

Article

Immersive Teleoperation of Adaptive Mobile Robots: Evaluating Human Factors and Network Resilience in Mixed Reality

Alikhan Khamidulla , Gourav Devappa Moger  and Huseyin Atakan Varol * 

Institute of Smart Systems and Artificial Intelligence, Nazarbayev University, Astana 010000, Kazakhstan; khamidulla.alikhan@gmail.com (A.K.); gourav.devappa@nu.edu.kz (G.D.M.)

* Correspondence: ahvarol@nu.edu.kz

Abstract

Efficient robotic teleoperation for industrial inspection requires interfaces that maximize intuitive control and situational awareness. While traditional mixed reality (MR) systems offer alternatives, standard controller-based methods lack environment customization and demand external peripheral hardware. This study introduces a peripheral-hardware-free, customizable 3D spatial virtual cockpit application deployed on a Meta Quest 3 headset for long-distance, non-line-of-sight teleoperation of the Improbability Roller-2, a mobile robotic platform that dynamically adjusts its wheel geometry to traverse varied terrain over a Virtual Private Network (VPN). The system's novel spatial cockpit architecture allows operators to scale multi-channel parameters dynamically without relying on physical hardware controllers. The framework was evaluated using transmission-quality benchmarks, an active industrial machine shop deployment, and a 20-participant user study assessing usability and cognitive load. Experimental results yielded a mean System Usability Scale (SUS) score of 82.75, corresponding to an excellent usability rating, and a low mean operator workload, with a NASA Task Load Index (NASA-TLX) score of 5.19 out of 21. Participants also rated the workspace customization feature positively, assigning it a rating of 4.6 out of 5. In terms of communication performance, the system successfully completed all inspection tasks even under severely degraded VPN conditions. These findings demonstrate that the proposed customizable 3D spatial interface provides a robust, scalable alternative to traditional controller-based setups for long-distance remote robotic inspection in real-world industrial settings.

Keywords: mixed reality; teleoperation; inspection robots; virtual reality; adaptive robots; human–robot interaction



Academic Editor: Marco Ceccarelli

Received: 29 June 2026

Revised: 2 August 2026

Accepted: 4 August 2026

Published: 6 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Rapid advancements in mixed reality (MR) technologies, which combine features of Virtual Reality (VR) and Augmented Reality (AR), together with the expanding capabilities of robotic systems, are creating new opportunities for human–robot interaction (HRI). Using MR tools in robotic systems can assist visual navigation and the visualization of results, support control and planning, facilitate the simulation of robotic platforms [1], and assist HRI, making communication more intuitive and natural [2]. Just as AR technologies are used in industrial environments to improve operator situational awareness and reduce cognitive workload [3,4], they can also support robot teleoperation by helping users overcome spatial awareness limitations and streamline complex tasks [5]. Accordingly, the use of AR and MR technologies in remote teleoperation has shown benefits in healthcare, such as

surgical procedures [6]; in industry, including metal welding [7], exoskeleton calibration [8], oil pipe inspection [9]; and in environmental monitoring [10,11].

Furthermore, researchers have developed and evaluated various technologies integrated into MR teleoperation systems. These technologies can be used to improve the visual feedback of the control system by implementing digital twins [12], reducing the latency of specialized visual aids that assist teleoperation [13], and displaying the distance between the robot and potential collisions [14] to assist operators in hazardous areas. In addition, tactile teleoperation technologies have been studied for navigating [15] and controlling [16] robots, as well as providing sensory feedback [12]. Other investigations have explored shared autonomy techniques for control transition [17], methods to adapt the motion trajectory [18], and algorithms to predict the long-term goals of the operator [19], while simultaneously examining human factors and operator presence [19–21]. To contribute to these ongoing advancements, the scope of this study focuses on the fields of human factors and presence, evaluating network architectures and visual feedback modalities during remote teleoperation.

The study of VR- and AR-based robot teleoperation has become a well-researched area in recent years. Crollo et al. [22] developed an immersive teleoperation system that connects a Meta Quest 3 headset to the CyberDog quadruped robot using the Robot Operating System (ROS 2) framework. The operator interface was based on 2D user interface (UI) elements and a first-person point of view. The authors evaluated the Quality of Service (QoS) of the proposed system, together with user-centered subjective measures. In their future work, they suggested conducting participant-level analyses by correlating technical communication metrics with subjective evaluation results, including the NASA Task Load Index (NASA-TLX) and the System Usability Scale (SUS), and extending the validation to more diverse operating conditions. Another prior study identified several key challenges in AR-based teleoperation systems, including network latency, operator cognitive load, and usability experience [5]. Recent studies have proposed methods to address these challenges. Hetrick et al. [21] investigated user experience and cognitive load during the teleoperation of a Baxter robot, including both gross and fine motor tasks. Using an HTC VIVE VR headset and physical controllers, the authors measured performance with the NASA-TLX for cognitive load and the SUS for usability, reporting low-to-moderate usability alongside moderate cognitive workload during task execution. Batistute et al. [23] evaluated the effectiveness, operational errors, and perceived task difficulty in remote robot control using MR technologies, comparing them with traditional and VR baselines. Developing an Extended Reality (XR) application in Unity to interface with a Turtlebot robot over a Local Area Network (LAN), they reported that while VR control outperformed both MR gesture-based and traditional keyboard interfaces, both MR and VR modalities generally improved overall teleoperation efficiency. Furthermore, Hernandez et al. [24] developed a master–slave VR teleoperation system, using ROS, to examine how user profiles influence perceived usability. The authors reported generally positive evaluations characterized by high usability and low cybersickness, while noting that users with engineering backgrounds or prior VR experience tended to evaluate the system more critically.

Black et al. [25] evaluated network latency in MR-based human–robot teleoperation systems. The researchers developed an MR-to-robot communication framework using Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) with the WebRTC development toolkit and analyzed network latency across a LAN over Wi-Fi and Ethernet, as well as a Wide Area Network (WAN) over cellular connectivity. Their findings demonstrated that Wi-Fi and Ethernet-based LANs sustain high-frequency teleoperation control, whereas mobile networks introduce substantial propagation delays. Alternatively,

Noguera et al. [26] conducted a comprehensive analysis of delays in Wireless LAN and Virtual LAN (VLAN) configurations, concluding that a VLAN is significantly better suited for robotic teleoperation due to an approximate 200 ms reduction in network delay. Beyond transmission delays, expanding teleoperation architectures to WANs introduces critical security vulnerabilities.

Another critical feature of MR robotic teleoperation is the visualization of the robot's perspective. Stotko et al. [27] developed a live, Simultaneous Localization and Mapping (SLAM)-based reconstruction stream from a mobile platform to a VR system, allowing operators to explore the environment independently of the robot's physical camera view. The authors emphasized that this method provides a higher degree of situational awareness. Similarly, Rosa-Garcia et al. [28] developed a remote AR disinfection system featuring Virtual-SLAM (VSLAM) visualization and a ROS-based remote architecture. Furthermore, Walker et al. [29] introduced the "Cyber-Physical Control Room," an immersive MR interface that combines live HD 360-degree stereo video with real-time dense 3D point clouds, enabling operators to simultaneously use both robot-egocentric and robot-exocentric perspectives. Their results demonstrated that this dual-perspective approach improves navigation effectiveness by 28% while significantly enhancing social engagement and team cohesion between remote operators and on-site partners. A summary of these VR/AR-based teleoperation systems and their underlying technologies is provided in Table 1.

Table 1. Summary of VR/AR Teleoperation Systems for Robotic Systems

Work	Robot Platform	Control Type	Robot Vision Representation	Communication Technology	Connection Environment
Hetrick et al. [21]	Baxter robot for pick and place tasks	Controller buttons	Point cloud and dual video streams	ROS Reality Bridge	Internet
Batistute et al. [23]	Turtlebot2i	Hand gestures and controller joysticks	A video stream	ROS Bridge	LAN
Hernandez et al. [24]	FANUC LR Mate 200iD/7L manipulator	Controller buttons and virtual manipulator	A video stream	ROS-TCP Connector	LAN
Rosa-Garcia et al. [28]	ROSMaster X3 Plus inspection robot	Controller handles	VSLAM	ROS	LAN
Walker et al. [29]	Quadruped Boston Dynamics robot	Keyboard and controllers	Point cloud stream	ROS-TCP	LAN
Fan et al. [12]	Dobot Magician manipulator	VR controller and Geomagic Touch haptic device	A video stream	WebSocket	-
Livatino et al. [13]	Bonnet inspection robot with drone assist	Controller buttons and joysticks	Single video stream with synthetic visual aids and drone view	ROS Network	-
Our work	Improbability Roller-2 hybrid mobile robot	Immersive hand-gesture interaction with a customizable cockpit	Dual wide-angle video streams	Low-level TCP and UDP sockets	Cross-Network VPN

Despite recent advancements and the broad availability of technologies supporting MR-based teleoperation, frameworks that integrate these components into a fully cohesive,

immersive environment remain limited. Prior studies [10,11,21,24] have examined the usability of MR interfaces using physical controllers, while Batistute et al. [23] developed a fully immersive system. However, these approaches lacked a simultaneous and comprehensive evaluation of usability and cognitive workload. Previous studies have primarily investigated MR-based teleoperation for conventional mobile robots operating with fixed locomotion mechanisms. However, adaptive robotic platforms capable of dynamically reconfiguring their wheel geometry introduce substantially different interaction requirements, as operators must continuously control both robot navigation and locomotion adaptation within geometrically constrained inspection environments. Overcoming these spatial constraints requires adaptive robotic platforms, yet teleoperating such variable-configuration systems introduces significant control and interface complexities. Furthermore, a research gap remains in the development of customizable immersive MR workspaces that naturally integrate adaptive robot control into a unified interaction environment. In particular, the usability of MR interfaces that enable real-time wheel-size modulation, together with conventional locomotion commands, has not been sufficiently investigated. Additionally, multiple frameworks [23,24,27–29] teleoperate mobile platforms exclusively over a LAN, inherently restricting the maximum operational range. Conversely, while Black et al. [25] implemented WAN control, their architecture relied on third-party cloud servers and was scoped for short-range telemedicine operations.

To address these limitations, this study presents a fully immersive MR application and an end-to-end remote teleoperation platform to perform various inspection tasks with adaptive mobile robots, as illustrated in Figure 1. Rather than focusing solely on robot navigation, the proposed interface integrates robot locomotion and morphological adaptation into a unified virtual workspace, enabling operators to interact with adaptive mechanisms through intuitive bare-hand gestures while maintaining immersive situational awareness. Specifically, this research addresses the following core research questions:

1. **RQ1:** How does a customizable and fully immersive MR environment influence system usability and operator cognitive load during teleoperation of an adaptive inspection robot capable of dynamically changing its wheel diameter?
2. **RQ2:** Can the QoS of an MR-based teleoperation system maintain sufficient operational stability to complete inspection tasks under highly variable network conditions?

