velocity control using the controller

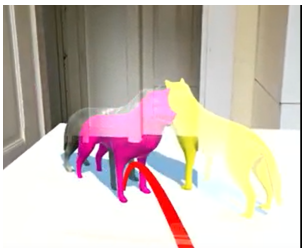

Figure 9. User study game, including the draggable pink wolf

For the setup, a game was created using Unity and deployed onto the Magic Leap device. The study was conducted independently of the real robot due to availability constraints. In the game, there are two wolves: one in yellow and gray shown. The goal of the game was to bring the yellow wolf (depicting the robot) to the position and orientation of the gray wolf. As soon as the goal position was reached, the goal wolf would jump to a new position, and the new target was set. The two different methods that were explained in the methodology section were used to control the yellow wolf. A third pink wolf was implemented for the ”Drag and Drop” option, which could be picked and placed in a different position. The yellow Wolf would then follow the pink wolf, simulating, how the real robot would walk to a position via drag-and-drop control. The participants had to try reaching 7 different positions per control method. To evaluate the two different methods, the time needed to finish the tasks was measured. Additionally, the NASA Task Load Index Questionnaire (NASA TLX) [14] and the System Usability Scale (SUS) [4] Questionnaire were used to evaluate the user experience.

The user study was conducted following the procedure: First, the users were introduced to our project and the Magic Leap. Then, the first interaction method was explained and they had time to test it until they felt comfortable. The game started as soon as they reached the first target position and from there on, the time was measured and printed out for each reached target, until the game finished. After that, the participants filled out the first two questionnaires (NASA TLX and SUS). The same procedure was repeated for the second interaction method. Finally, the participants were asked for their direct opinion, on which method they preferred and why. To avoid the influence of learning effects, the order of the tested control methods was randomized.

4.1.2 User Study Results

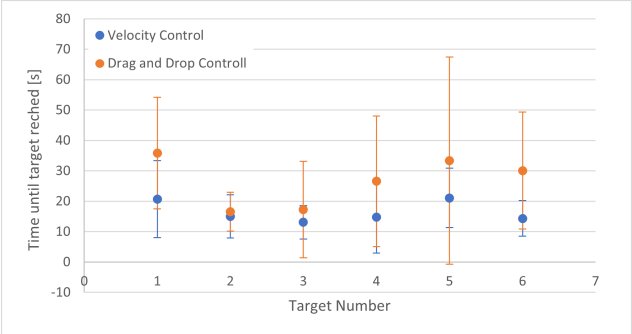

Figure 10. Average time needed for reaching each target position (level), including the standard deviation

From a total of 12 participants, there were five women and seven men aged between 21 and 27 years. According to the time measurements it can be found that the goal was reached faster using the velocity control as shown in Figure 10. From the questionnaires, it was found that the participants felt more successful in reaching the targets using the velocity control, although they felt slightly less stressed using the Drag-and-Drop control. According to the sys-

tem usability scale, their preferences between both control methods were indifferent. The final question asking for the direct preference of the participants and their reasoning, showed that five out of twelve preferred the Velocity Control over the drag and Drop. However, four of the participants argued that they would have preferred the direct control of the wolf using the velocity control if the reference frame of the wolf had not been changing when rotating it. Others argued in favor of the Drag-and-Drop with the fun of the object interaction and a less demanding feeling. For the velocity control, the participants wrote that they had to get used to it at first, but found it more intuitive afterward. they also found it easier to rotate the wolf by button press instead of rotating the controller.

4.1.3 Study Discussion

According to the results, Velocity Control was more efficient in reaching target positions due to taking less time. However, based on the feedback, the users preferred a third-person point-of-view reference frame of the controllable game object rather than the currently implemented first-person point-of-view. But this could be depending on the framing of the chosen layout. If the virtual room is rendered in its original size, and the room is slowly built in a virtual world, as in a first-person game, the preference might change. If the virtual environment of the robot is shown as a smaller version like a 3D map, the directions can be seen with a fixed coordinate frame. Although the velocity control seemed more efficient, many users preferred Drag-and-Drop. Therefore, one idea to improve user-friendliness could be to implement both control versions and let the user set their preferences in a setting section in the application.

4.2. Real World Tests

In this test, we deploy the Unity application to the Magic Leap 2 device. These are connected to a Boston Dynamics Spot robot, with an external workstation that facilitates communication between them. The external workstation brings up the ROS-TCP Endpoint and the robot. In addition, it also manages the processing of the data being streamed from the robot. The ML2 device and the external workstation are connected to the same WiFi network. The ROS-TCP connections on both side are set to the same IP address and port number.

It should be noted that at the time of writing, there are unresolved build issues that prevent the application from being fully deployed onto the device. Therefore, it needs to be connected to a computer that is running the Unity application. However, the interaction and rendering happen on the Magic Leap device itself. Additionally, due to the limited availability of the robot, at the time of writing, we were not able to test out Velocity Control on the physical robot.

