Telerobotic Control in Virtual Reality

Gharaybeh, Zaid
Electrical and Computer Engineering
University of Washington
Seattle, WA
zaidg@uw.edu

Chizeck, Howard
Electrical and Computer Engineering
University of Washington
Seattle, WA
chizeck@uw.edu

Stewart, Andrew
Electrical and Computer Engineering
University of Washington
Seattle, WA
andy@apl.uw.edu

***Abstract*—More than 400 underwater sites in and around the U.S. are potentially contaminated with hazardous undetonated munitions due to military testing activities [1]. The main remediation methods currently employed are 1. sending divers to manually retrieve the munitions and 2. blowing the munitions in place. Manual remediation is dangerous to the divers and blow-in-place strategies are environmentally damaging and harmful to nearby marine life. Teleoperation is an alternative remediation method that does not put people's lives at risk, and, if successfully carried out, is environmentally friendly. However, traditional teleoperation methods suffer from unintuitive and ineffective operator input control due to limitations such as poor depth perception by viewing the worksite from 2D displays and poor operator input control methods by using awkward input devices. Virtual reality interfaces immerse users in simulated environments and allow them to view and interact with simulated objects naturally. In this work, intuitive and effective input control methods based on Virtual Reality were designed and implemented as a ROS (Robot Operating System) package to teleoperate a robot arm for the remediation of underwater munitions using commercial off-the-shelf VR hardware. The control methods and ROS package are generalizable for other telerobotic applications.**

***Index Terms*—robotic, teleoperation, telerobotic, control, virtual, reality, VR, ros, rviz**

I. Introduction

A. Motivation

Due to past military training and weapons testing activities, undetonated and hazardous munitions were left in areas on the scale of millions of acres. Many remain in underwater environments such as ponds, rivers, and coastal regions. These munitions are an explosive hazard and an environmental hazard due to their toxic constituents leaking out to the water over time. Current methods for their remediation include sending divers to manually dismantle them or triggering them to explode using blow-in-place strategies as shown in figure 1. Blow-in-place strategies are extremely environmentally harmful as all the chemicals locked in the munition are instantly released into the water, and the shock waves released by the explosion harm nearby marine life. Manual remediation is dangerous to the divers as inadvertent detonation of the munitions could lead to dire results. A cleaner and safer method is needed.

Teleoperation is the control of a robot/machine from a distance. Telerobotic systems include robotic systems that are remotely operated to conduct critical tasks that are difficult to automate and that rely on expert knowledge. Utilizing an expert-driven telerobotic system to conduct the remediation mitigates the individual and environmental safety concerns. A telerobotic approach is much safer than manual remediation approaches because there is no more need for personnel to be present on site. It offers an environmentally cleaner approach than blow-in-place strategies because it does not entail the detonation of anything and no harmful chemicals are released into the water.

We thank SERDP (Strategic Environmental Research and Development Program) for supporting this research.

Fig. 1: Current Remediation Methods [1] [2]

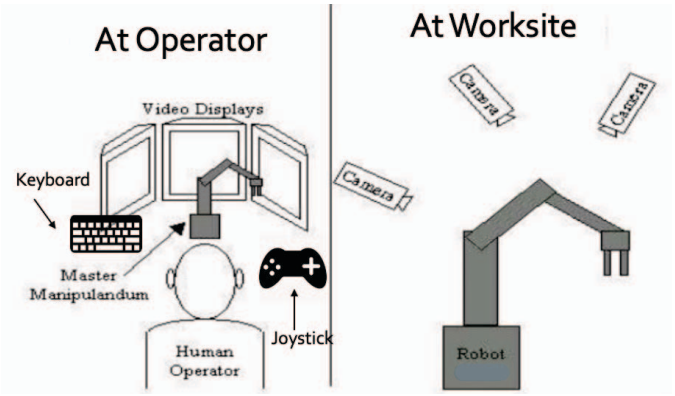


Fig. 2: Teleoperation using traditional methods [3]

However, teleoperation approaches that utilize 2D monitor(s) for the display of the remote worksite and input devices such as joysticks, keyboards, or manipulandums for the control of the remote robot (figure 2) suffer from poor operator controllability for several reasons. Firstly, using a 2D monitor decreases the operator's ability to perceive depth. Robust depth perception is crucial for precise and accurate control of objects in 3D space. Moreover, if the cameras used

978-0-578-57618-3 ©2019 MTS

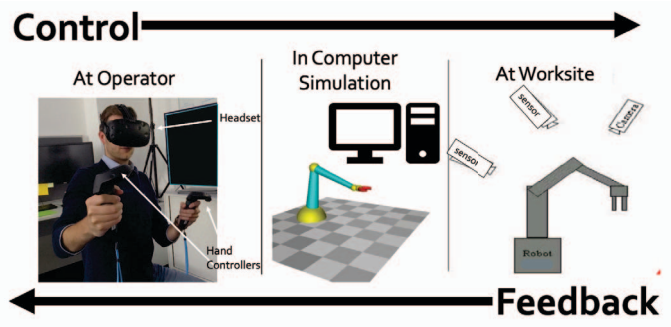


Fig. 3: Virtual Reality Teleoperation [4] [5]

at the worksite are stationary, only a limited set of static views of the worksite would be available to the operator. This forces the operator to have to continuously mentally rotate/scale/translate the robot he sees on the 2D monitor to conduct the desired movement using the input device. Lastly, keyboards, mice, and joysticks are ineffective and difficult to use to control complex robots in 3D because they don't leverage humans' natural ability to manipulate objects in 3D space. While the use of a master manipulandum such as the Phantom Omni offers haptic feedback and more intuitive control than keyboards/mice/joysticks, drawbacks of their use include the inability to control individual joints/only being able to control by moving the end-effector, and compatibility issues with robot arms with varying structures and degrees of freedom.