The main contributions of this study are summarized as follows:

- **Customizable Immersive MR Environment:** The design of a fully immersive and customizable MR workspace that enables natural bare-hand interaction and reproduces a virtual inspection cockpit for adaptive robot teleoperation.
- **Adaptive Locomotion Interface:** The design of a usable and efficient MR interaction technique for wheel-size adjustment and adaptive steering, allowing the robot to dynamically reconfigure its locomotion for inspection tasks in complex, geometrically constrained environments.
- **Low-Level Communication Architecture:** The design and implementation of a direct, secure one-to-one communication framework between the robot and the MR application utilizing low-level TCP/UDP socket streaming over a Virtual Private Network (VPN).
- **Linearized Visual Feedback System:** The development and evaluation of a linearized, pseudo-360-degree video streaming module utilizing a low-latency capture pipeline and Kannala–Brandt distortion correction to optimize situational awareness.
- **System and Human-Factors Evaluation:** A dual-method experimental evaluation quantifying both network communication performance (QoS metrics across LAN, WAN, and cloud VPN architectures) and human factor outcomes (usability by SUS, workload index via NASA-TLX, and user profile impact) across 20 participants.

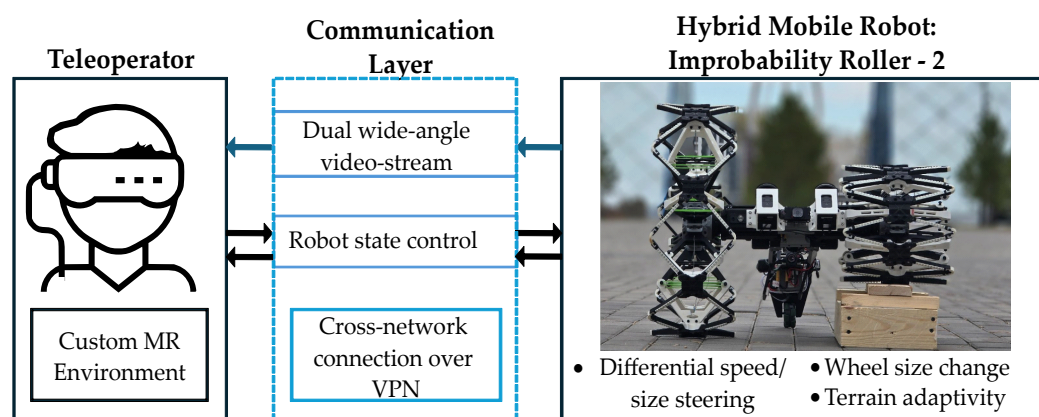


Figure 1. System architecture of the proposed MR-based teleoperation framework for remote robotic inspection.

2. Materials and Methods

2.1. System Design

The system was developed to control the Improbability Roller-2 [30], an improved iteration of the original Improbability Roller-1 platform [31]. The Improbability Roller-2 is a specialized mobile robotic platform capable of dynamically altering its wheel diameter to adapt its locomotion configuration to varying terrains. This adaptive wheel mechanism, which dynamically scales the wheel diameter from 275 mm to 515 mm, is uniquely suited for remote inspection tasks, as it allows the platform to alter its physical clearance to navigate confined structural spaces or overcome unpredictable obstacles within industrial environments. The robot is actuated by four integrated geared motors (Dynamixel Pro H42P series), connected to the Jetson Orin NX (16 GB RAM) embedded single-board computer via an RS-485 serial communication interface. Two of these motors drive the wheels, while the remaining two actuate the internal link mechanism responsible for modifying the wheel size. A low-level motor control pipeline was developed in C utilizing the Dynamixel Software Development Kit (SDK) [32] and deployed on an NVIDIA Jetson Orin NX. Additionally, the robot is equipped with two wide-angle fisheye cameras interfaced directly with the same Jetson onboard computer. Motion control commands are streamed wirelessly from the custom MR application running natively on a Meta Quest 3 head-mounted display via established TCP and UDP socket connections. The comprehensive end-to-end system architecture is illustrated in Figure 2.

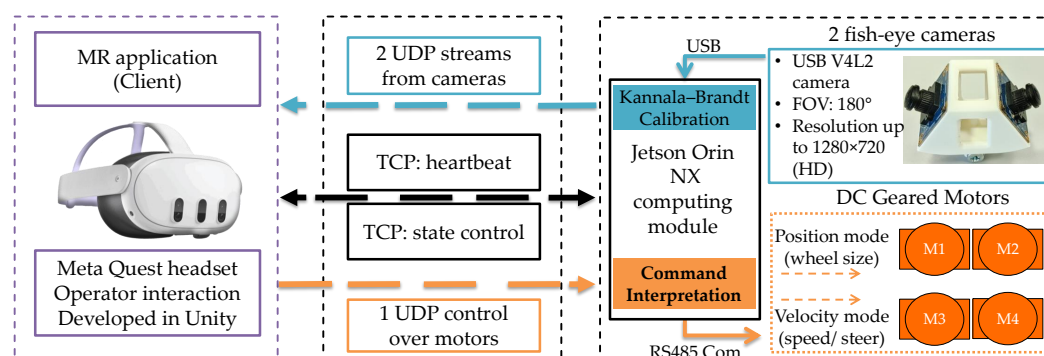


Figure 2. Communication and data transformation architecture of the teleoperation system. The layout illustrates the network connection between the Meta Quest MR application and the onboard computing module, detailing the parallel UDP video streams, TCP control channels, and peripheral hardware interfaces.

2.2. Adaptive Robot Control Framework

The mobile platform supports longitudinal translation, in-place rotation, differential steering with speed and size, and dynamic wheel-size adjustment. Longitudinal movement and in-place rotational maneuvering are achieved by a pair of primary drive motors operating in velocity control mode, which apply synchronized or equal-and-opposite target velocities to the wheels.

To execute differential steering, this velocity-controlled pair scales its output based on both the operator's baseline speed input (v) and the wheels' active physical configuration. Specifically, to account for both active driving transitions and stationary turning conditions, the control logic determines the final target wheel velocities using these parameters alongside a normalized steering coefficient ($k \in \mathbb{R}$, where $-1 \leq k \leq 1$) through the following relationships:

$$v_L = v(1 - k) \quad (1)$$

$$v_R = v(1 + k) \quad (2)$$

where v_L and v_R are the calculated target velocities for the left and right motors, respectively.

In addition, two auxiliary motors, operating in position control mode, adjust the wheel diameters. The wheel size can be modified symmetrically to alter the robot's overall clearance, or asymmetrically to assist with alternative directional steering. In the asymmetric steering mode, a directional command selectively contracts one wheel while maintaining the default diameter on the other. This mechanical discrepancy produces unequal effective linear velocities across the chassis, generating passive turning curvature even when the drive motors maintain identical angular speeds. This geometric variation is mathematically structured as follows:

$$R_L = \begin{cases} R_L^0 + (R_L^{max} - R_L^0)k, & \text{if } k < 0 \quad (\text{Left Turn}) \\ R_L^0, & \text{if } k \geq 0 \end{cases} \quad (3)$$

$$R_R = \begin{cases} R_R^0, & \text{if } k \leq 0 \\ R_R^0 - (R_R^{max} - R_R^0)k, & \text{if } k > 0 \quad (\text{Right Turn}) \end{cases} \quad (4)$$

where R_L and R_R are the active operating radii of the left and right wheels, R_L^0 and R_R^0 represent their baseline unactuated default radii, and R_L^{max} and R_R^{max} represent the maximum size of the wheels, respectively. This design choice permits precise directional control by updating the effective physical wheel contact geometry, resulting in differential motion without requiring modifications to the baseline drive motor velocities.

2.3. Mixed Reality User Interface and Operator Workflow

The operator-side interface of the teleoperation system is based on MR features and was created using the Unity real-time development engine [33]. This engine facilitates the generation and management of a virtual 3D space into which immersive interactive assets are integrated. To simulate a physical control station, custom-designed 3D models were imported into the environment to establish a virtual cockpit. These models included a table, a control lever, a steering wheel, interactive buttons (for steering mode change, emergency stop, and snapshot), a wheel-size adjustment slider, and a console featuring dedicated torque on/off buttons. Through these components, the application registers operator commands for longitudinal translation, in-place rotation, differential steering, active wheel-diameter transformation, and motor-state activation. These custom virtual cockpit elements, shown in Figure 3, were created in the SolidWorks 2024 design software and exported in .glb format. While most of the models were exported as single blocks

with additional color properties, the steering wheel and lever models were exported as separate blocks. The steering wheel consisted of the wheel itself and its base, while the lever included the base, lever stick, and two lever handles, all exported separately.

All interactions with these virtual components are executed via bare-hand tracking gestures, eliminating the need for peripheral handheld controllers. This hand detection pipeline and associated MR features native to Meta Quest head-mounted displays were implemented using the Meta XR All-in-One SDK [34]. The SDK provides classes such as `OVHand.cs` for retrieving hand-tracking data, including hand state, confidence, and tracking status; `OVRSkeleton.cs` for accessing hand joint and bone transformations; `HandVisual.cs` for rendering the virtual hand model; `HandGrabInteractor.cs` for detecting and performing hand grab interactions; `HandGrabInteractable.cs` for defining objects that can be grasped by hand; `HandGrabPose.cs` for specifying predefined hand poses during grasping; `PokeInteractor.cs` for detecting fingertip touch interactions; `RayInteractor.cs` for enabling distance-based object selection; and `HandFilter.cs` for filtering interactable objects for hand interactions.

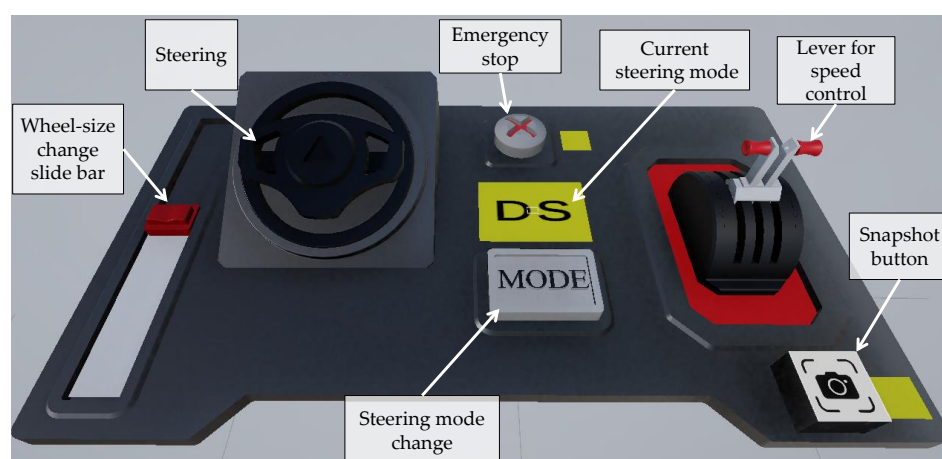


Figure 3. The virtual cockpit and customizable control layout within the Mixed Reality environment.