4.2.1 Drag-and-Drop (Robot Position Control)

Figure 11 and the respective demo video showcase the Drag-and-Drop interactability. The point cloud data provides a rendering of the robot's environment, enabling the user to see what the robot sees. The position (represented by the yellow cylinder), showcases the position of the robot as reported by its odometry. During this test, the green object was dragged and dropped to several free points in the room, and during each time, the robot navigated to that spot with the desired position.

5. Conclusion

Mixed reality devices provide another control and interaction method with robotic systems. Our project demonstrates that robotic devices utilizing the ROS framework open up the possibility of designing more intuitive and interactive interfaces via MR frameworks, providing two-way communication between them.

The Magic Leap 2 interface provides an intuitive user experience, allowing visualization of the robot's environment in 3D point clouds and control through drag-and-drop and touchpad velocity methods. The user study results highlight the efficiency of velocity control, with potential for accommodating both methods based on user preferences. Real-world tests showcase the Drag-and-Drop interaction for robot position control. Despite deployment issues, the system demonstrates promising potential for practical applications, emphasizing the need for continued development.

Potential future extensions to this project are improving the real-time performance of visual rendering and communication. Additionally, control possibilities to enable more precise manipulation of the robot or better visualizations of sensor information could be further explored. Incorporating connectivity solutions, such as Tailscale [2], allows the operator to remotely manage and guide the robot, aligning with the principles of Industry 4.0 and the Connected Industry.

References

[1] Microsoft hololens. https://www.microsoft.com/de-ch/hololens. Accessed on January 7, 2024. 2

[2] Tailscale. https://tailscale.com/. Accessed on January 7, 2024. 6

[3] B. Bejczy, R. Bozyil, E. Vaicekauskas, S. B. K. Petersen, S. Bøgh, S. S. Hjorth, and E. B. Hansen. Mixed reality interface for improving mobile manipulator teleoperation in contamination critical applications. *Procedia Manufacturing*, 51: 620–626, 2020. 1

[4] John Brooke. Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, 189, 1995. 5

[5] C. Ulloa C, D. Domínguez, J. Del Cerro, and A. Barrientos. A mixed-reality tele-operation method for high-level control of a legged-manipulator robot. *Sensors (Basel)*, 22(21):8146, 2022. 1

[6] Jiaqi Chen. Pointcloudrender.cs. https://github.com/cvg/hololens_ros/blob/main/UnitySample/Assets/Scipts/PointCloudRenderer.cs, . 4

[7] Jiaqi Chen. Unity robotics visualizations package, . 3, 4

[8] Jiaqi Chen, Boyang Sun, Marc Pollefeys, and Hermann Blum. A 3d mixed reality interface for human-robot teaming, 2023. https://arxiv.org/pdf/2310.02392.pdf. 2, 5

[9] J. W. S. Chong, Jonathan Wun Shiung Chong, Soh-Khim Ong, Andrew Y. C. Nee, and K. Youcef-Youmi. Robot programming using augmented reality: An interactive method for planning collision-free paths. *Robotics and Computer-Integrated Manufacturing*, 2009. 2

[10] Matthew M. Cousins, Chenguang Yang, Junshen Chen, Wei He, and Zhaojie Ju. Development of a mixed reality based interface for human-robot interaction. *International Conference on Machine Learning and Computing*, 2017. 2

[11] Boston Dynamics. Spot - the agile mobile robot. https://bostondynamics.com/products/spot/. Accessed on January 7, 2024. 3

[12] Andre Gaschler, Maximilian Springer, Markus Rickert, and Alois Knoll. Intuitive robot tasks with augmented reality and virtual obstacles. *IEEE International Conference on Robotics and Automation*, 2014. 2

[13] Jan Guhl, Son Tung, and Jörg Krüger. Concept and architecture for programming industrial robots using augmented reality with mobile devices like microsoft hololens. *IEEE International Conference on Emerging Technologies and Factory Automation*, 2017. 2

[14] Sandra G. Hart and Lowell E. Staveland. *Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research*. 1988. 2, 5

[15] Juan Miguel Jimeno. Champ. https://github.com/chvmp/champ, . Accessed on January 7, 2024. 3

[16] Juan Miguel Jimeno. Champ spot configuration. https://github.com/chvmp/robots/tree/master/configs/spot_config, . Accessed on January 7, 2024. 3

[17] Steven M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. *The Annual Research Report*. 3

[18] C. Liang, C. Liu, X. Liu, L. Cheng, and C. Yang. Robot teleoperation system based on mixed reality. In *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 384–389, 2019. 1

[19] Magic Leap, Inc. Magic leap. https://www.magicleap.com/magic-leap-2, 2024. Accessed on January 7, 2024. 1, 2

[20] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1366–1373, 2017. 2

[21] Mikhail Ostanin and Alexandr Klimchik. Interactive robot programming using mixed reality. *IFAC-PapersOnLine*, 2018. 2

[22] M. Sauer, M. Hess, and K. Schilling. Towards a predictive mixed reality user interface for mobile robot teleoperation. *IFAC Proceedings Volumes*, 42(22):91–96, 2009. 1

[23] M. Sauer, F. Zeiger, and K. Schilling. Mixed-reality user interface for mobile robot teleoperation in ad-hoc networks. *IFAC Proceedings Volumes*, 43(23):77–82, 2010. 1