B. Teleoperation in Virtual Reality Advantages

When teleoperating in virtual reality, the operator wears a VR Headset that tracks the position and orientation of his head in real time and allows him to view a simulated environment of the worksite in 3D. He holds two hand controllers that are also equipped with real-time position and orientation tracking and provide input methods (buttons, triggers, and joysticks). He is immersed in the simulation and is able to interact with simulated objects naturally using the hand controllers, where visual representations of the hand controllers (their shapes, positions, and orientations relative to the headset) are viewable in the simulation.

Fig 3 shows how teleoperation in VR works. In our application, the operator views an immersive 3D simulation of the underwater environment (including the sea floor and the munition) by visualizing sensor data. He controls a simulated model of the worksite robot and conducts the remediation in the simulation. The real robot mirrors the simulated robot's movement by streaming the joint values of the simulated robot over the internet, retrieving the munition.

Teleoperation in VR resolves the issues discussed earlier that traditional teleoperation methods suffer from. Firstly, teleoperation in Virtual Reality brings back depth perception. Secondly, the use of a simulated 3D model of the worksite allows for free and unconstrained movement of the operator's frame of view. Lastly, the use of hand controllers with 3D position and orientation tracking enables effective and natural/intuitive interaction with simulated 3D objects. For these reasons, VR teleoperation delivers more effective and intuitive control than the 2D monitor + input methods discussed earlier.

C. Contributions

Control methods to teleoperate a robot arm in virtual reality are developed and implemented as a ROS package for the underwater remediation task using commercial off-the-shelf VR hardware. The methods and ROS package are generalizable for the control of any 6DoF robot arm for other applications, and modifiable for the control of other robotic systems.

D. Related Work

It has been previously shown that non-expert users favor and are more efficient with a VR teleoperation interface compared to a keyboard and monitor interface [6]. We believe that with a robust, intuitive, and effective control implementation, this extends to expert users. Previous works that used Virtual Reality for teleoperation include the "Homunculus" model [7] where a view from a 2D camera from Baxter's head was projected in virtual reality, creating a an experience where the operator feels as if he is the Baxter robot, with additional controls in the virtual reality space. Another work used a live point cloud from an RGB-D camera to populate the VR space and had an operator control a robot from the first person perspective to collect data to be used to train a deep imitation learning model [8]. Other works designed control methods to control remote vehicles and agents [9] [10].

II. Setup

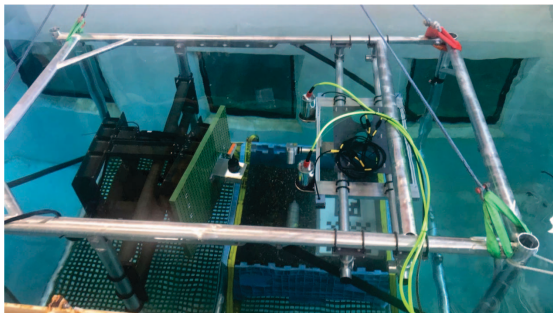
A. Remediation Hardware

The hardware considered for the remediation task resides in UW's Applied Physics Lab (APL). It includes the Robot Arm (Schilling Titan4 Manipulator), which is a 6DoF 6-Rotary robot arm, an underwater LiDAR sensor, and cameras (see figures 4b and 4a). The LiDAR used is not real time and needs several seconds of scanning to produce an image. Testing is done by attaching the robot arm to a frame within working range of a large open-top container filled with gravel holding a dummy munition, submerging the frame at UW's Ocean Sciences Building water tank/pool, and controlling the robot arm using joystick teleoperation to retrieve the dummy munition.

These devices interface with a machine holding commercial software from Olis Robotics. The Olis software directly controls the Schilling Robot Arm.

B. 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. The RVIZ (ROS visualization) environment is a great visualization/simulation tool for our purposes, and there exists a plugin,



(a) Hardware: Testing at UW Ocean Sciences Building's immersion tank - Schilling is detached in this pic

(b) Hardware: Schilling Titan4 Manipulator [11]

"oculus_rviz_plugin" that projects the RVIZ simulation to the Oculus DK2 VR headset. There also exists a plugin to integrate the Razer Hydra hand controllers into the ROS environment.