The interaction with the custom virtual cockpit elements was implemented using dedicated Unity scripts. The steering wheel interaction consists of several stages. First, the steering wheel object was configured with the `HandGrabInteractable` component from the Meta XR SDK to detect whether one or both hands were grasping it.

During each update cycle, the current hand orientation is compared with the orientation recorded in the previous frame. The angular velocity of the hand is first estimated to reject sudden tracking artifacts and unrealistic hand movements. The steering wheel rotation is then computed only from the change in the *hand* rotation ($\Delta\theta$) around the local *y*-axis:

$$\Delta\theta = \text{DeltaAngle}(\theta_{t-1}, \theta_t), \quad (5)$$

where θ_{t-1} and θ_t denote the hand rotation angle around the *y*-axis in the previous and current frames, respectively, and `DeltaAngle( $\cdot$ )` computes the shortest signed angular difference.

The steering *wheel* (Θ_t) orientation is subsequently updated as

$$\Theta_t = \begin{cases} \Theta_{t-1} + \Delta\theta, & \Theta_{\min} \leq \Theta_{t-1} + \Delta\theta \leq \Theta_{\max} \\ \Theta_{t-1}, & \text{otherwise} \end{cases} \quad (6)$$

where Θ_t is the steering wheel rotation, while $\Theta_{\min}$ and $\Theta_{\max}$ define the allowable steering limits.

To ensure stable robot control, the steering wheel was additionally augmented with the custom *WheelSender.cs* class. Instead of directly transmitting the measured rotation, the steering angle is first smoothed using linear interpolation:

$$\Theta_t^s = (1 - \alpha)\Theta_{t-1}^s + \alpha\Theta_t, \quad (7)$$

where Θ_t^s is the smoothed steering angle and $\alpha = \Delta t \cdot s$ is the interpolation coefficient determined by the frame time Δt and the smoothing factor s .

The filtered steering angle is transmitted to the robot only when the change exceeds a predefined dead-zone threshold while remaining below the maximum admissible variation:

$$D_{\min} < |\Theta_t^s - \Theta_{t-1}| < D_{\max}, \quad (8)$$

where $D_{\min}$ suppresses insignificant fluctuations caused by hand tremor and sensor noise, whereas $D_{\max}$ rejects abrupt angular changes that could produce unsafe robot commands. Finally, the validated steering angle is transmitted to the robot using the dual-protocol communication pipeline described in Section 2.4.

Lever control, responsible for regulating the robot's longitudinal movement, required a more sophisticated interaction pipeline. The virtual lever model consisted of three parts, with only the upper handle configured with the *HandGrabInteractable* component to detect user-grasping events. The selected handle was additionally attached to the custom *TimeManagedLeverControl.cs* script, which manages the complete interaction logic.

Similarly to the steering wheel, the hand movement is first validated by estimating the hand linear velocity. The instantaneous hand speed is computed as

$$v_h = \frac{\|\mathbf{p}_t - \mathbf{p}_{t-1}\|}{\Delta t}, \quad (9)$$

where $\mathbf{p}_t$ and $\mathbf{p}_{t-1}$ denote the current and previous hand positions, respectively. If v_h exceeds a predefined threshold, the interaction is immediately terminated to prevent unintended lever manipulation caused by rapid tracking artifacts or abrupt user movements.

The validated hand positions are then transformed from the world coordinate system into the lever local coordinate system. The local displacement along the x -axis is converted into the lever rotation increment according to

$$\Delta\phi = -\frac{\Delta x}{\Delta t} \cdot \frac{1}{60}, \quad (10)$$

where Δx is the local hand displacement along the lever axis. The lever rotation is subsequently updated as

$$\phi_t = \text{Clamp}(\phi_{t-1} + \Delta\phi \cdot k, \phi_{\min}, \phi_{\max}), \quad (11)$$

where k is the rotation sensitivity coefficient, while $\phi_{\min}$ and $\phi_{\max}$ define the allowable mechanical limits of the virtual lever.

Finally, the resulting robot speed command is obtained by linearly scaling the lever rotation,

$$v_r = \phi_t \cdot c, \quad (12)$$

where c is a calibration coefficient converting the lever rotation into the robot longitudinal velocity command. Whenever the lever is released and its rotation falls within a predefined neutral dead zone, the lever automatically returns to its zero position and a zero-speed command is transmitted to the robot through the communication pipeline described in Section 2.4.

Other custom 3D elements were used in the same way as general 2D Unity objects. To make our 3D objects work similarly to Unity 2D UI elements, we created *CustomButton* and *CustomSlider* classes. These classes imported *OVRHand* classes from Meta to track hand interaction, and used custom methods, called on Unity's *OnTriggerEnter()* method, to change the transform position of the object along the Y and X axes, for the button and slider, respectively.

Furthermore, the environment is augmented with two curved virtual projection planes that display the video streams captured by the dual onboard fisheye cameras. These planes are arranged to partially surround the operator's viewpoint, generating a panoramic layout that expands the perceived field of view (FoV). Additionally, several UI customization features inspired by interactive video game design were incorporated to maximize ergonomic adaptability. Using dedicated sliders, operators can dynamically scale the physical dimensions of the virtual cockpit, adjust the height and position of the control lever, adjust its sensitivity, and modify the velocity limits of the drive motors. These parameters allow the operator to tailor the layout of the virtual workspace to match individual ergonomics, optimizing usability and reducing physical strain. The resulting workspace layout is illustrated in Figure 3.

The teleoperation sequence, from the operator's perspective, follows a structured series of operational steps. Initially, the operator configures the spatial parameters of the virtual cockpit to align with their specific body proportions. Following this calibration, the operator commands the robot's linear speed by translating the virtual lever model forward or backward. Concurrently, directional adjustments are input via the steering wheel asset to execute turning vectors or in-place rotations. Depending on the operational requirements, the operator can toggle between the two distinct steering modalities: differential motor velocity driving or asymmetrical wheel-size adaptation, by interacting with the same steering control wheel. Finally, the interface allows the user to make secondary system adjustments, such as symmetrical wheel-diameter scaling, engaging or disengaging motor torque, or terminating the remote connection.

2.4. Dual-Protocol Communication Architecture

Wireless communication between the onboard NVIDIA Jetson computing module and the Meta Quest 3 headset is implemented utilizing a dual-protocol socket architecture. TCP and UDP sockets are selectively deployed based on the underlying payload requirements for real-time responsiveness, transport reliability, and network overhead. As detailed in Table 2, UDP sockets handle low-latency operations, including real-time video streaming, high-frequency teleoperation control inputs, and network discovery broadcasting. Conversely, TCP connections manage connection persistence, state synchronization, and critical control commands requiring guaranteed delivery.

The communication layer was implemented using three dedicated Unity classes. The *MotorSending.cs* class manages both TCP and UDP communication with the robot controller. During initialization, it establishes a TCP connection using the *TcpClient* class, authenticates, retrieves the initial motor parameters, and starts a dedicated background thread to receive messages asynchronously via a *NetworkStream*. Robot control commands are stored in a packed *ControlUDPPacket* structure consisting of four floating-point variables: speed, representing the longitudinal movement speed; san, representing the steering angle command; prot, representing the on-the-spot rotation command; and wbr, representing the wheel-size-based rotation command. The structure is serialized into a byte array using *Marshal.StructureToPtr()* and transmitted through a *UdpClient* every 20 ms from a Unity coroutine. Incoming TCP messages are received on a background thread,

synchronized through a thread-safe queue, and processed in the Unity main thread during the *Update()* loop.

Video reception was implemented in the *CameraStreaming.cs* class using a dedicated UDP receiving thread. Each UDP packet contains a frame identifier, a chunk index, the total number of chunks, and the JPEG payload. Received chunks are stored in a dictionary indexed by the frame identifier until all fragments are available. The frame is then reconstructed by concatenating the received chunks in the correct order, queuing them for rendering, and decoding them into a Unity texture on the main thread.

Connection supervision was implemented in the *TCPHeartbeat.cs* class using a dedicated *TcpListener*. A background listening thread accepts incoming heartbeat connections and continuously monitors the associated *NetworkStream*. Loss of the heartbeat immediately closes the current connection, resets the communication state, and restarts the listening procedure for subsequent reconnections.

Initially, all robot locomotion commands were transmitted via TCP to ensure reliable delivery. However, preliminary testing revealed that this configuration introduced significant latency due to protocol overhead and packet retransmission mechanisms, which degraded teleoperation responsiveness and operational safety. Consequently, the control pipeline was redesigned to stream locomotion vectors over UDP, encapsulating operator inputs into lightweight JSON packets transmitted at a fixed frequency. This approach prioritizes low latency over transport-layer verification, ensuring that only the most recent positional payload is processed while discarding outdated packets.

The communication sequence begins with the Meta Quest 3 UDP broadcasting an encrypted secret token across the network to automatically discover and authenticate the Jetson host. Upon validation, a persistent TCP handshake establishes the session. Once connected, two wide-angle video streams are captured on the robot and streamed to the headset as raw byte arrays containing encoded image data over independent UDP streams. To continuously monitor network integrity, a single-byte TCP heartbeat signal is exchanged between the nodes every second. High-frequency locomotion adjustments—encompassing baseline speed, rotation vectors, target wheel sizes, and differential steering coefficients—are continually streamed from the virtual cockpit via UDP JSON packets. Meanwhile, discrete secondary instructions, such as enabling or disabling motor torque and session termination commands, are reliably handled via TCP JSON strings. This cross-platform framework was implemented using the native C# *System.Net.Sockets* API within Unity on the client side, which interfaces directly with low-level C socket libraries deployed on the Jetson single-board computer.

While standard robotic middleware frameworks such as ROS provide high-level abstractions for teleoperation and failure handling [35], the lightweight binary communication protocol described in this work was developed to maximize communication efficiency without relying on additional middleware layers. This design provided complete control over packet formatting and parsing, reduced software overhead, and enabled seamless interoperability with the custom C-based embedded software running on the robot.

Table 2. Network Interfaces, Protocols, and Data Formats of the Teleoperation System.

Network Implementation	Port	Protocol	Communication Data Types
Broadcast discovery server	5000	UDP	Array of characters
Front camera video stream	5001	UDP	Raw byte array
Back camera video stream	5002	UDP	Raw byte array

Table 2. Cont.

