Tele-robotics via An Efficient Immersive Virtual Reality Architecture

Nasif Zaman
Department of Computer Science and Engineering
University of Nevada, Reno
zaman@nevada.unr.edu

Alireza Tavakkoli
Department of Computer Science and Engineering
University of Nevada, Reno
tavakkol@unr.edu

Christos Papachristos
Department of Computer Science and Engineering
University of Nevada, Reno
cpapachristos@unr.edu

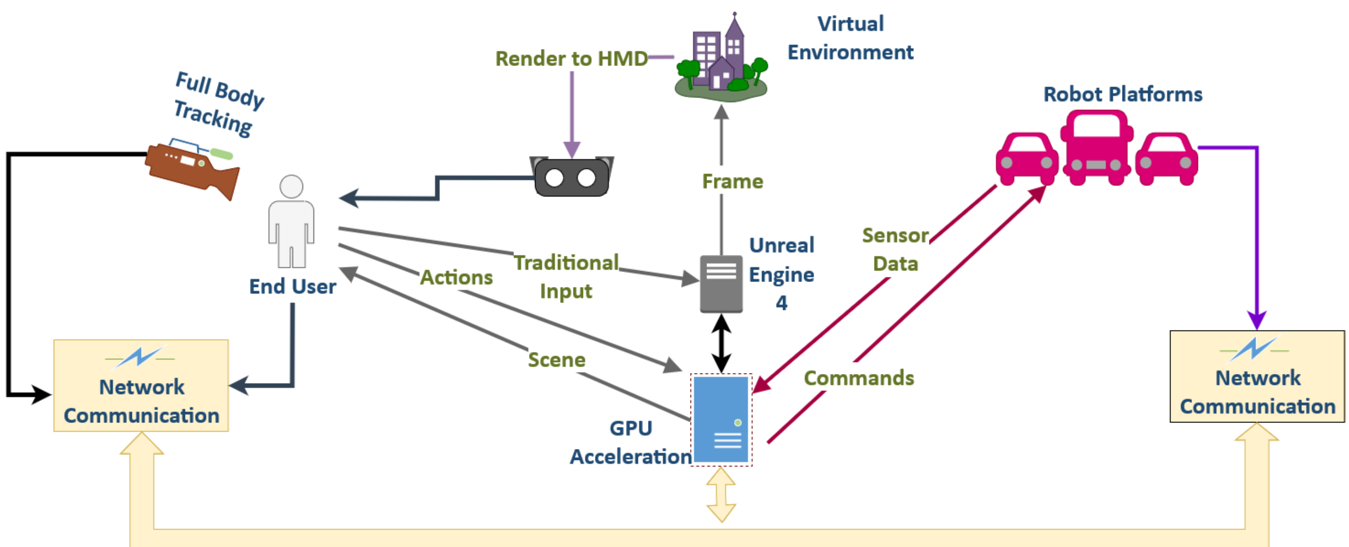

Figure 1: The overview of the proposed Virtual Reality Environment for Telerobotics.

ABSTRACT

There are currently important limitations in allowing for a more efficient man-machine collaboration in environments hostile to human presence. An immersive and intuitive user interface has the potential to bridge the gap between the human and the robot he/she is tasked to operate in the remote environment. In this work, we propose a novel architecture that allows for collecting large amount of sensory data to build models of the world and its inhabitants and present this information to the teleoperator of the robot. This environment maintain interfaces that are intuitive to the operator and accurately represents the robot's real-world state and environment. The proposed game client is capable of handling multiple users, much like a traditional multiplayer game, while visualizing multiple robotic agents operating within the real world. We also present a set of planned user studies to validate the performance of the proposed architecture compared to traditional tele-robotic applications.

CCS CONCEPTS

• Human-centered computing → Virtual reality; **• Computer systems organization → External interfaces for robotics**.