C. VR Hardware

The VR headset and hand controllers used to build and test the platform are the Oculus DK2 headset and Razer Hydra hand controllers. The reason for these choices boils down to good performance and Linux/ROS/RVIZ compatibility; these choices allowed for full ROS locality of the software. More advanced VR hardware can be integrated as it becomes ROS/Linux compatible or by implementing a high throughput low latency Linux-Windows pipe (on the same computer using a virtual machine or on different computers) and streaming the ROS simulated environment (in ROS Visualizer (RVIZ)) to Windows, where it is viewable by the VR headset, and streaming the positions/orientations of the headset and hand controllers to ROS. Tools such as ROS-Bridge and/or ROS-sharp may help with this setup.

III. Method

A. Approach

Figure 5 shows the block diagram of the virtual reality teleoperation system. The scope of this work is shown by the green square and is categorized by the four yellow blocks in the figure: 1. visualization of the worksite and all simulated objects, 2. free navigation of viewing frame in the simulated environment, 3. control of the simulated robot, and 4. communication between the ROS environment and the worksite.

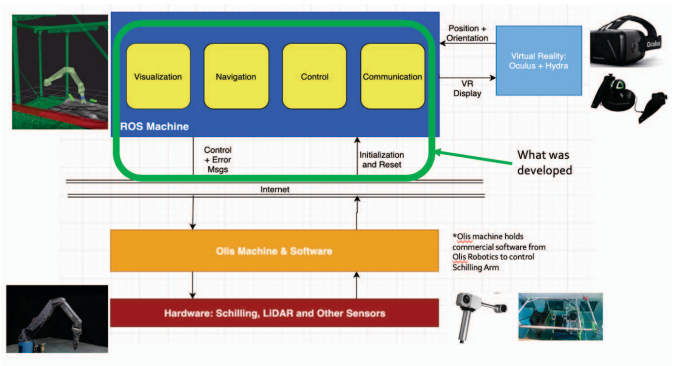


Fig. 5: Block Diagram of Teleoperation System

Additional functionalities were also designed and implemented for use in other telerobotic applications. These include using Simulink to model the dynamics of an XY cartesian robot, and drawing bounding boxes to serve as virtual fixtures for haptic feedback. Initially, we intended to attach the robot arm at its base to an XY Cartesian robot/gantry to serve the purpose of optimizing the position of the robot arm above the munition for operability, but eventually we opted for a more robust arm and abandoned the gantry. Moreover, in our setup, we didn't have haptic feedback devices. These functionalities will be discussed later.

B. Visualization

See figure 6 for all visualized components.

1) Robot Arm Model and Visualization: A model of the Schilling Titan4 Arm was constructed as a URDF file and loaded into RVIZ. The arm is a 6DoF 6-Rotary Robot arm shown in figure 7a

2) LiDAR Point Cloud Visualization: The LiDAR scanner at the worksite is used to capture the munition and the surrounding seafloor. The scan takes about a minute to complete. Once done, an .stl file is produced and sent to the ROS environment. Once the file arrives, it is visualized as a point cloud object in the RVIZ environment, as shown in figure 7b.

3) Razer Hydra and Marker Visualizations: The Razer Hydra Controllers were visualized in RVIZ as Marker objects. These visual representations mirror the actual Hydra controllers' positions and orientations in real time. Moreover, additional Markers were visualized to assist in control and navigation functionalities: an arrow is projected from each Hydra controller visualization. The left arrow is short and of fixed length, while the right arrow is of variable length and can be changed using the right joystick. At the tip of the right arrow, there exists a small translucent red sphere Marker. Its purpose is to give an enhanced sense of where the right arrow's tip is located at, since the longer the right arrow is, the smaller the red sphere appears from the operator's viewing frame. Further, since it is translucent, if the tip of the right arrow is behind any object, say behind the Robot arm, the operator wouldn't be able to see it, signifying that the arrow's tip is behind the object. If it is in front of the object, the operator