Network Implementation	Port	Protocol	Communication Data Types
Heartbeat establishment	12345	TCP	Single-byte signal
Wheel size change and motor state commands	5050	TCP	JSON-formatted strings
Movement control data streaming	5051	UDP	JSON packets

2.5. Video Streaming and Rectification

To provide visual telepresence from the mobile platform to the virtual cockpit, two wide-angle fisheye cameras with a 180° FoV were mounted on the robot chassis. The camera modules are interfaced with the onboard computer via USB and accessed at a low level through the Video4Linux2 (V4L2) API using the linux/videodev2.h C library. A custom, multi-threaded capture module was developed to configure camera registers, initialize memory-mapped hardware buffers (V4L2_MAPPED_BUFFERS), and retrieve raw image data in real time. Due to the physical mounting constraints on the chassis, the effective combined horizontal perspective was partially restricted by geometric structural occlusions. Each camera stream was transmitted as a JPEG-compressed image with a target resolution of 640 × 640 pixels and a JPEG quality factor of 85. The resulting frame dimensions and frame rate were subsequently monitored as part of the QoS evaluation.

To mitigate fish-eye lens distortion and present the operator with a rectilinear, interpretable visual field, spatial correction based on the Kannala–Brandt radial symmetric distortion model [36] was deployed. This rectification pipeline was implemented in C++ using the OpenCV framework. The intrinsic camera calibration matrix was defined with a focal length adjusted to 50% of the absolute image width, centering the principal point precisely on the sensor plane array. The distortion coefficients (D) were manually tuned as $D = [k_1, k_2, k_3, k_4] = [-0.1, 0.05, 0.0, 0.0]$. The underlying fisheye projection model is described by

$$r = \sqrt{x^2 + y^2} \quad (13)$$

$$\theta = \arctan(r) \quad (14)$$

$$\theta_d = \theta \left(1 + k_1 \theta^2 + k_2 \theta^4 + k_3 \theta^6 + k_4 \theta^8 \right) \quad (15)$$

$$x' = \frac{\theta_d}{r} x, \quad y' = \frac{\theta_d}{r} y \quad (16)$$

where (x, y) are normalized undistorted image coordinates, (x', y') are the corresponding distorted coordinates, θ is the angle between the incoming light ray and the optical axis, and k_1 – k_4 are the radial distortion coefficients. During runtime, image undistortion was performed by numerically applying the inverse mapping of this model through OpenCV's `fisher::undistortImage()` function. The resulting rectified frames were then JPEG-encoded and transmitted through the network streaming pipeline.

Finally, the incoming visual streams received by the MR client application within Unity are rendered onto two parametric curved 3D mesh components. The raw byte streams of the decoded frames are ingested asynchronously, buffered into a synchronized frame queue, and dynamically mapped onto localized Graphics Processing Unit (GPU) texture assets. These textures are bound to optimized shaders within custom Unity materials and updated every rendering loop execution cycle. This curved layout wraps around the operator's viewport, minimizing perspective distortion and expanding the user's peripheral field during teleoperation, as illustrated in Figure 4.

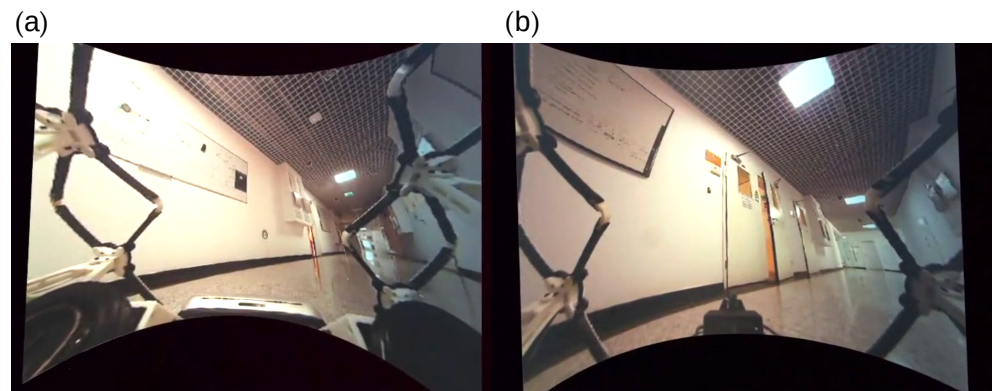


Figure 4. Views of the video stream in the Virtual Environment: (a) front and (b) back camera stream visualization.

2.6. Safety and Fault Handling

To ensure safe operation of the system during remote inspection routines, several multi-layered safety and fault-handling protocols were implemented. A secure authentication mechanism using a pre-shared token key is performed during the initial handshake to restrict host access to authorized client headsets. Following successful authentication, a persistent TCP-based heartbeat routine continuously monitors the integrity of the connection between the operator cockpit and the robot platform. In the event of a packet timeout or unexpected network disconnection, all active low-level sockets dedicated to motor actuation are immediately closed, forcing the robot control program to transition into a failsafe state that commands zero velocity to all wheel units. Additionally, an instantaneous emergency stop function is integrated directly into the primary MR UI, allowing the operator to immediately deactivate motor torque, terminate active network sockets, and safely abort the background control program.

Furthermore, a redundant secondary safety path is established via a standalone command-line utility independent of the graphics engine. If the MR rendering interface crashes or becomes unavailable, the operator can establish an out-of-band secure shell (SSH) connection directly to the Jetson onboard computer. Executing this dedicated utility immediately torques off all motors, halting the operation in unpredictable environments.

3. Experimental Setup

This study integrates secure networking architectures with MR environments to establish an immersive remote teleoperation framework for indoor inspection robots. To evaluate the platform comprehensively, diverse validation methodologies are applied across identical industry system components, combining technical QoS benchmarks with human-centric end-user experiments to gather complementary empirical data regarding performance and system usability [37]. Specifically, the experimental framework quantifies and contrasts task execution efficiency across a private cloud-based VPN and a baseline LAN configuration. Concurrently, a structured user study involving 20 human subjects evaluates the usability, cognitive workload, and overall physical comfort of the virtual cockpit interface, focusing directly on robot path navigation, bare-hand gesture interaction, and remote task completion within the custom MR workspace.

3.1. Qos Evaluation Under Diverse Network Conditions

To validate the proposed communication architecture, we tested system performance across four network setups: a LAN, a Metropolitan Area Network (MAN) over campus Wi-Fi, and a WAN using cellular and Wi-Fi internet routing. In addition to characterizing

network behavior and comparing the QoS of VPN-based communication across different network configurations, the measurements were used to identify the most challenging communication conditions for subsequent teleoperation and inspection experiments. The experiments analyzed how varying network behaviors affect teleoperation, focusing directly on packet lag, connection drops, and video stream frames. Additionally, a VPN was integrated to secure remote data transfer between the operator cockpit and the mobile robot, permitting end-to-end routing across WAN boundaries without public IP addresses.

To optimize VPN deployment, these evaluations were executed over two communication mediums: a cellular mobile modem and enterprise building Wi-Fi. The baseline LAN experiments used a high-performance TP-Link Archer router, completely isolated from external internet traffic, to ensure peak throughput. The MAN evaluations were conducted within a dedicated private university subnet. Finally, the remote network overlay testing was conducted using Tailscale [38], a mesh VPN architecture. These specific configurations were selected to reflect realistic field deployment scenarios, ranging from controlled, single-hop local environments to fully remote operations over unpredictable public networks.

To account for temporal fluctuations in network traffic, all experiments were systematically replicated across three daily periods: early morning, characterized by low network load; mid-afternoon, during peak infrastructure traffic; and late evening. Prior to each testing iteration, baseline throughput and jitter profiles were quantified to characterize the underlying channel conditions. The technical evaluation tracked and quantified specific QoS parameters during the video stream, including frames per second (FPS) rate, compressed frame size, packet reassembly time, queue drops, dropped video frames, and mean frame dimensions. These performance metrics were chosen to capture both the real-time responsiveness of the control loop and the quality of visual perception.

Alternatively, an additional developer-led evaluation was conducted under intentionally degraded network conditions to assess the robustness of the proposed teleoperation system under worst-case communication conditions. The testing was performed three times, executing the six tasks for navigating the robot, adjusting the wheel size, and completing inspections, the same tasks described in the following subsection. During each run, the time to completion was recorded, and the number of attempts, collisions, and navigation misses were counted and compared with LAN and VPN runs under optimal network conditions. The results contribute to assessing the feasibility and operational reliability of VPN-based remote teleoperation in suboptimal network environments.

3.2. User Study and Usability Evaluation

A formal user study involving 20 human subjects (13 male and 7 female, aged between 19 and 42) was conducted to evaluate the usability of the proposed platform, the efficacy of the integrated MR interaction modalities, and the cognitive workload imposed on the operator during remote teleoperation. The sample was initially intended to be randomly selected; however, for the evaluation of VR-based teleoperation systems, participants with technical backgrounds were prioritized, as their feedback is considered more objective [24]. As a result, most of the participants (19 out of 20) were from technical fields.

The experimental evaluation combined standardized psychometric instruments with customized quantitative assessment metrics. Interface usability was measured using the SUS, while operator cognitive workload was quantified via the NASA-TLX. Additionally, participants completed a post-trial questionnaire containing six 5-point Likert-scale items designed to evaluate the ergonomic performance of the virtual cockpit control interfaces. The questionnaire asked participants to rate (1) the ease of moving the robot longitudinally with the lever, (2) their satisfaction with differential steering, (3) the clarity of the video

stream to complete tasks, (4) the accuracy of adjusting the wheel size, (5) the ease of customizing the environment to body proportions, and (6) the efficiency of the environment setup in facilitating teleoperation. The experiment was conducted following approval from the Institutional Research Ethics Committee (IREC) of Nazarbayev University. All participants provided their informed consent prior to participation. More details on ethical considerations are provided in the Institutional Review Statement and the Informed Consent Statement.

Before starting the teleoperation experiments, participants were instructed to use the MR system without connection to the robot. They were familiarized with the MR goggles, the operating system, the application itself, and each object's functionality. During the evaluation, participants were instructed to teleoperate the mobile platform through 3 tasks related to control characteristics of the adaptive robot and one section containing 3 tasks designed to replicate realistic industrial inspection routines:

1. **Basic Locomotion:** Translating the robot longitudinally along a straight path using the virtual control lever.
2. **Path Navigation:** Steering the chassis through a structured corridor layout utilizing a combination of progressive differential velocity steering and stationary, in-place rotations.
3. **Geometric Adaptation:** Negotiating a low-clearance structural barrier by symmetrically reducing the diameter of the wheel from 515 mm to 275 mm through the virtual console controls.
4. **Remote Industrial Inspection:** Three tasks involving navigating the robot to locate, read, and photograph three analog gauge meter visual targets positioned throughout the environment. To evaluate situational awareness, these targets were distributed across three distinct accessibility tiers: low-difficulty (affixed openly to a corridor wall), intermediate-difficulty (placed beneath an office table), and high-difficulty (situated at the terminus of the corridor above a heating radiator, matching the maximum viewing height of the onboard cameras).