KEYWORDS

robotic, teleoperation, telerobotic, control, virtual, reality, VR, ros, unreal engine

VAMHRI '20, March 23, 2020, Cambridge, UK
2020. ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Tele-operation plays a vital role in space exploration, military reconnaissance, undersea operations, robotic surgery, training of personnel, and search and rescue operations in unsafe locations [4]. Despite the push for more autonomous robotic agents [17], tele-operation will still be necessary as a "default mode" for users, both to fix and to prevent errors caused by autonomous behavior as well as to boost the user's trust of the robot agent [3]. In addition, the task of robot tele-operation will become more commonplace as more and more viable robotic platforms become affordable and mainstream [13]. However, traditional robot control schemes are cumbersome due to information overload for the end-user in complex scenes [14]. In order to provide enough information to operators to effectively and efficiently control a robot remotely, the end user must have access to real-world visual odometry, positional and environment map data, as well as any laser range finder, sonar, or other sensor data used by the robot for obstacle detection and localization. In addition, the amount of data is further increased when multiple collaborative robots are considered. Traditional robot control schemes employ simple user interfaces to display information as a Heads-Up

Display (HUD) [3]. This traditional user interface is used to both offer information about the robot's current state as well as the state of its environment.

Games are traditionally thought of as a non-scientific medium of entertainment [15]. However, under the hood games are an immersive medium rendered through 3D virtual environments. In addition, these mediums are becoming more and more immersive through the advent of Virtual Reality (VR) Head-Mounted Displays (HMD's). Furthermore, these HMD's are becoming more and more commercially available [5]. VR will play an increasingly important role in tele-operation due to the synergy between robots and VR technology [1]. VR provides the user with increased immersion and enables the user to interact directly with their environment in an intuitive way. In addition, the robot provides a source of force feedback to any given tele-operation system. In theory, as an immersive and 3D medium capable of intuitive user interaction, a VR-enabled game is an ideal medium for the tele-operation of remote robotic platforms. However, there are many problems that must be addressed in order to utilize a game as a user interface for robot tele-operation. Below, we present the main contributions of this paper that addresses the outlining challenges of an effective immersive virtual environment useful for teleoperation of remote robotic agent.

2 CONTRIBUTIONS

Game Engine Integration: A range of new functionalities including, computer vision, robotics control frameworks, and parallel processes in support of the entire system are introduced and added to the base Unreal Engine 4 game engine in order to enable controlling of remote physical agents.

Heterogeneous Architecture: This architecture provides easy integration of robot clients and their unique robot interfaces into a server and an end-user dynamic and immersive virtual reality. The architecture allows for the offloading of computationally-intensive tasks, which facilitates communications between the end-user's UE4 game client and the real-world robot platform in order to place users virtually in the same environment as the robot to interact with both the robot and its environment. User input is then translated to real-world actuators on the robot platform.

2.1 Related Work

There has been much work in the past to improve end-user effectiveness for the purpose of remote robot tele-operation. The simplest and most traditional approach to robot tele-operation is through the direct manual control of the robotic platform through a direct video feed. Shiroma et al. investigate optimal camera position and orientations for this approach for the purpose of remotely-operated search and rescue robots [20] and found that a 3rd-person camera positioned such that the robot platform is in the center of its view is most effective, as it provides a clear view of the robot's immediate surroundings on all sides. To further improve this approach, Farkhatdinov et al. provided a study of speed, position, and command strategies [6]. In addition, this work includes text and haptic feedback to provide additional information to the end-user. These traditional approaches do not provide 3D environmental data, thereby causing to the overall efficiency of the operator.

To allow for improved operator spacial awareness, Nielsen et al. proposed combining a camera feed with a 2D map interpretation, as well as a 3D interpretation of the 2D map with the projection of the 2D camera feed into 3D space [14]. However, the environmental data in this system was limited to 2D data outside of the current camera's perspective. This approach would benefit from the decoupling of 3D data from the robots current location, as this would allow for the independent exploration of the virtual environment by the operator. One of the first uses of environments to visualize reachability by a robot manipulator was proposed by Tsumaki et al. [22]. This visualization is provided through a deformable mesh that visualizes the 6-Dimensional reachability data of the 6 Degrees of Freedom manipulator in a way that is intuitive to the user. However, this technique is limited in scale to that of a single manipulator, and does not include the robot's surrounding environment. In order to provide visual feedback with high levels of fidelity Mc Fadden et al. proposed the use of a machine learning system to decouple the object of interest and present the object acquired from a stereoscopic camera within the virtual environment [12].