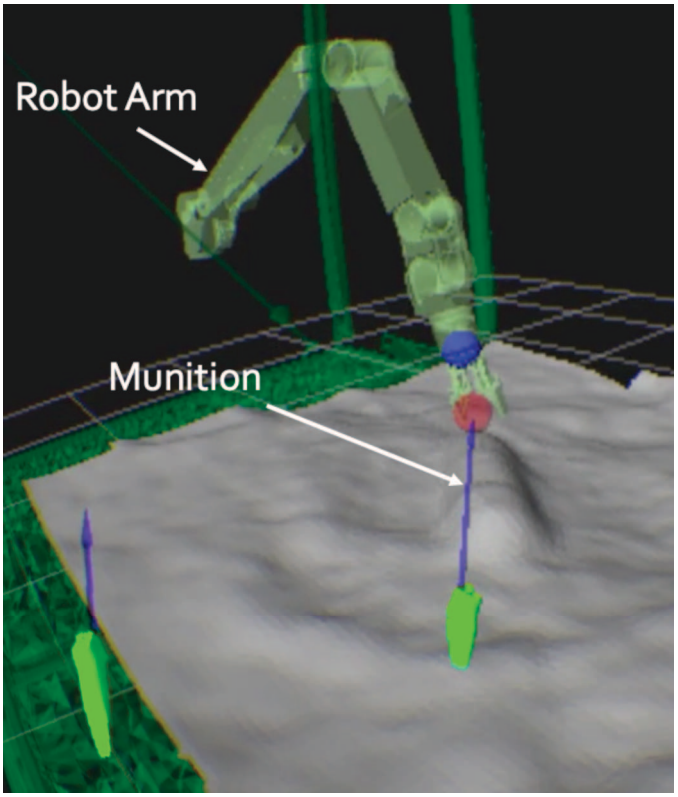


Fig. 6: Visualized components including Razer Hydra controllers, LiDAR point cloud, Schilling robot arm, and Marker visual aids: blue arrows projected from each controller, a translucent red sphere at the tip of the right arrow, and a blue sphere signifying closest joint being pointed at by right arrow.

would be able to see the sphere. If it is at the object, part of the sphere will be visible and part of it will not. Finally, a blue sphere is constantly displayed at the joint of the robot arm closest to where the operator is pointing with his right controller.

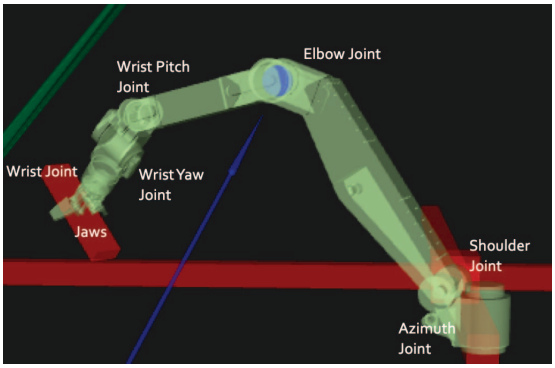
The additional RVIZ Marker visualizations are subtle and do not affect operator's the view of the simulated environment. They provide meaningful functionalities, as will be discussed.

C. Navigation

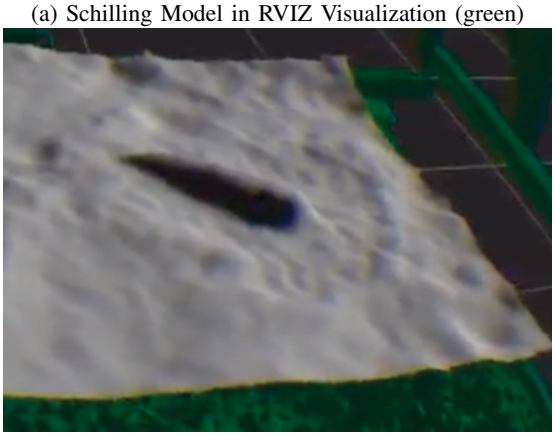
A crucial advantage of conducting the teleoperation in VR is to be able to move around in the simulation and view the worksite from any position and orientation. This was achieved by implementing five methods of navigating the simulation: a *rotation* method, three *translation* methods, and a *zoom* method.

A neutral forward gaze by the operator is defined to be looking straight along the x-axis of his viewing frame. Looking directly upwards is assumed to be looking straight along the z-axis of his viewing frame.

1) Rotate: Besides being able to rotate his view in the simulation naturally by turning his head around, the operator is also able to change the yaw of his default viewing frame (rotate left/right with a neutral gaze) by engaging the left bumper and



(a) Schilling Model in RVIZ Visualization (green)



(b) Point Cloud Visualized in RVIZ, Munition Shown

tilting his left wrist. The view continuously adjusts to match the direction his wrist is pointing towards.

2) Translate: To move freely in the simulation, three methods of translation were designed and implemented. In the first method, the operator engages the right bumper, and moves his right controller around in 3D space. The operator's viewing frame mirrors this movement in the opposite direction in real time. For example, if the user engages the bumper while the right controller is directly in front of him, and then he brings the right controller towards his body, he would move directly forwards in the simulation. This can best be described as "grabbing and holding on to a point in space and moving towards/away from it in any direction."

The second and third translation methods were designed based on the premise that the teleoperator will be viewing the same object (be it the munition or the robot arm) from multiple angles frequently.

The second translation method is based on rotating along the surface of an imaginary sphere whose center is the red sphere marker object in RVIZ that sits at the tip of the right Hydra's arrow marker. To do this, the operator points the right controller towards any point, modifies the length of the right Hydra's arrow, and then engages the right bumper; this turns the translucent red sphere at the tip of the right hydra's arrow opaque and fixes it in space. Then, the operator twists the right controller in any way. The position of the operator's viewing frame moves along the surface of an imaginary sphere such

that he is always pointing towards its center (which is the red sphere that signifies the original point at which he first engaged the right bumper). Because the right arrow's length is modifiable, the translation can be about any arbitrary sphere of any radius and any origin.

The third method of translation is a modified version of the second one but instead of the rotation happening about the imaginary sphere with the center being the right arrow's tip, it happens about the closest frame being pointed towards (regardless of the right arrow's length or pose). This allows the operator to freely rotate about the Robot's links without needing to modify the right arrow's length or point it exactly at the link.

The operator has the freedom to choose the translation method he favors by running the respective ROS node.