The comprehensive spatial layout of the evaluation environment utilized for these simulated inspection tasks is illustrated in Figure 5. The evaluation consisted of the following tasks performed sequentially: (1) Forward navigation, (2) First meter identification, (3) Curved path navigation, (4) Confined space navigation, (5) Second meter identification, and (6) Third meter identification.

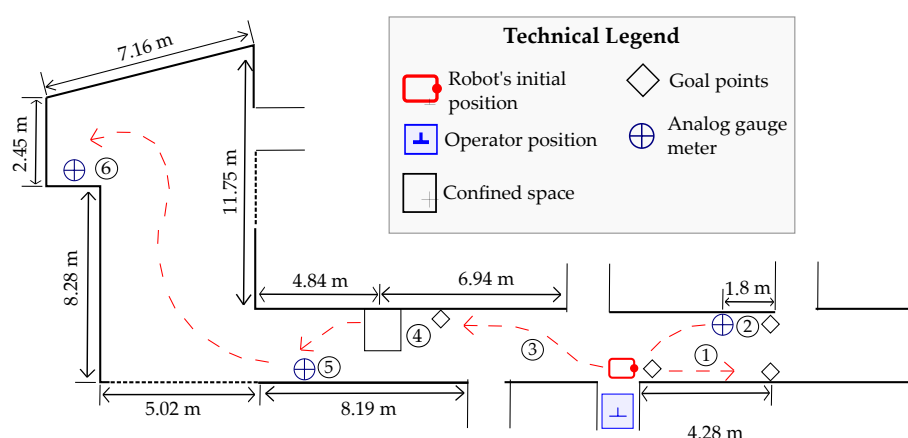


Figure 5. Floor plan and experimental layout of the simulated inspection zone. The dashed red line illustrates the robot's navigation trajectory across six sequential goal points (1–6), highlighting structural dimensions.

Beyond subjective psychometric data, the evaluation incorporated an objective, observational performance-logging framework to track task-execution accuracy and efficiency. For each trial, a researcher monitored the operator to record quantitative operational metrics, including directional steering errors (incorrect orientation inputs), collisions with surroundings, and the absolute number of discrete attempts required to successfully complete the task objectives. The absolute time to completion was simultaneously monitored to evaluate raw operational throughput.

To synthesize these objective metrics into a unified metric, a standardized task success score ranging from zero to three was established. A score of three denoted seamless task completion with full operator independence; a score of two corresponded to minor trajectory tracking errors or the receipt of minimal external guidance (one to two verbal hints); a score of one reflected frequent navigation misses or multiple procedural corrections; and a score of zero designated a total failure to complete the objective within the allocated time limit. This success score was logged for each task and subsequently aggregated across the participant pool to enable rigorous comparative statistical analysis.

4. Results

The developed system consists of a fully immersive MR teleoperation platform designed to control an adaptive inspection robot with reconfigurable wheel geometry over multiple network connections. This section presents the experimental evaluation of the system, including its usability, operator workload, communication performance, and operational effectiveness under diverse network conditions. The video results of the system and teleoperation processes, including long-distance tests of connection and inspection, are submitted as Supplementary Materials (Video S1). An inspection in a real industrial area, featuring diverse terrain conditions, obstacle configurations, and navigation to rooms with equipment and machinery, is shown in the Supplementary Materials (Video S2).

4.1. Qos Evaluation Under Diverse Network Conditions

Network characterization and video stream benchmarking were executed across three temporal daily periods utilizing four distinct infrastructure configurations: an isolated private Wi-Fi LAN, the university MAN, a Tailscale VPN overlay routed over LAN/MAN topologies, and a Tailscale VPN tunnel established over a cellular mobile modem (Cell). The consolidated empirical results are summarized in Table 3.

Table 3. Network characterization and video stream performance across three measurement sessions.

Network	Session	RTT Avg. (ms)	RTT Mdev (ms)	UDP Loss (%)	FPS Mean	Avg. Reassembly (ms)
LAN	1st	24.5	41.9	33.0	15.7	17.6
	2nd	56.4	53.7	26.0	16.3	17.6
	3rd	9.9	9.1	0.0	16.0	14.0
MAN	1st	47.2	56.9	44.0	15.9	15.0
	2nd	—(session not established)—				
	3rd	8.8	7.5	0.0	15.7	14.2
VPN	1st	200.8	80.0	28.0	14.6	31.5
	2nd	165.8	5.0	85.0	2.6	140.4
	3rd	146.0	4.4	1.0	9.2	54.3
Cell	1st	232.4	69.6	54.0	6.2	71.9
	2nd	281.8	98.6	67.0	5.4	82.2
	3rd	234.1	81.5	83.0	3.8	111.2

The primary constraint on overall system performance was Wi-Fi channel congestion, which exhibited severe temporal fluctuations that directly affected both the LAN and MAN layers. For the baseline LAN, packet loss decreased from 33% during peak morning traffic to 0% in the evening, resulting in a measured iPerf3 throughput increase from 13.4 Mbps to 19.9 Mbps. The MAN topology followed an identical performance trajectory: a 44% packet loss rate and a 10.9 Mbps throughput during morning sessions, improving to 0% loss and 20.0 Mbps throughput in the evening, though the afternoon MAN connection could not be established due to absolute network timeouts. Under low-congestion conditions, the MAN achieved the lowest mean round-trip time (RTT) across all evaluated configurations (8.8 ms, standard deviation $\sigma = 7.5$ ms during the evening), marginally outperforming the isolated LAN (9.9 ms, $\sigma = 9.1$ ms). The reported FPS values correspond to a fixed streaming configuration described in subsection 2.5.

Video stream quality closely reflected the underlying network conditions. In low-congestion evening sessions, LAN and MAN both sustained 15.7–16.0 FPS against a target of 17 FPS, with mean UDP reassembly times of 14–15 ms and fewer than two frames dropped across the entire session.

VPN performance was highly session-dependent. During the afternoon, the Tailscale client routed traffic through a distant Designated Encrypted Relay for Packets (DERP) relay node, resulting in 85% packet loss, 2.6 FPS, and a mean reassembly time of 140.4 ms. In the evening, the same VPN configuration achieved only 1% loss and full 20 Mbps throughput, yet the frame rate remained at 9.2 FPS with 1,242 frames dropped. This apparent paradox is explained by VPN-induced fragment jitter: per-packet encryption overhead increased the mean inter-arrival spread of UDP fragments to 54.3 ms (P95: 96 ms), causing the Quest reassembly buffer to time out and discard frames even when all fragments eventually arrived. Cell produced the poorest results across all sessions, with packet loss ranging from 54% to 83%, mean FPS of 3.8–6.2, and a mean reassembly time of 72–111 ms, which is consistent with the high latency jitter ($\sigma_{\text{RTT}} = 70\text{--}99$ ms) imposed by the cellular backhaul and relay routing.

To evaluate the operational limits of the proposed system, a separate developer-led experiment was conducted under severely overloaded VPN conditions. The objective was to bench-test system performance against the previously established LAN baseline and determine whether inspection tasks could still be completed under adverse communication conditions. During peak afternoon congestion, the video stream reached its lowest throughput of 2.6 FPS, representing 15.9% of the concurrent LAN baseline and 17.8% of the nominal VPN performance, while recording the maximum UDP packet loss percentage. The network configuration was evaluated through three experimental runs across six tasks and compared with LAN, the best-performing network identified during the estimation phase. As shown in Table 4, during testing, one out of three connection attempts required re-establishment. During one run, the robot collided with a wall due to delayed video stream updates, and two deviations in the navigation path were observed. The average time required to traverse the corridor increased by 47.3 s compared to the LAN baseline. Similarly, the times to locate the first, second, and third meters were higher by 14.3, 28, and 114.3 s, respectively. Despite these suboptimal results, the teleoperation process demonstrated signs of user learning, as the time required to locate the third meter decreased by 196 s between the first and final runs. Moreover, all 18 task executions were completed successfully, indicating that VPN-based long-distance teleoperation remains a viable approach for inspection tasks in constrained network environments.

Table 4. Task completion time and failures during network evaluation.

Session	Task	Collisions	Path Errors	Time (s)
Dev 1 VPN	Move robot forward	0	0	52
	Path with steering	0	0	97
	Going Under	0	0	62
	First meter	0	0	18
	Second meter	0	0	43
	Third meter	1	0	304
Dev 2 VPN	Move robot forward	0	0	29
	Path with steering	0	1	110
	Going Under	0	0	50
	First meter	0	0	18
	Second meter	0	0	43
	Third meter	0	1	220
Dev 3 VPN	Move robot forward	0	0	25
	Path with steering	0	0	95
	Going Under	0	0	50
	First meter	0	0	43
	Second meter	0	0	43
	Third meter	0	0	110
Dev LAN	Move robot forward	0	0	12
	Path with steering	0	0	53
	Going Under	0	0	30
	First meter	0	0	12
	Second meter	0	0	15
	Third meter	0	0	67

4.2. User Study and Usability Evaluation

This section analyzes operator task completion performance alongside key usability and cognitive workload metrics collected during the 20-participant user study.

4.2.1. Task Completion Analysis

The evaluation tracked operator performance across six navigation and inspection tasks. Two participants chose to withdraw from the study before completing all procedures: one following the curved path navigation task, and another after indicating the second meter. To maintain statistical consistency, data from incomplete sessions were excluded from the final metrics. The resulting task success scores and completion times are summarized in Table 5.

Table 5. Statistical Summary of Task Performance Metrics (N = 20).

Task Name	N	Avg. Success Score (0–3)	Mean Time (s)	Std. Dev. (s)	Min (s)	Max (s)
Forward navigation	20	2.90	28.90	9.60	15	44
First meter identification	20	2.60	40.40	37.00	5	151
Curved path navigation	20	2.37	114.25	48.90	57	240
Navigating confined space	16	2.75	81.19	42.90	31	157
Second meter identification	19	2.79	43.30	17.21	8	68
Third meter identification	18	2.89	142.00	43.82	90	238

The highest-scoring task was forward navigation, with a mean score of 2.9. The mean score decreased across subsequent tasks, reaching its lowest point on the curved path

navigation task. However, the success score increased across later tasks and reached a local peak of 2.89 at third meter identification as users became accustomed to the system.

Although the absolute time required to find each gauge increased, the completion time normalized by distance decreased progressively: 23.76 s per meter of travel to find the first gauge, 14.35 s per meter of travel for the second, and 11.15 s per meter of travel for the final one. The curved path navigation task was the most difficult to complete and exhibited the highest standard deviation (48.9 s), reflecting the greatest variability in operator control performance during this maneuver.