In addition to the operator's spatial awareness of remote environment, a significant understanding of the tele-operated robotic agent's kinematics and the operator's own embodiment within the virtual environment are important factors. Wilson et al. proposed a full body motion calibration to seamlessly integrate large-scale hand and body kinematics data with fine-grain finger movements [26, 27]. This technique has the benefit of allowing low-cost motion trackers such as leap motion to be used for helping the remote operator use natural hand movements to perform complex tasks. Intuitive user interfaces for controlling remote robots investigated by Regenbrecht et al. [18, 19] have shown to improve the performance of teleoperation tasks [25].

Virtual reality interfaces offer intuitive means to directly map operator's actions to those of the robot they are controlling [23]. For example, the da Vinci Robot System is an immersive haptic telesurgery system which has improved surgical performance for both novice and experienced users [2]. Although powerful, the da Vinci robot and its interface is very task-specific to the surgical domain and stationary. Mallwitz et al. [10] developed a portable and easily-dressable exoskeleton that allowed a human user to naturally teleoperate a complex humanoid robot. This system is very intuitive to control, but again is limited to specific robots and is extremely expensive, heavily limiting the potential operator-base compared to web-based interfaces. Zhang et al. [28] used an HTC Vive to teleoperate a PR2 with a novel framework which allows the definition of different learning scenarios and instantiates these scenarios in a high quality game engine where a perceptual agent can act and learn in. Lipton et al. [9] also used a commercially available VR system for performing teleoperation on a Baxter. Whitney et al. [24] demonstrates the use of a ROS interface called ROS Reality to manipulate robots using VR. Gharaybeh et. al. [8] shows a promising application of teleoperation in exploring ocean floor for explosives.

One of the most significant impact of teleoperation is observed in the domain of space exploration. According to Berka et al. [7] NASA astronauts on the ISS perform a variety of tasks required for

ISS housekeeping and system maintenance through telerobotics. The remote monitoring and operation of many ISS systems by ground control has become an accepted practice for certain ISS tasks during the past decade. In terms of robots, these tasks are limited to coarse positioning maneuvers of external payloads/structures using manipulator arms. The use of Virtual Relaity would greatly enhance these workflows.

3 METHOD

We chose the following components for ease of integration, customization and testing.

3.1 Software

ROS is a popular, open-source development platform for robotic applications. It was chosen for the development of our platform for its modular, distributed design, active community, and wide range of relevant features and plugins.

3.2 Game Engine

Unreal Engine 4 is a popular game engine that is used widely for simulation, game and virtual reality development. It provides extensive customizability in the form of plugins. We used an unreal plugin called ROSIntegration [11] to communicate between ROS and Unreal.

3.3 Robot

Husky was considered as the Unmanned Ground Vehicle for this task as it supports ROS from its factory settings. It uses an open source serial protocol and offers API support for ROS. It was equipped with UR5, Universal's 6 degree of freedom manipulators. The UR5 has a reach radius of 850mm and payload of up to 5kg. For testing we mounted UR5 on Husky and tele-operated it through the real world and a simulated version through virtual world, simultaneously.

3.4 Visualization Hardware

The Vive Pro Eye is the VR headset used to build and test motion planning and virtual simulation. For motion planning, simulation and analysis we used two computers. A Windows machine with Intel Core i7-8700 CPU, 32 GB Memory and RTX 2080 GPU running UE4 with Vive. It then communicated with a linux machine with Intel Core i7-8750H CPU, 32 GB Memory and RTX 2070 running ROS. Good performance and Linux/ROS/Unreal compatibility prompted the use of these hardware.

4 THE PROPOSED IMPLEMENTATION

4.1 Challenges

One of the main problems with robotic operation is understanding how the robot would behave in complex and distant environments due to some motion. Motion planning is also impeded by the communication lag between the components that have to pass considerable information between long distances.