3) Zoom: The third navigation method implemented is zoom. The operator moves his head to look directly in a direction he would like to zoom in to/out from, engages both bumpers, and brings the controllers away from each other to zoom in or towards each other to zoom out. The viewing frame moves along the line his head is looking towards.

D. Control

Control functionalities were designed to function from any position and orientation of the operator's viewing frame. This allows the operator to move to his preferred position and orientation in the simulated environment from which to control the robot during the various stages of the remediation process.

1) Individual Joint Control: The operator may control (rotate) any individual joint simply by pointing towards it using the right controller (right hydra's arrow in VR), engaging the right Trigger, and twisting his wrist the same way he would when using a screwdriver (along the controller's x-axis). In RVIZ, a blue sphere Marker object is continuously displayed on the joint currently being pointed at by the right controller to signify the joint he is pointing towards and to prevent the operator from initiating any unintended joint movements. Once he engages the right Trigger, the blue sphere is fixed on the joint he was pointing at when he first engaged the Trigger, and only that joint becomes controllable. This prevents him from moving any other joint while controlling that individual joint.

Triggers are unlike buttons/bumpers. Buttons/bumpers can only take on the boolean values 0,1. Triggers' can take on any value from the range [0,1], depending on its level of engagement (how hard the user is pressing on it). The rotation is scaled based on the level of engagement of the Trigger i.e. if the Trigger is pressed lightly, with a full rotation of the controller, the joint rotates only slightly. With full engagement of the Trigger, the joint's rotation mirrors the roll of the operator's wrist. This allows fine-tuned joint control.

2) Direct Inverse Kinematics: Inverse kinematics of the Schilling arm's end effector were solved in Matlab using the Levenberg–Marquardt algorithm. With motion along continuous paths, the algorithm is able to compute IK solutions at a rate of approximately 19Hz for the Schilling Titan4 robot arm model on the development computer. Since motion of the robot arm follows continuous paths, the initial guess given to the optimization algorithm is the current pose of the robot arm. For this application, 19Hz is practically enough to conduct the remediation process in real time.

To perform inverse kinematic control of the end effector, the operator engages a button on the right controller and moves the controller in any (X,Y,Z) motion from any viewpoint in RVIZ. The end effector mirrors the same (X,Y,Z) movement as that of the controller at 19Hz by streaming in real time the next point along the path to the Matlab node where the IK computation is solved and the results published back to update the pose of the simulated robot.

3) Path Inverse Kinematics: Besides controlling the end effector directly, a path can be drawn for the end effector to follow, with its current position as the starting point. The operator engages the draw button and moves the right hydra freely, using the right arrow as a "pen" to draw the path in 3D space. The implementation allows the operator to draw straight line segments, "free drawing" of any custom path, or any combination of the two. Modifying the length of the right arrow changes the length of the "pen", allowing the operator to draw with great precision and accuracy.

To draw a custom path, the operator engages the draw button and moves the right controller. This continuously adds new points to the path at the tip of the right hydra's arrow. New points are added to the path only when the tip of the right hydra's arrow is above a certain distance threshold from the last added point. This ensures that while the operator is not moving the right controller (pen), no additional points are needlessly added to the path at the same position.

To draw straight line segments, the operator disengages the draw button, points at a different point in space using the "pen", and engages it again. This fills up the straight line segment from the last point in the path to the point at which the operator just engaged the draw button with new points. A single path may consist of an unlimited number of straight line segments and custom drawn curves. The operator can also clear the path by hitting the designated button if he wishes to draw another.

After drawing a path, the operator can initiate the motion of the robot arm's end effector to move along the path and is able to pause and resume the motion at any point.

4) Undo Motion: A simple undo motion functionality was implemented. The operator hits the designated button and the robot arm smoothly undos whatever motion it just followed.

E. Communication

A method of communication between the worksite computing interface and the remote simulation holding ROS computer was designed and implemented. The interface supports streaming the joint states of the simulated robot to the worksite robot for its control. It also supports sending set/reset messages from the worksite computer to the remote ROS computer to initialize/reset the state of the simulated robot to match that of the worksite's. To get the stream of joint configuration information, the worksite computer SSH's into the ROS computer

and accesses the stream by subscribing to the schilling_state topic. It then parses the output and redirects it to control the robot arm.

1) Safety Mechanism: It would be reckless to stream the raw joint states of the simulated robot to control in real time the robot at the worksite, especially for safety critical tasks such as underwater munition remediation. This is because if the simulated robot moves abruptly at any time for any reason it would risk dangerous collisions at the worksite.

To counter this, a safety mechanism is implemented such that the ROS machine will stop streaming the joint values of the simulated robot arm to the real robot if at any time any of the joints moves too fast (or jumps) in the simulation. Instead, an error message, "ERROR: joint state maximum step value exceeded" will be displayed in RVIZ, and the stream stops.

Jumps in the schilling's joints' values occur in rare situations when the only solution of an inverse kinematics computation entails a radically different joint configuration from the present one. This may happen when a joint limit is met while performing inverse kinematic control of the end effector or when the end effector traverses a path that exits its workspace and enters back into it at a different point. The safety mechanism can also be triggered if the operator moves the robot too fast.