4.2.2. Subjective Usability and Workload

System usability was quantified using the SUS post-test questionnaire. The compiled survey responses yielded a mean global SUS score of 82.75, exceeding the standard industry benchmark baseline of 68 and positioning the interface within the “Good to Excellent” usability category [39]. As illustrated in Figure 6, the individual score distribution was consistently high, aligned with the specific user satisfaction ratings compiled in Table 6. Within these itemized metrics, the *Environment customization tools* and the user-perceived speed of the *System setup efficiency* features received high evaluation marks, with mean scores of 4.65/5 and 4.6/5, respectively. Locomotion mechanics, including the *Longitudinal movement (lever)* and *Wheel size adjustment* slider functions, scored averages of 4.3/5 and 4.8/5, respectively, closely matched by the positive evaluation of the *Differential steering* mechanism at 4.25/5, highlighting the effectiveness of the MR interface in controlling the adaptive wheel-size reconfiguration of the robot. Conversely, the interactive elements tied directly to variable network performance, specifically the *Video stream quality*, recorded the lowest relative baseline with a mean score of 3.75/5.

Table 6. Mean User Satisfaction Ratings for Specific System Features and Interface Components.

System Feature/Interface Component	Mean Rating (1–5)
Longitudinal movement (lever)	4.30
Differential steering	4.25
Video stream quality	3.75
Wheel size adjustment slider	4.80
Environment customization tools	4.65
System setup efficiency	4.60

Subjective workload was evaluated using the NASA-TLX, resulting in a mean global workload score of 5.19 out of 21. As illustrated in Figure 7, the *Effort* subscale recorded the highest median score (approximately 7.8) and a broad interquartile range extending from 4.0 to 13.0.

Conversely, the median scores for *Performance*, *Physical Demand*, and *Frustration* remained low, settling between 1.0 and 4.0. While the majority of participants reported minimal task-induced frustration, the *Frustration* subscale exhibited a notably wide distribution, with its upper whisker extending significantly higher than the other low-demand dimensions. Additionally, the *Temporal Demand* data shows a highly compressed distribution, mostly bounded below 7.0, with a distinct statistical outlier highlighted above the upper whisker, indicating an isolated instance of elevated time pressure during the evaluation.

Participants from robotics-related fields achieved an average task completion time of 68 s with an average success score of 2.77. Computer science participants demonstrated a comparable average completion time of 70 s, although with a slightly lower average success score of 2.57. Participants from mechanics-related fields required an average of 83 s to complete the tasks while maintaining an average success score of 2.71.

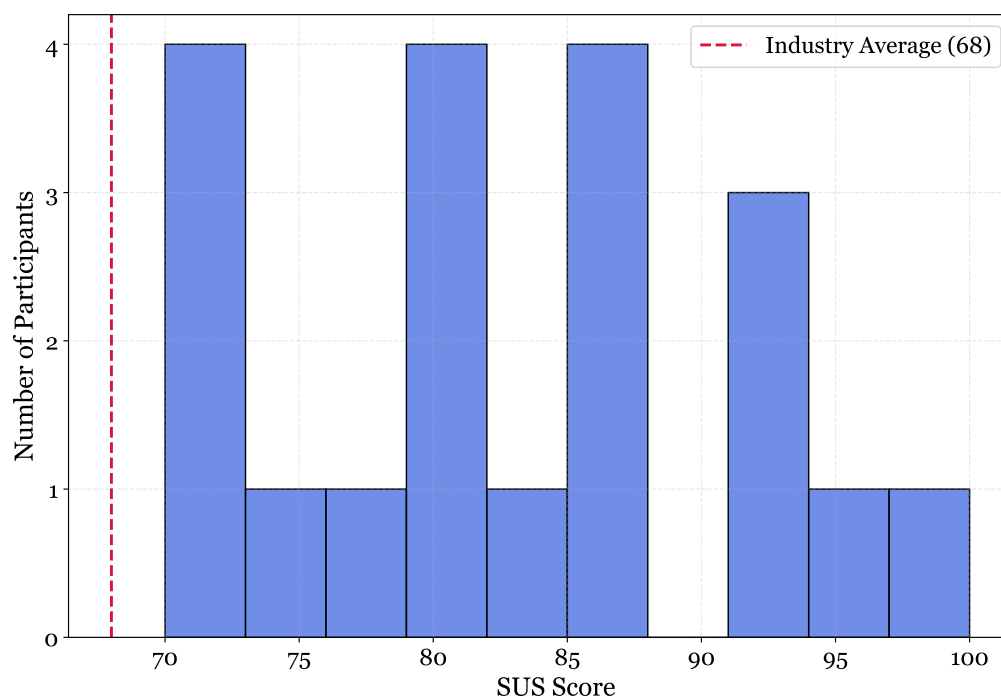


Figure 6. Distribution of SUS scores (0–100) across the participant cohort, with the dashed red line denoting the standard industry usability benchmark (SUS = 68) [39], with the proposed interface achieving a mean score of 82.75.

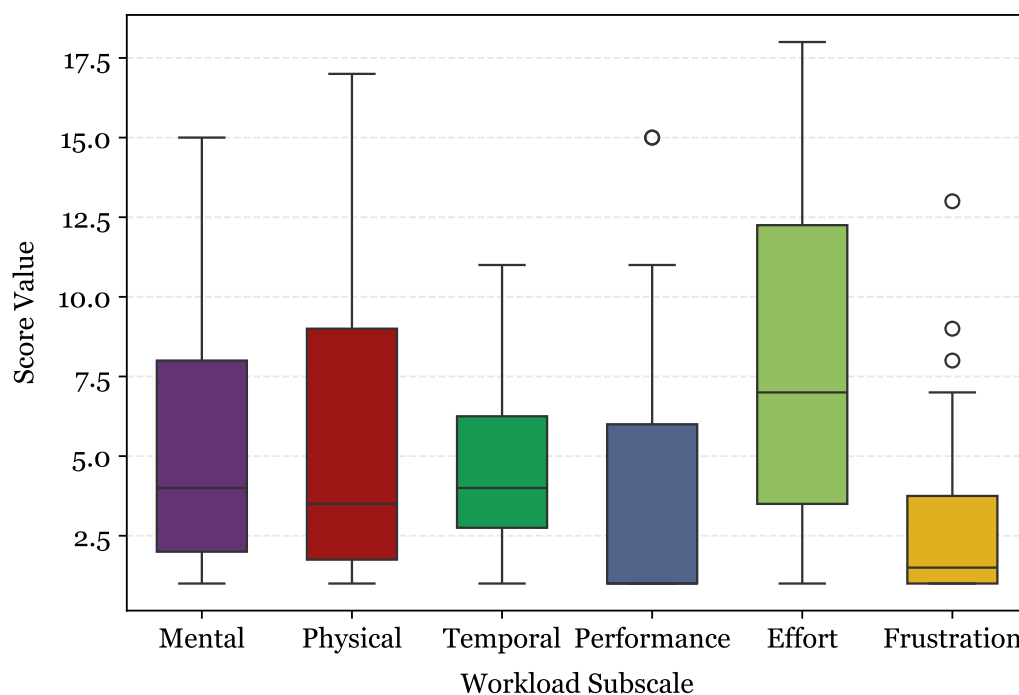


Figure 7. NASA-TLX workload evaluation results across the six core subscales. The boxes encompass the 25th to 75th percentiles, horizontal internal bars represent the medians, and the whiskers extend to $1.5 \times$ Interquartile Range, with exterior circle markers indicating individual statistical outliers.

Group differences were evaluated using one-way analysis of variance (ANOVA), with statistical significance assessed using the F-statistic. ANOVA indicated no statistically significant differences between the professional or educational background groups in terms of task completion time ($F = 1.94$, $p = 0.15$) or task success score ($F = 0.32$, $p = 0.89$).

The relationship between prior VR experience and task performance, as well as the correlation between teleoperation experience and performance, was also evaluated using Spearman's correlation coefficient (r), and statistical significance was assessed using the corresponding p -value (p). As shown in Table 7, no statistically significant correlations were observed, as all calculated p -values exceeded the threshold of 0.05. However, several non-significant trends were visible, including a weak negative correlation between VR experience and task completion time ($r = -0.32$), a weak negative correlation between teleoperation experience and completion time ($r = -0.26$), a weak positive correlation with the SUS score ($r = 0.36$), and a weak negative correlation with the NASA-TLX score ($r = -0.23$). These results indicate that the interface provided a uniform user experience across varying levels of prior operator training and domain expertise.

Table 7. Correlation Analysis Between User Experience Factors and Performance Metrics.

Experience Factor	Completion Time		Success Score		SUS Score		NASA-TLX	
	r	p	r	p	r	p	r	p
VR/MR Experience	−0.32	0.16	−0.01	0.97	0.05	0.84	0.40	0.08
Teleoperation Experience	−0.26	0.27	0.04	0.87	0.36	0.12	−0.23	0.33

4.2.3. User Profile Impact

To investigate the influence of participant background on the experimental outcomes, an additional analysis was conducted focusing on user profile characteristics, including prior experience with virtual and MR systems, robotic teleoperation, and related technical fields. The participant sample was selected to maintain a varied and relatively random distribution of technical backgrounds. As illustrated in Figure 8, the plurality of participants (7 individuals) were from robotics-related fields, while the remaining participants represented disciplines such as computer science, mechanics, cybersecurity, data science, and marketing. Overall, 19 participants had technical backgrounds, while 1 participant represented a non-technical field.

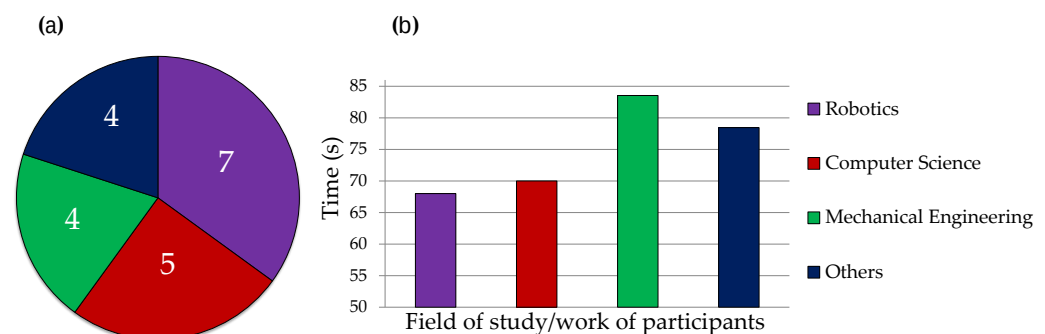


Figure 8. (a) Distribution and (b) time to completion of tasks of the participants by profession field.