For improved navigation in any environment, it is imperative that the operator has reliable visual information. However, robots are sometimes operated in unexplored and dynamic sites. As these information cannot be collected beforehand, the robots send back sensor data to the remote operator. The operator needs to be able to quickly visualize these sensory data for effective performance. Virtual Reality (VR) simulation is a very effective way to simulate this complex data stream. Leveraging Unreal Engine 4 (UE4) for 3D visualization and simulation has two important advantages. The operator can move the robot around in this simulated environment and the physics engine (PhysX or Chaos) would produce realistic visual simulation regarding the motion. In addition, the rich and extensible components of UE4 can be used to simulate complex and dynamic environments for robust path planning.

This approach has some inherent limitations. First, for accurate simulations, the robot would need to send back sensory data constantly. Time lag would create dissociation between the data received and the current situation. Secondly, sensory noise or transmission error can lead to unpredictable outcome.

We attempt to resolve these issues in our approach detailed below.

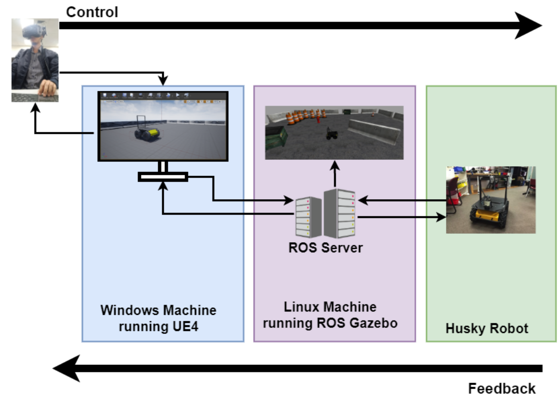

Figure 2: **Block Diagram of Teleoperation System**

4.2 Approach

Figure 2 shows our proposed teleoperation system. The scope of this is shown by the blue square. The robot sends its sensory data to the ROS server, which then converts them to simulation components compatible with unreal engine. The unreal engine then visualizes these additional scene components in the virtual environment. Wearing a VR headset, the operator can explore the immediate environment of the robot. The environment is dynamically updated as the robot moves, or as the environmental conditions change. Additionally, the robot also sends its own odometry to the server. The operator can see two versions of the husky at the same time. One (**Husky_UE**) is from Unreal's own simulation and the other version (**Husky_Real**) is generated from the received odometry information. Primarily the location of the Husky robot's base platform and it's wheel locations are used to infer the location of the rest of the **Husky_Real**. The motion planning for the robot involves:

- Wait for the immediate environment of the robot to be visualized.
- Place **Husky_UE** in the location and pose of **Husky_Real**

- Execute the motion plan on **Husky_UE** which triggers motion in **Husky_Real**
- After a fixed time interval, compare the position of the two counterparts.

The robot does not need to continuously communicate with the server, so time lag would play minimal role in motion execution. The various components of this implementation are described below.

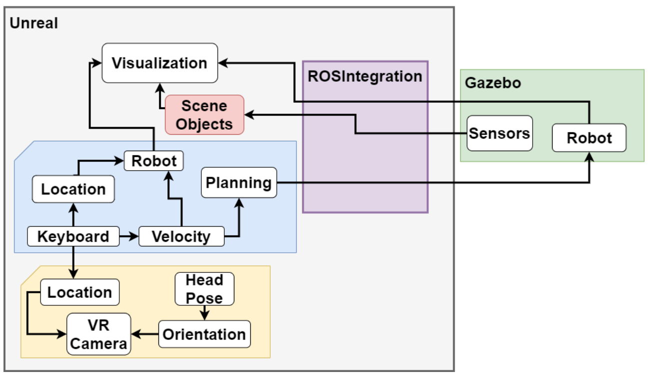

Figure 3: Overview of the simulation inside Unreal informed through ROS Gazebo. ROSIntegration plugin facilitates communication between the UE4 and ROS components.

4.2.1 Simulation and Visualization. In our approach the majority of motion planning computation is offloaded to the server. The server broadcasts the motion plan for a fixed time interval. After receiving the motion plan, the robot operates autonomously based on an established framework [16], until it receives new instructions. Meanwhile, the server calculates the discrepancy between the simulated position of the robot and the observed pose to send corrected control commands. The robot sends back its sensory information at fixed intervals or in cases where it diverges substantially from received navigation proposition.