To reset this error and continue the stream, the simulated joint configuration has to be reset to match the worksite robot's joint configuration (the configuration just before the max jump error occurred, assumingely).

2) Initialization and Reset of Simulated Robot Joint Configuration: A method was implemented to set the state of the simulated robot to match that of the real robot (such as when first initializing the simulation or when resetting the simulated robot's state after a max jump error). To initiate a set/reset command, the worksite computer (assuming its connected to the ROS computer) runs a ROS topic publish command to the schilling_set topic with the current configuration of the robot arm.

F. Additional Functionalities Implemented

1) Bounding Boxes: A method for drawing an unlimited number of rectangular cuboid bounding boxes was implemented. The bounding boxes are intended to behave as virtual fixtures. Haptic feedback was not implemented in this work so the bounding boxes only exist as visual Marker objects in RVIZ (see figure 8) and their coordinates/shape in the ROS environment. The operator uses the tip of the right arrow to place the first vertex of the bounding box, then hits the bounding box draw button again at another location to fix the opposite vertex in place. He is free to navigate after placing the first vertex and a continuous visual representation of the bounding box will be rendered until he fixes the opposite vertex. The operator can also remove the last drawn bounding box.

2) Gantry and Underwater Dynamics Modelling: An (X,Y) cartesian robot (gantry) was designed in simulation to bring the Schilling arm to the optimal (X,Y) position above the munition before conducting the remediation. We did not have the hardware for this system, however, so the full system solution was not implemented.

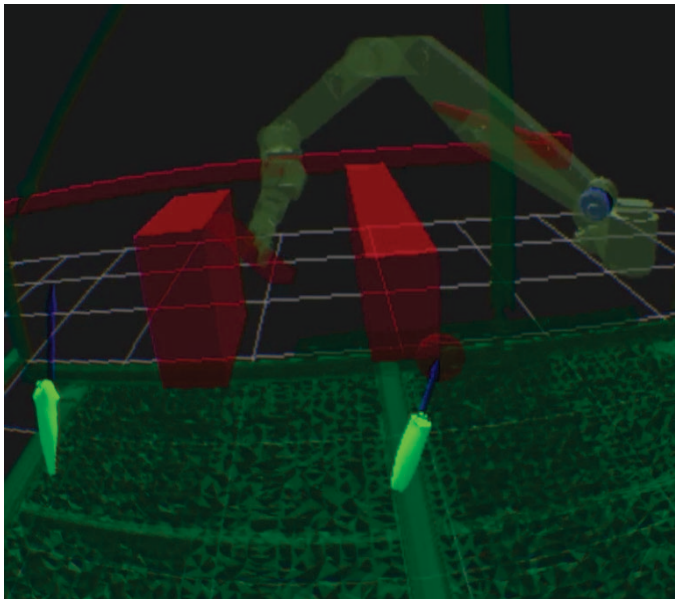


Fig. 8: Bounding Box Visualization in RVIZ

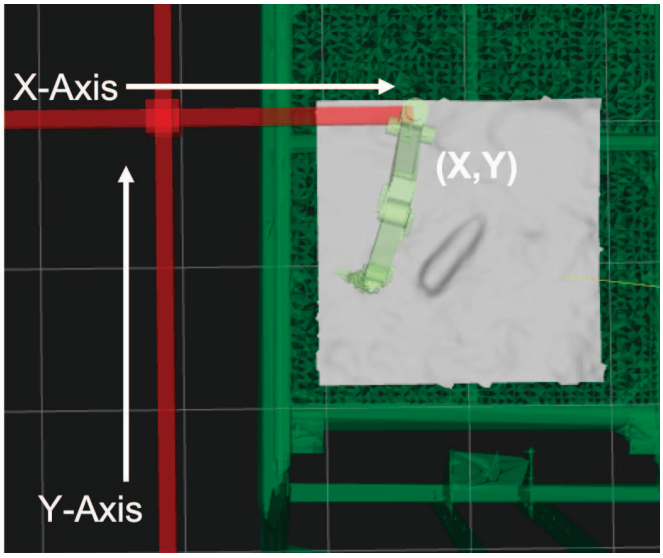


Fig. 9: Top view: Gantry for bringing Schilling arm to optimal position above the munition from which the retrieval is conducted.

We assume that the real gantry is composed of two force controlled, orthogonal, decoupled axes. A URDF robot model of the gantry was created and implemented in ROS and visualized in RVIZ. In ROS, the gantry was modeled as two position controlled, orthogonal, decoupled rods (axes) with the Schilling arm's base attached at the gantry's end effector, as shown in figure 9.