5. Discussion

The primary objective of this study was to evaluate the effects of an immersive MR teleoperation architecture on operator efficiency and workload during remote robotic inspections. For this purpose, the Improbability Roller-2 platform was used to introduce more complex multi-axial control challenges, rather than relying on a conventional wheeled robot configuration. By adhering to the corroborative verification principle [37], pairing objective performance metrics with subjective human-factors evaluations provides a comprehensive diagnostic view of the system's operational viability.

Subjective evaluation of the interface yielded a mean global SUS score of 82.75, aligning with the “Good to Excellent” baseline classification. For additional context, Hetrick et al. [21] reported usability scores of 47.18 for positional keyboard control and 38.25 for trajectory-based VR controllers, whereas Crollo et al. [22] achieved a SUS score of 90.30 using an immersive VR interface with a quadruped robot and a predominantly 2D user interface. Although these studies employed different robotic platforms, interaction paradigms, and experimental protocols, they collectively suggest that immersive teleoperation interfaces generally provide high perceived usability. Therefore, these values should be interpreted only as contextual references rather than direct performance comparisons.

The qualitative feedback highlights that combining a customizable environment with immersive controls makes the system easy to use and helpful for teleoperation. Features for moving the robot, modifying wheel dimensions, and personalizing the workspace received high user evaluations, indicating that allowing operators to dynamically adjust system parameters within the interface directly supports the teleoperation workflow. The highest satisfaction score of 4.8/5 for the wheel-size adjustment functionality demonstrates the effectiveness of the developed MR system in controlling adaptive mobile robots during inspection tasks that require navigating beneath obstacles. Additionally, the high average task success rates show the system was successful during inspections, while the negative correlation between usability scores and completion times indicates that the interface helped reduce task duration. Therefore, **RQ1** is positively addressed, demonstrating that a customizable and fully immersive MR environment can effectively support both teleoperation and adaptive wheel-diameter control while maintaining high usability and low cognitive workload.

In alignment with the findings of Hernandez et al. [24], participants from different technical backgrounds demonstrated relatively comparable task performance and success rates during the MR teleoperation tasks. The absence of statistically significant differences indicates that the proposed interaction system is intuitive and adaptable across users with varying professional or educational specializations. Furthermore, prior robotic teleoperation experience showed only a weak and statistically insignificant relationship with task performance, reinforcing that the interface remains accessible to operators without specialized training. This is further supported by the NASA-TLX, which yielded a low mean global score of 5.19 out of 21. However, subscale distributions showed that the *Effort* category had the highest median workload, indicating that continuous tactical focus was required to control the vehicle’s unique multi-axial locomotion.

Looking at specific interface limitations, the steering controls received lower user feedback scores, revealing clear areas for refinement. Despite a moderate user satisfaction score for the raw video stream quality (3.75/5), the omnidirectional visual configuration received critical feedback regarding its physical layout. Participants noted that the requirement to physically rotate their heads to check the rear-view projection screen was non-intuitive, which disrupted their situational awareness during complex maneuvering tasks. To mitigate these visual tracking constraints, users suggested incorporating distinct audio-visual confirmation cues upon successful photo capture. Furthermore, participants highlighted the need to integrate hand-targeted haptic feedback to minimize the cognitive load associated with continuous visual verification during remote interactions.

However, the user study was limited by the distribution of participants, as most had technical backgrounds in engineering or computer science. Consequently, the results may not fully generalize to non-technical operators or users without prior exposure to interactive digital systems. Additionally, the small sample size of experienced VR developers and users likely contributed to the statistically insignificant correlation results observed. Furthermore, while the current evaluation provides absolute user-centered metrics, it lacks

direct relative performance comparisons against alternative control systems. Future work should incorporate comparative studies evaluating the proposed spatial cockpit interface against traditional modalities such as physical joysticks and keyboard-and-mouse setups, as well as alternative 2D interfaces, bare-hand gesture controls, and AR point-and-click tracking mechanisms.

While the VPN connection between the robot and the MR headset had lower performance compared to a direct LAN connection, it removed the geographic restrictions of local networks, which is necessary for remote inspection. Even during peak network congestion, when UDP packet loss reached 85%, operators successfully completed all 18 inspection tasks. Therefore, **RQ2** can be answered positively, as the proposed teleoperation system maintained sufficient operational stability to successfully complete inspection tasks even under highly variable and severely degraded network conditions.

Beyond standard manufacturing environments, the proposed hardware-free spatial interface framework addresses an operational need in high-consequence industries where human entry poses safety risks. Deploying a controller-free teleoperation interface via a standalone MR headset makes this system a viable alternative for hazardous operations in nuclear decommissioning, deep underground mining, oil and gas processing, and disaster response. In these settings, continuous monitoring and video feedback are mandatory to handle unexpected structural anomalies or radiation hazards without exposing personnel to danger. Furthermore, minor modifications to the transport layer would allow for integration with cellular networks rather than private VPNs. This protocol adaptability expands the utility of the framework to remote or infrastructure-compromised zones lacking local network access. Ultimately, because the core contribution focuses on the spatial interface architecture, the underlying communication network can be modified to fulfill the specific bandwidth and distance requirements of the target environment.

6. Conclusions

This study presented an MR robot teleoperation system designed to eliminate the need for external peripheral control hardware by leveraging an immersive virtual workspace. The core contribution lies in the empirical assessment of this interface during remote indoor inspection tasks across varied elevation zones, differing network topologies, and long physical distances. Quantitative testing yielded a mean SUS score of 82.75 alongside a low mean cognitive workload score of 5.19 out of 21 on the NASA-TLX. Subjective user ratings confirmed that the custom workspace configuration dashboard directly optimized operational ergonomics, earning an efficiency rating of 4.6 out of 5.0. Crucially, the evaluation demonstrated that operators successfully adapted to and navigated the control framework regardless of their teleoperation or VR experience level, proving that immersive MR interfaces can democratize robotic deployment by eliminating the need for specialized pilot training. Additionally, the system maintained stable operation over a VPN link during extreme network stress testing and all inspection missions, despite 85% UDP stream packet loss, a 165.8 ms network RTT, and a drop in video throughput to 2.6 FPS. These stress-test outcomes validate that the proposed framework is resilient enough for real-world industrial and disaster-response environments where communication infrastructure is deeply compromised.

Based on these outcomes, future research will target specific technical updates to the interface and communication pipeline. First, the visual feedback layout will replace the continuous 360° surround planes with a decoupled front-and-rear dual-camera display configuration to eliminate situational-awareness blind spots. Second, wearable tactile feedback assets, such as haptic gloves, will be developed to improve operator precision during complex multi-axial maneuvering tasks. Finally, the framework will integrate

cellular communication modules to extend geographic deployment range and decrease transport-layer latency by bypassing local intermediary routing hops.

Supplementary Materials: The following supporting information can be downloaded at <https://www.mdpi.com/article/doi/s1>, Video S1: MR Teleoperation of Adaptive Robot; Video S2: Industrial area inspection.

Author Contributions: Conceptualization, all authors; methodology, A.K., G.D.M.; software, A.K.; validation, G.D.M. and H.A.V.; investigation, H.A.V.; resources, H.A.V.; writing—original draft preparation, A.K.; writing—review and editing, G.D.M. and H.A.V.; supervision, G.D.M. and H.A.V.; project administration, G.D.M.; funding acquisition, H.A.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and approved by the Institutional Review Ethics Committee of Nazarbayev University (Submission ID:1219/30032026 approved on 15 April 2026).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The data supporting the findings of this study are not publicly available due to project-specific restrictions. However, the data may be made available to the corresponding author upon reasonable request, subject to institutional approval.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MR	Mixed Reality
VR	Virtual Reality
AR	Augmented Reality
XR	Extended Reality
SUS	System Usability Scale
NASA-TLX	NASA Task Load Index
FoV	Field of View
ROS	Robot Operating System
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
LAN	Local Area Network
MAN	Metropolitan Area Network
WAN	Wide Area Network
VPN	Virtual Private Network
VLAN	Virtual Local Area Network
SLAM	Simultaneous Localization and Mapping
VSLAM	Virtual Simultaneous Localization and Mapping
SDK	Software Development Kit
DERP	Designated Encrypted Relay for Packets
QoS	Quality of Service
HRI	Human–Robot Interaction
UI	User Interface
GPU	Graphics Processing Unit
SSH	Secure Shell
FPS	Frames Per Second
RTT	Round Trip Time