In phase 1, we used UE4 to send motion commands to a Husky simulation. ROS packages called Gazebo provides the necessary interfaces to simulate a robot. Gazebo is a 3D rigid body simulator for robots. It integrates with ROS using ROS messages, services and dynamic reconfigure, very similar to physical robot communication.

Initially, only the robot is visible to the operator. As figure 3 shows, there is bidirectional communication between Unreal Engine and the robot simulation in gazebo, **Husky_Real**. **Husky_Real** publishes its sensory data such as point cloud and odometry as ROS messages. The ROSIntegration plugin is used to subscribe to these messages as needed. Point cloud data from depth sensors are used to generate the scene components and the odometry information is used to localize the robot in its environment. After receiving adequate information from **Husky_Real**, the scene object meshes are generated and visualized in VR. Teleoperation in VR offers the operator a crucial advantage by allowing camera repositioning to better explore the vicinity of the robot from any position and orientation. Turning the headset rotates the camera, while keyboard commands move the camera along each axis. The operator then creates a motion plan using the keyboard to move **Husky_UE** in this simulated environment. This triggers a translated motion plan for the **Husky_Real**.

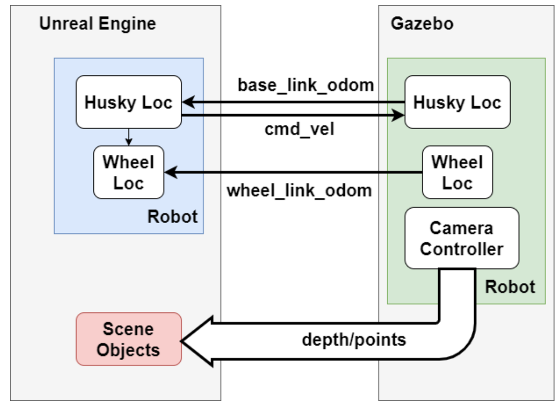

Figure 4: ROS messages advertised and subscribed to by the ROS clients

4.2.2 Control. Husky was designed with ROS in mind and integrates with the move base package with minimal configuration allowing the rover to accept simple navigation goals. Husky Simulator is a ROS package created by Clearpath Robotics that describes the physical characteristics and the geometry of the Husky robot in ".urdf" or ".urdf.xacro" files. ROS has mathematical formulations that use these kinematic and dynamic models to control the robot and simulate in Gazebo. ROS Gazebo is used to actuate the **Husky_Real** via the available control library. We limited the control of the **Husky_Real** to focus only on the movement of the robot base and wheels. A rostopic message **twist_marker_server\cmd_vel** is published through Unreal ROSIntegration plugin for desired amount of seconds to make **Husky_Real** move in any direction. The movement scale in Unreal and Gazebo are not the same. So, a transformation between the coordinate system in gazebo C_{gazebo} and unreal C_{ue}, **A** was employed to convert the topic message from and to Husky Gazebo. A sequence of markers can be set inside Unreal to plan the motion of **Husky_UE** and use:

$$C_{gazebo} = A_{n \times n} * C_{ue}$$

to find the *CGZ* to send the rostopic message to ROS Gazebo.

5 EXPERIMENTAL SETUP AND RESULTS

5.1 Experiment Setup

A replica of the ROS gazebo simulation was created inside the Unreal Engine visualization. **Husky_UE** was created importing the .uae files to .fbx that describe the 3D model of the robot. The other simulation components were imported similarly onto Unreal Engine. An Unreal Engine plugin called LiDAR Point Cloud would process the **depth/points** from the **Husky_Real** and generate mesh during runtime.

5.2 Results

The control and navigation methods in VR were designed to be intuitive. Intuitive in this context means minimal cognitive load is needed for the operator to apply these methods. Examining each

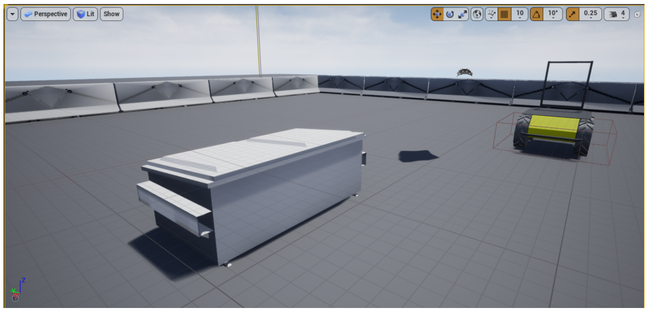

Figure 5: A snippet of Husky_UE

component shows that this is the case. Control methods are intuitive as well. Traditional keys for moving camera have been used to make the exploration intuitive and easy. Intuitive motion of the head corresponds to the motion of the camera. As the current set of motion commands are limited to wheel movements, the control methods are fairly intuitive. And the distributed architecture makes the process more efficient in terms of hardware resource and network communication.