To allow for (X,Y) position control of the gantry in ROS, the underwater system dynamics were modeled in Simulink and a

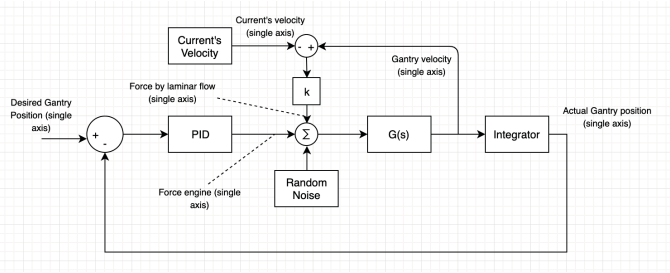


Fig. 10: Closed Loop Control System for Single Axis of Gantry

closed loop control system for each (force controlled) axis was implemented in Simulink. A ROS - Simulink pipe was used to send move-to-position requests from ROS to simulink and stream back the Simulink model's output to move the gantry in RVIZ constrained by the system's dynamics. The pipe was a feature of Matlab's Robotics Toolbox submodule.

To model the dynamics in Simulink, we assume that three main forces act on the gantry in the underwater setting: 1. the gantry's engine force that drives each of its axes, 2. a damping force on each axis proportional to the difference between the axis' velocity and the underwater currents velocity (in the respective axis direction) due to the laminar flow of the liquid, and 3. Random noise as a result of random movements of nearby marine animals, vibrations, and/or robot movements.

For each axis:
$F = ma$
$F_{engine} + k(V_{gantry} - V_{current}) + F_{noise} = ma$
$F_{engine} + F_{laminar} + F_{noise} = ma$
$F_{total}(t) = m\dot{v}(t)$
In the Laplace domain:
$F_{total}(s) = msV(s)$
$\frac{V(s)}{F(s)} = G(s) = \frac{1}{ms}$

A discretized PID controller was used to control the engine output and its parameters were tuned in Simulink for low overshoot, quick response, and zero SSE (assuming no noise). The closed loop control system for a single axis is shown in figure 10. Preliminary control methods in ROS were implemented for the gantry's control using the Razer Hydra's joystick.

Please note that this Simulink model does not take into account the effect of the shape of the Schilling arm on the laminar flow force in this implementation.

IV. Results

The control system was described as "efficient", "natural" and "intuitive" by inexperienced participants, and we believe this will extend to experienced users.

A. Ease of Use

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 and that they make as much use of his innate relevant mental faculties for manipulating objects in 3D space. Examining each method shows that this is the case. Rotating is simple and natural since the operator merely changes his orientation by pointing the controller towards the desired direction. Zooming uses the same method people are accustomed to when using their smartphones (hold screen and pinch/un-pinch, then release) but using their hands instead of their fingers. A 3D VR environment that allows for unconstrained motion of the viewing frame could be described as an environment where the user feels weightless, such as an outer space shuttle. In the first method of translation, the operator "holds on to a point in space, and brings his body towards/away from it". Besides using a couple spray cans, this could be the most intuitive way of moving as a weightless agent in such an environment. In the other translation methods, the user "rotates about a point in 3D space". This is less intuitive, but still requires minimal cognitive load especially in cases where the operator wants to frequently view/control the Schilling arm from multiple orientations. In these cases, the second and third methods require less cognitive load than the first, as using the first method necessitates rotating after translating to the point where the user wants to view/control the robot. If the operator is using controllers with enough bindings (or if he customizes his own bindings), all three methods of translation could be made available simultaneously to him at any time.

Control methods are intuitive as well. Individual joint control mimics the use of a screwdriver and gives freedom of control of any non-fixed joint naturally. The mirroring method of inverse kinematic control is the simplest way of controlling the end effector naturally from any position and orientation in the simulation. Drawing paths and bounding boxes emulates the use of a pen. And due to the flexibility of being able to pause the robot arm mid-path, draw straight lines, and easily erase boxes/paths with one click of a button, these methods require minimal cognitive load and are easy to use.

B. Performance

Performance in this context means 1. Low execution time, 2. high efficiency, 3. high effectiveness, and 4. high safety. All navigation and control methodologies implemented are a click of a button away and, once initiated, their effects are displayed to the operator instantly as real time visual feedback. Their execution time is also low. Navigation and control methods are simple, robust, and differentiated to ensure efficient and effective navigation in the simulated environment and control of the robot arm in various circumstances while giving the operator the freedom to choose his preferred methodologies. Solving inverse kinematics using the Levenberg–Marquardt algorithm allowed for smooth, responsive, and almost real-time (19Hz) Schilling end effector control (this may be higher for an optimized high performance machine). Lastly, the safety mechanism ensures high safety of the system, as no abrupt unintentional movements of the simulated robot may affect the real robot's state at any time.

C. Generalizability

The navigation methods are generalizable for any teleoperation in VR application, since they are robot agnostic. The control methods are suitable for the control of any robot arm, but may not be suited for the control of a remote agent/vehicle, or other different robotic systems.

V. Conclusions

Telerobotic operator input control methods implemented in VR are more intuitive and effective than operator input control methods based on traditional teleoperation approaches. Effective and intuitive telerobotic control methods were designed and implemented as a ROS package for the teleoperation of a robot arm for munition remediation using commercial off-the-shelf VR hardware.