References

1. Yu, J.; Wang, T.; Shi, Y.; Yang, L. MR Meets Robotics: A Review of Mixed Reality Technology in Robotics. In Proceedings of the 2022 6th International Conference on Robotics, Control and Automation (ICRCA), Xiamen, China, 26–28 February 2022; pp. 11–17. <https://doi.org/10.1109/ICRCA55033.2022.9828895>.
2. Makhataeva, Z.; Varol, H.A. Augmented Reality for Robotics: A Review. *Robotics* **2020**, *9*, 21. <https://doi.org/10.3390/robotics9020021>.
3. Akhmetov, T.; Moger, G.D.; Varol, H.A. Augmented Reality Multistation Warning System Using Wearable Artificial Intelligence. In Proceedings of the 2025 11th International Conference on Control, Automation and Robotics (ICCAR), Kyoto, Japan, 18–20 April 2025; pp. 483–492. <https://doi.org/10.1109/ICCAR64901.2025.11072945>.
4. Akhmetov, T.; Moger, G.; Tursynbek, I.; Varol, H.A. Thermal Perception Using Augmented Reality for Industrial Safety. In Proceedings of the 2023 3rd International Conference on Robotics, Automation and Artificial Intelligence (RAAI), Singapore, 14–16 December 2023; pp. 91–96. <https://doi.org/10.1109/RAAI59955.2023.10601208>.
5. Rosa-Garcia, A.D.L.; Marrufo, A.I.S.; Luviano-Cruz, D.; Rodriguez-Ramirez, A.; Garcia-Luna, F. Bridging Remote Operations and Augmented Reality: An Analysis of Current Trends. *IEEE Access* **2025**, *13*, 36502–36526. <https://doi.org/10.1109/ACCESS.2025.3544633>.
6. Ai, L.; Kazanzides, P.; Azimi, E. Mixed reality based teleoperation and visualization of surgical robotics. *Healthc. Technol. Lett.* **2024**, *11*, 179–188. <https://doi.org/10.1049/htl2.12079>.
7. Wang, Q.; Cheng, Y.; Jiao, W.; Johnson, M.T.; Zhang, Y. Virtual reality human-robot collaborative welding: A case study of weaving gas tungsten arc welding. *J. Manuf. Process.* **2019**, *48*, 210–217. <https://doi.org/10.1016/j.jmapro.2019.10.016>.
8. Franco, O.A.M.; Giurin, G.; Tefera, Y.T.; Di Natali, C.; Monica, L.; Caldwell, D.G.; Ortiz, J. Integrating Automatic Force Assistance Configuration with Mixed Reality for Active Exoskeletons. In Proceedings of the 2026 IEEE/SICE International Symposium on System Integration (SII), Cancun, Mexico, 11–14 January 2026; pp. 1054–1060. <https://doi.org/10.1109/sii64115.2026.11404649>.
9. Dalabekov, A.; Akimbay, D.; Akhmetov, T.; Zholtayev, D.; Yeshmukhametov, A. Mixed reality based in-pipe inspection. *Front. Virtual Real.* **2026**, *7*, 1693545. <https://doi.org/10.3389/frvir.2026.1693545>.
10. Xu, S.; Wu, L.; Liao, W.; Fujimura, S. Incorporating Drone Into Mixed Reality for Enhanced Remote Collaboration: A User Study on Inspection Task. *IEEE Trans. Vis. Comput. Graph.* **2026**, *32*, 2741–2757. <https://doi.org/10.1109/TVCG.2026.3654668>.
11. Salunkhe, S.A.; Nedunghat, P.; Morando, L.; Bobbili, N.; Li, G.; Loianno, G. Intuitive Human-Drone Collaborative Navigation in Unknown Environments Through Mixed Reality. In Proceedings of the 2025 International Conference on Unmanned Aircraft Systems (ICUAS), Charlotte, NC, USA, 14–17 May 2025; pp. 862–868. <https://doi.org/10.1109/ICUAS65942.2025.11007818>.
12. Fan, W.; Guo, X.; Feng, E.; Lin, J.; Wang, Y.; Liang, J.; Garrad, M.; Rossiter, J.; Zhang, Z.; Lepora, N.; et al. Digital Twin-Driven Mixed Reality Framework for Immersive Teleoperation With Haptic Rendering. *IEEE Robot. Autom. Lett.* **2023**, *8*, 8494–8501. <https://doi.org/10.1109/LRA.2023.3325784>.
13. Livatino, S.; Guastella, D.C.; Muscato, G.; Rinaldi, V.; Cantelli, L.; Melita, C.D.; Caniglia, A.; Mazza, R.; Padula, G. Intuitive Robot Teleoperation Through Multi-Sensor Informed Mixed Reality Visual Aids. *IEEE Access* **2021**, *9*, 25795–25808. <https://doi.org/10.1109/ACCESS.2021.3057808>.
14. Szczurek, K.A.; Prades, R.M.; Matheson, E.; Rodriguez-Nogueira, J.; Castro, M.D. Mixed Reality Human–Robot Interface With Adaptive Communications Congestion Control for the Teleoperation of Mobile Redundant Manipulators in Hazardous Environments. *IEEE Access* **2022**, *10*, 87182–87216. <https://doi.org/10.1109/ACCESS.2022.3198984>.
15. Li, S.; Gao, P.; Chen, Y. A Bilateral Teleoperation Strategy Augmented by EMGP-VH for Live-Line Maintenance Robot. *IEEE Trans. Hum.-Mach. Syst.* **2024**, *54*, 362–374. <https://doi.org/10.1109/THMS.2024.3412910>.
16. Sun, P.; Li, W.; Li, J.; Liu, Y.; Wang, J.; Ding, L.; Zhou, C. More Precise and Faster: Dual-Scale Teleoperation for Manipulator in Large Workspace. *IEEE Trans. Fuzzy Syst.* **2026**, *34*, 27–40. <https://doi.org/10.1109/TFUZZ.2025.3621224>.
17. Rastegarpanah, A.; Mineo, C.; Contreras, C.A.; Shaarawy, A.; Paragliola, G.; Stolkin, R. Haptic Teleoperation in Extended Reality for Electric Vehicle Battery Disassembly Using Gaussian Mixture Regression. *J. Field Robot.* **2026**, *43*, 1130–1151. <https://doi.org/10.1002/rob.70079>.
18. Penco, L.; Momose, K.; McCrory, S.; Anderson, D.; Kitchel, N.; Calvert, D.; Griffin, R.J. Mixed Reality Teleoperation Assistance for Direct Control of Humanoids. *IEEE Robot. Autom. Lett.* **2024**, *9*, 1937–1944. <https://doi.org/10.1109/LRA.2024.3349809>.
19. SharafianArdakani, P.; Hanafy, M.A.; Kondaurava, I.; Ashary, A.; Rayguru, M.M.; Popa, D.O. Adaptive User Interface With Parallel Neural Networks for Robot Teleoperation. *IEEE Robot. Autom. Lett.* **2025**, *10*, 963–970. <https://doi.org/10.1109/LRA.2024.3518085>.
20. De Ocampo, F.O.; Hernández-Melgarejo, G.; Ramírez-Treviño, A.; Fuentes-Aguilar, R.Q. Presence Assessment in Virtual Reality: A Systematic Literature review. *Appl. Sci.* **2026**, *16*, 3102. <https://doi.org/10.3390/app16063102>.
21. Hetrick, R.; Amerson, N.; Kim, B.; Rosen, E.; Visser, E.J.D.; Phillips, E. Comparing Virtual Reality Interfaces for the Teleoperation of Robots. In Proceedings of the 2020 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, USA, 24 April 2020; pp. 1–7. <https://doi.org/10.1109/SIEDS49339.2020.9106630>.

22. Criollo, E.; Oñate, W.; Caiza, G.; Andaluz, V.H.; Varela-Aldás, J. Immersive AR–ROS 2 Teleoperation Architecture for a Physical Quadruped Robot. *Robotics* **2026**, *15*, 134. <https://doi.org/10.3390/robotics15070134>.
23. Batistute, A.; Santos, E.; Takieddine, K.; Lazari, P.M.; Giane Da Rocha, L.; Teixeira Vivaldini, K.C. Extended Reality for Teleoperated Mobile Robots. In Proceedings of the 2021 Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and 2021 Workshop on Robotics in Education (WRE), Natal, Brazil, 11–15 October 2021; pp. 19–24. <https://doi.org/10.1109/LARS/SBR/WRE54079.2021.9605474>.
24. Hernandez, M.N.; de Miranda, N.C.; Domingues, B.H.R.; Negri, D.; de Souza, D.; Secco, I.; Trabasso, L.G.; Simoni, R. Influence of User Profiles on Usability Perception in Virtual Reality-Based Robotic Teleoperation. In Proceedings of the 2025 27th Symposium on Virtual and Augmented Reality (SVR), Salvador, Brazil, 30 September–3 October 2025; pp. 156–164. <https://doi.org/10.1109/SVR67689.2025.00032>.
25. Black, D.G.; Andjelic, D.; Salcudean, S.E. Evaluation of Communication and Human Response Latency for (Human) Teleoperation. *IEEE Trans. Med. Robot. Bionics* **2024**, *6*, 53–63. <https://doi.org/10.1109/TMRB.2024.3349612>.
26. Noguera Cundar, A.; Fotouhi, R.; Ochitwa, Z.; Obaid, H. Quantifying the Effects of Network Latency for a Teleoperated Robot. *Sensors* **2023**, *23*, 8438. <https://doi.org/10.3390/s23208438>.
27. Stotko, P.; Krumpfen, S.; Schwarz, M.; Lenz, C.; Behnke, S.; Klein, R.; Weinmann, M. A VR System for Immersive Teleoperation and Live Exploration with a Mobile Robot. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 4–8 November 2019; pp. 3630–3637. <https://doi.org/10.1109/IROS40897.2019.8968598>.
28. De La Rosa-Garcia, A.; Rodriguez-Ramirez, A.G.; Robles, B.A.; Soto-Marrufo, I.; Ortiz-Muñoz, D.; Alonso-Mendoza, V.M.; Luviano-Cruz, D.; Garcia-Luna, F. AR-Based teleoperation of an omnidirectional mobile robot for UV-C disinfection. *Robotics* **2026**, *15*, 94. <https://doi.org/10.3390/robotics15050094>.
29. Walker, M.E.; Gramopadhye, M.; Ikeda, B.; Burns, J.; Szafir, D. The Cyber-Physical Control Room: A Mixed Reality Interface for Mobile Robot Teleoperation and Human-Robot Teaming. In Proceedings of the 2024 19th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Boulder, CO, USA, 11–14 March 2024; pp. 762–771.
30. Moger, G.; Varol, H.A. Improbability Roller-2: A Hybrid Mobile Robot with Variable Diameter Transformable Wheels. In Proceedings of the 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hangzhou, China, 19–25 October 2025; pp. 16489–16494. <https://doi.org/10.1109/IROS60139.2025.11246886>.
31. Moger, G.; Varol, H.A. Design and Implementation of a Mobile Robot With Variable-Diameter Wheels. *IEEE/ASME Trans. Mechatron.* **2025**, *30*, 2529–2538. <https://doi.org/10.1109/TMECH.2024.3457011>.
32. ROBOTIS. DYNAMIXEL SDK Overview. 2026. Available online: https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/overview/ (accessed on 12 May 2026).
33. Unity Technologies. Unity 6: Download the Latest Release of Unity 6 | Unity. Available online: <https://unity.com/releases/unity-6/> (accessed on 7 May 2026).
34. Meta Platforms, Inc. Develop with Unity—Meta Horizon. 2026. Available online: <https://developers.meta.com/horizon/develop/unity/> (accessed on 28 April 2026).
35. Pico, N.; Mite, G.; Morán, D.; Alvarez-Alvarado, M.S.; Auh, E.; Moon, H. Web-Based Real-Time Alarm and Teleoperation System for Autonomous Navigation Failures Using ROS 1 and ROS 2. *Actuators* **2025**, *14*, 164. <https://doi.org/10.3390/act14040164>.
36. Kannala, J.; Brandt, S. A generic camera calibration method for fish-eye lenses. **2004**, *1*, 10–13. <https://doi.org/10.1109/ICPR.2004.1333993>.
37. Fisher, M.; Cardoso, R.C.; Collins, E.C.; Dadswell, C.; Dennis, L.A.; Dixon, C.; Farrell, M.; Ferrando, A.; Huang, X.; Jump, M.; et al. An Overview of Verification and Validation Challenges for Inspection Robots. *Robotics* **2021**, *10*, 67. <https://doi.org/10.3390/robotics10020067>.
38. Tailscale Inc. Tailscale. 2026. Available online: <https://tailscale.com/> (accessed on 15 June 2026).
39. Lewis, J.R. Measuring Perceived Usability: The CSUQ, SUS, and UMUX. *Int. J. Hum.-Comput. Interact.* **2018**, *34*, 1148–1156. <https://doi.org/10.1080/10447318.2017.1418805>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.