6 PLANNED HUMAN PERFORMANCE AND USABILITY STUDY

To validate the benefits and usability of the proposed architecture, a human usability study is designed, in which human subjects are asked to perform two structured tasks remotely via a robotic agent, as described below.

The Navigation Task: In the first task, called *Navigation*, the subject is asked to navigate a robot through a maze. There are several physical and non-physical hazard zones populating the maze. The subject are given 20 minutes to complete the navigation. The time it takes to completely navigate the maze, as well as the number of physical and non-physical (hazard zone) hits are recorded for each subject.

The Exploration Task: In the second task, called *Exploration*, the subject is asked to explore the office corridors to find three symbols placed randomly within the environment. Again, the subject, is given 20 minutes to find all three symbols. The time to complete the task, as well as the number of symbols found by the subject are recorded.

6.1 Methods

6.1.1 Study Population. Institutional research ethics approval was obtained prior to the study's implementation. We plan to recruit 60 subjects in this round of the study. Participants are among students, staff, and faculty at the University of Nevada, Reno.

6.1.2 Experiment Setup. A prospective, randomized, repeated measures study design is planned which participants complete 2 sessions of the study: (1) Virtual Reality (VR) treatment session that utilized the proposed VR architecture to operate a remote robot, and (2) Regular Operated Vehicle (ROV) control session that utilized regular user interfaces and a monitor to control the robot. In order to counterbalance the study, participants were randomly assigned one of the two sessions first and asked to returned to participate in the other session after a period of 30 days has passed. The period between sessions is set to be more than 30 days to reduce the treatment effect on participants.

In each session participants were asked to complete the following task:

Navigation: The subject is asked to navigate a robot through a maze populated by several physical and non-physical hazard zones within 20 minutes. Dependent variables are the time it takes to completely navigate the maze (T_{nav}), the number of physical hits (h_{phys}), and the number of non-physical hazard zone hits (h_{haz}).

6.1.3 Instrument. We designed a 5-question instrument to validate the performance of the proposed VR-mediated teleoperation framework and to compare its performance with traditional user interfaces [21]. Questions **Q1**, **Q2**, and **Q5**, marked with *, were designed to validates the proposed system with respect to traditional interfaces. As such, we hypothesize that there will be no statistically significant difference in responses for these questions. Questions **Q3** and **Q4**, marked with **†**, were designed to show the benefit of the proposed interfaces compared to traditional teleoperation interfaces. We hypothesize that VR-mediated interface score significantly lower on **Q3** and significantly higher on **Q4**.

Q1*: The interfaces were intuitive for me to use.
Q2*: The interfaces were user friendly.
Q3†: I had trouble completing the tasks with the remote robot.
Q4†: The quality of the visual feedback was good.
Q5*: I believe the robot completed all of the tasks that I had delegated or commanded using available interfaces.

6.1.4 Statistics. We plan to evaluate the performance of the immersive virtual reality interfaces with traditional telerobotic interfaces using both inferential and descriptive statistics. For the inferential statistics we use a two-way analysis of variance (ANOVA). These analyses validate the reliability of the proposed immersive VR interfaces with the traditional teleoperation systems and their superior performance.

In order to validate the proposed VR-mediated teleoperation interface, we employ the 5-item instrument to compare traditional teleoperation interfaces with the proposed architecture. Q1, Q2, and Q5 aim to distinguish usability and reliability of the proposed interfaces compared with traditional interfaces. We evaluate the following three hypotheses:

H1: There is no statistically significant difference on Q1 answers for the ROV and VR conditions.
H2: There is no statistically significant difference on Q2 answers for the ROV and VR conditions.
H3: There is no statistically significant difference on Q5 answers for the ROV and VR conditions.