A. Future Work

1) Gantry Control Methods in Virtual Reality: Methods for controlling the gantry in VR have been explored but not implemented. Direct control of the gantry can be implemented similar to direct control of the Schilling's end effector - by dragging and dropping the gantry's (X,Y) position from anywhere in the simulation (ignoring the z-displacement of the controller), or by pointing the right arrow to the gantry's plane of motion and "drawing" the motion in real time. Drawing paths for the Gantry to follow can be implemented similar to drawing paths for the Schilling, except that instead of drawing in 3D space freely, the user draws on the gantry's (X,Y) plane within its workspace as if he is drawing on a piece of paper, using the right Hydra's arrow as a pen, from above or below the gantry.

2) Automation: Automating some or all aspects of the remediation process is a big next step for this project.

References

[1] serdp estcp, "Munitions in the underwater environment," https://www.serdp-estcp.org/Featured-Initiatives/Munitions-Response-Initiatives/Munitions-in-the-Underwater-Environment, 2019 (accessed April 7, 2019).

[2] A. Stewart, "Augmennted co-robotics for remediation of military munitions underwater, dr. andrew stewart university of washington applied physics laboratory brief to the scientific advisory board," 2019 (accessed April 7, 2019).

[3] M. Memeo, "Teleoperation," https://www.slideshare.net/MariacarlaMemeo/teleoperation/5, 2019 (accessed June 10, 2019).

[4] "Teleoperation visual," https://upload.wikimedia.org/wikipedia/commons/e/ee/Reality_check_ESA384313.jpg", 2019 (accessed June 10, 2019).

[5] http://www.math.union.edu/~dpvc/talks/2000-11-23.MTCM/robot.html.

[6] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex, "Comparing robot grasping teleoperation across desktop and virtual reality with ros reality," 2017.

[7] J. Lipton, A. Fay, and D. Rus, "Baxter's homunculus: Virtual reality spaces for teleoperation in manufacturing," 2017.

[8] T. Zhang, Z. McCarthy, O. Jow1, D. Lee1, X. Chen, K. Goldberg1, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," 2018.

[9] B. Wilson, M. Bounds, D. McFadden, J. Regenbrecht, L. Ohenhen, A. Tavakkoli, and D. Loffredo, "Veto: An immersive virtual environment for tele-operation," *MDPI*, 2018.

[10] J. Regenbrecht, A. Tavakkoli, and D. Loffredo, "A robust and intuitive 3d interface for teleoperation of autonomous robotic agents through immersive virtual reality environments," *IEEE Symposium on 3D User Interfaces (3DUI)*, 2017.

[11] Razer, "Razer hydra guide," https://dl.razerzone.com/master-guides/Hydra/HydraOMG-ENG.pdf, 2019 (accessed April 7, 2019).

[12] SERDP, "Munitions underwater," www.serdp-estcp.org/Program-Areas/Munitions-Response/Munitions-Underwater, 2019 (accessed April 7, 2019).

[13] ROS, "Ros home page," http://www.ros.org/, 2019 (accessed April 7, 2019).

[14] Mathworks, "Simulink," https://www.mathworks.com/products/simulink.html, 2019 (accessed April 7, 2019).

[15] O. Robotics, "Oculus rviz plugins," https://github.com/OSUrobotics/oculus_rviz_plugins, 2019 (accessed April 7, 2019).

[16] Mathworks, "Ros support," https://www.mathworks.com/hardware-support/robot-operating-system.html, 2019 (accessed April 7, 2019).

[17] Roadtoovr, "Reactive grip brings tactile feedback to the razer hydra, other motion input devices [video]," https://www.roadtovr.com/tactical-haptics-reactive-grip-razer-hydra-gdc-2013/, 2019 (accessed April 7, 2019).

[18] E. Group, "subsea manipulator arms," https://www.ecagroup.com/en/solutions/subsea-electrical-manipulator-arms, 2019 (accessed April 7, 2019).

[19] ROS, "Rviz," https://github.com/ros-visualization/rviz, 2019 (accessed April 7, 2019).

[20] ——, "tf," http://wiki.ros.org/tf, 2019 (accessed April 7, 2019).

[21] Aleeper, "razer_hydra ros node," https://github.com/aleeper/razer_hydra, 2019 (accessed April 7, 2019).

[22] J.-L. Blanco, "A tutorial on se (3) transformation parameterizations and on-manifold optimization," 2019 (accessed April 7, 2019).

[23] ROS, "Displaytypes/marker," http://wiki.ros.org/rviz/DisplayTypes/Marker, 2019 (accessed April 7, 2019).

[24] Razer, "Razer hydra portal 2 bundle," https://www2.razer.com/au-en/gaming-controllers/razer-hydra-portal-2-bundle, 2019 (accessed April 7, 2019).

[25] D. R. cursos, "Titan 4 manipulator," https://www.youtube.com/watch?v=ZaQLyIvC5aw, 2019 (accessed April 7, 2019).

[26] OSUrobotics, "ros_ovr_sdk," https://github.com/OSUrobotics/ros_ovr_sdk, 2019 (accessed April 7, 2019).

[27] J. Vertut, *Teleoperation and Robotics: Applications and Technology. Vol. 3*. Reading, MA: Springer Science & Business Media, 2013.