[24] R. Schraft and C. Meyer. The need for an intuitive teaching method for small and medium enterprises. *VDI Berichte*, 1956:95, 2006. 1

[25] Unity Technologies. Ros tcp connector. https://github.com/Unity-Technologies/ROS-TCP-Connector, . Accessed on January 7, 2024. 3

[26] Unity Technologies. Ros tcp endpoint. https://github.com/Unity-Technologies/ROS-TCP-Endpoint, . Accessed on January 7, 2024. 3

[27] Michael Walker, Hooman Hedayati, Jennifer Lee, and Daniel Szafir. Communicating robot motion intent with augmented reality. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 316–324, New York, NY, USA, 2018. Association for Computing Machinery. 2

[28] M. E. Walker, H. Hedayati, and D. Szafir. Robot teleoperation with augmented reality virtual surrogates. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 202–210, 2019. 1

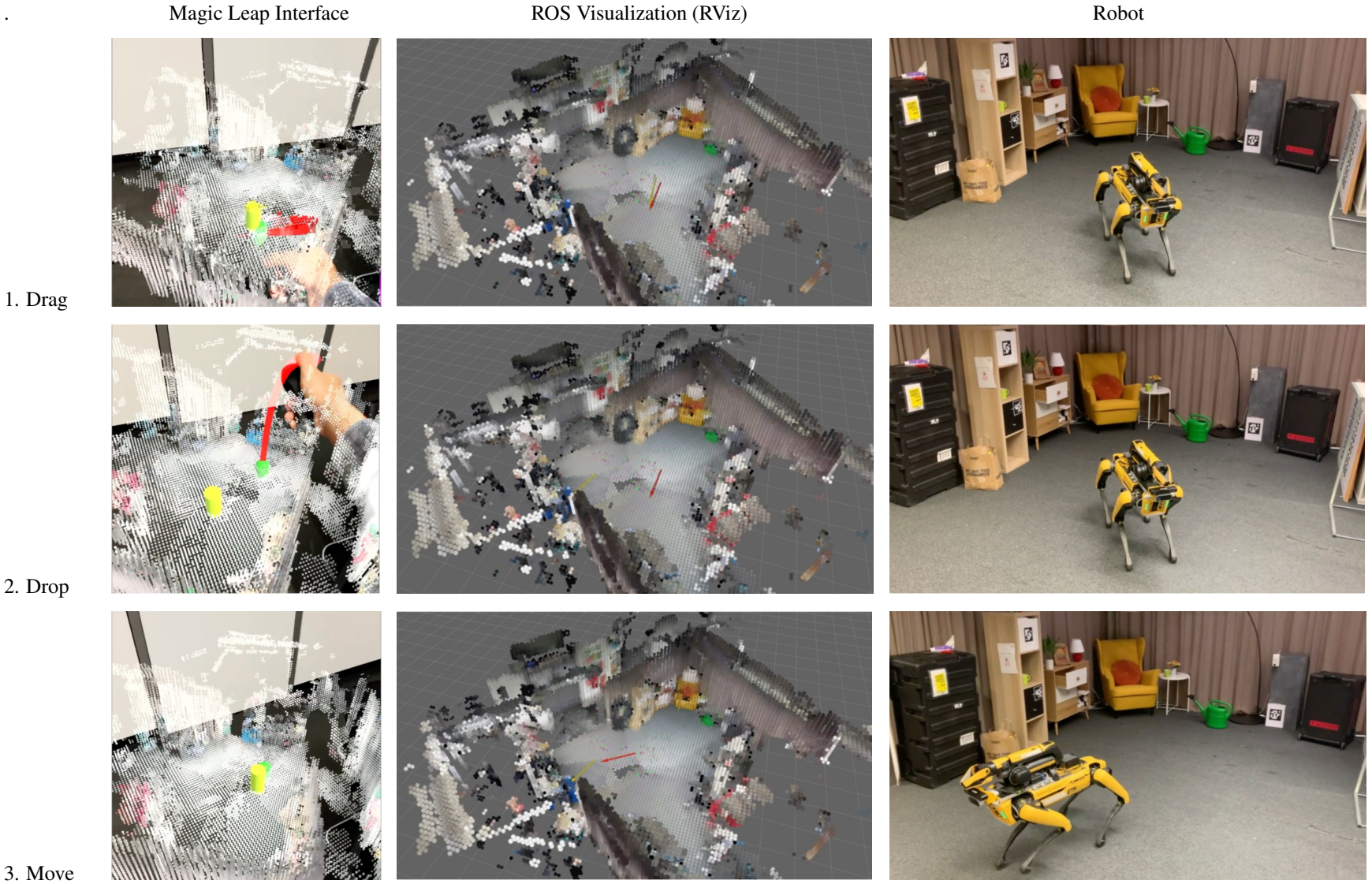

Figure 11. Demonstration of the system. Columns: left shows the first-person view from the Magic Leap 2 device, middle shows the visualization of the point cloud and pose arrows in RViz, and right column shows the robot pose in the real world. Rows: left is where the user drags the goal pose, middle is where the user drops the goal pose at the desired location, and right is the movement response of the robot to the pose command.