In order to demonstrate the effectiveness of the proposed VR-mediated teleoperation interface over traditional interfaces. We evaluate the following five hypotheses:

H4: There is a statistically significant difference on Q3 answers for the ROV and VR conditions.
H5: There is a statistically significant difference on Q4 answers for the ROV and VR conditions.

H6: There is a statistically significant difference in navigation time between the ROV and VR conditions.
H7: There is a statistically significant difference in the number of physical hits between the ROV and VR conditions.
H8: There is a statistically significant difference in the number of hazard hits between the ROV and VR conditions.

7 CONCLUSIONS AND FUTURE WORK

The architecture described in this paper removes some of the restrictions of remote teleoperation with an efficient framework that leverages intuitive VR interfaces. This architecture allows end users to communicate with multiple robots in real-world remote environments and handle a variety of robot sensor data and commands. Distributed computation with accelerated hardware and sparse communication between components enable tele-robotics at a wider and more reliable scale. This distributed system and game engine integration could enable a large number of participants to provide imitable demonstrations to robots for distributed learning or provide a platform for reinforcement learning systems within virtual environments. Feedback from the proposed user study would be helpful in optimizing it for these future applications.

REFERENCES

[1] GC Burdea. 1999. The Synergy Between Virtual Reality and Robotics, Invited Review. *IEEE Trans. on Robotics and Automation* 15, 3 (1999), 400–410.
[2] John C Byrn, Stefanie Schluender, Celia M Divino, John Conrad, Brooke Gurland, Edward Shlasko, and Amir Szold. 2007. Three-dimensional imaging improves surgical performance for both novice and experienced operators using the da Vinci Robot System. *The American Journal of Surgery* 193, 4 (2007), 519–522.
[3] Jessie YC Chen, Ellen C Haas, and Michael J Barnes. 2007. Human performance issues and user interface design for teleoperated robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37, 6 (2007), 1231–1245.
[4] Tung Dang, Frank Mascarich, Shehryar Khattak, Huan Nguyen, Nikhil Khedekar, Christos Papachristos, and Kostas Alexis. 2019. *Field-hardened Robotic Autonomy for Subterranean Exploration.* Technical Report.
[5] Rae A Earnshaw. 2014. *Virtual reality systems.* Academic press, NY.
[6] Ildar Farkhatdinov, Jee-Hwan Ryu, and Jury Poduraev. 2009. A user study of command strategies for mobile robot teleoperation. *Intelligent Service Robotics* 2, 2 (2009), 95–104.
[7] T. Fong, C. Provencher, M. Micire, M. Diftler, R. Berka, B. Bluethmann, and D. Mittman. 2012. The Human Exploration Telerobotics project: Objectives, approach, and testing. In *2012 IEEE Aerospace Conference.* 1–9.
[8] Z. Gharaybeh, H. Chizeck, and A. Stewart. 2019. Telerobotic Control in Virtual Reality. In *OCEANS 2019 MTS/IEEE SEATTLE.* 1–8. https://doi.org/10.23919/OCEANS40490.2019.8962616
[9] Jeffrey I Lipton, Aidan J Fay, and Daniela Rus. 2017. Baxter's homunculus: Virtual reality spaces for teleoperation in manufacturing. *IEEE Robotics and Automation Letters* 3, 1 (2017), 179–186.
[10] Martin Mallwitz, Niels Will, Johannes Teiwes, and Elsa Andrea Kirchner. 2015. The capio active upper body exoskeleton and its application for teleoperation. In *Proceedings of the 13th Symposium on Advanced Space Technologies in Robotics and Automation. ESA/Estec Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA-2015). ESA.*
[11] Patrick Mania and Michael Beetz. 2019. A Framework for Self-Training Perceptual Agents in Simulated Photorealistic Environments. In *International Conference on Robotics and Automation (ICRA).* IEEE, Montreal, Canada, 4396–4402.
[12] D McFadden, A Tavakkoli, J Regenbrecht, and B Wilson. 2017. Augmented Virtuality: A Real-time Process for Presenting Real-world Visual Sensory Information in an Immersive Virtual Environment for Planetary Exploration. In *AGU Fall Meeting Abstracts.* AGU, New Orleans, GA, 1–2.
[13] Russell Nickerson. 2009. Pioneering the personal robotics industry. In *IEEE International Conference on Technologies for Practical Robot Applications, 2009.* IEEE, IEEE, NY, 179–185.
[14] Curtis W Nielsen, Michael A Goodrich, and Robert W Ricks. 2007. Ecological interfaces for improving mobile robot teleoperation. *IEEE Transactions on Robotics* 23, 5 (2007), 927–941.
[15] Diana Oblinger. 2006. Simulations, games, and learning. *ELI White Paper* 1, 1 (2006).
[16] Christos Papachristos, Shehryar Khattak, and Kostas Alexis. 2017. Uncertainty-aware receding horizon exploration and mapping using aerial robots. In *2017 IEEE international conference on robotics and automation (ICRA).* IEEE, 4568–4575.
[17] Christos Papachristos, Shehryar Khattak, Frank Mascarich, Tung Dang, and Kostas Alexis. 2019. Autonomous Aerial Robotic Exploration of Subterranean Environments relying on Morphology–aware Path Planning. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS).* IEEE, 299–305.
[18] Jace Regenbrecht, Alireza Tavakkoli, and Donald Loffredo. 2016. An Intuitive Human Interface for Remote Operation of Robotic Agents in Immersive Virtual Reality Environments. In *2016 IEEE International Symposium on Human Robot Interactive Communication (RoMAN).* IEEE, New York, 1–2.
[19] Jace Regenbrecht, Alireza Tavakkoli, and Donald Loffredo. 2017. A robust and intuitive 3D interface for teleoperation of autonomous robotic agents through immersive virtual reality environments. In *2017 IEEE Symposium on 3D User Interfaces (3DUI).* IEEE, IEEE, Charleston, SC, 199–200.
[20] Naoji Shiroma, Noritaka Sato, Yu-huan Chiu, and Fumitoshi Matsuno. 2004. Study on effective camera images for mobile robot teleoperation. In *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on.* IEEE, USA, 107–112.
[21] Alireza Tavakkoli, Brandon Wilson, and Matthew Bounds. 2020. An Immersive Virtual Environment for Teleoperation of Remote Robotic Agents for Everyday Applications in Prohibitive Environments. In *in Proceedings of IEEE VR Workshop on Everyday VR.* IEEE, Atlanta, GA, 1–6.
[22] Yuichi Tsumaki, Mitsugu Takahashi, Woo-Keun Yoon, and Masaru Uchiyama. 2002. Virtual radar: An obstacle information display system for teleoperation. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, Vol. 2. IEEE, USA, 1185–1190.
[23] David Whitney, Eric Rosen, Elizabeth Phillips, George Konidaris, and Stefanie Tellex. 2020. Comparing robot grasping teleoperation across desktop and virtual reality with ROS reality. In *Robotics Research.* Springer, 335–350.
[24] David Whitney, Eric Rosen, Daniel Ullman, Elizabeth Phillips, and Stefanie Tellex. 2018. Ros reality: A virtual reality framework using consumer-grade hardware for ros-enabled robots. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE, 1–9.
[25] Brandon Wilson, Matthew Bounds, David McFadden, Jace Regenbrecht, Loveth Ohenhen, Alireza Tavakkoli, and Donald Loffredo. 2018. VETO: An Immersive Virtual Environment for Tele-Operation. *Robotics* 7, 2 (2018), 26.
[26] Brandon Wilson, Matthew Bounds, and Alireza Tavakkoli. 2016. A full-body motion calibration and retargeting for intuitive object manipulation in immersive virtual environments. In *2016 IEEE 9th Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS).* IEEE, Charleston, SC, 1–5.
[27] Brandon Wilson, Matthew Bounds, and Alireza Tavakkoli. 2016. Hand motion calibration and retargeting for intuitive object manipulation in immersive virtual environments. In *2016 IEEE Virtual Reality (VR).* IEEE, Charleston, SC, 313–314.
[28] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. 2018. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 1–8.