This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

A Shared Autonomy System for Precise and Efficient Remote Underwater Manipulation

Amy Phung, Gideon Billings, Andrea F. Daniele, Matthew R. Walter, Richard Camilli

***Abstract*—Conventional underwater intervention operations using robotic vehicles require expert teleoperators and limit interaction with remote scientists. We present the SHared Autonomy for Remote Collaboration (SHARC) framework that enables novice operators to cooperatively conduct underwater sampling and manipulation tasks. With SHARC, operators can plan and complete manipulation tasks using natural language or hand gestures through a virtual reality (SHARC-VR) interface. The interface provides remote operators with a contextual 3D scene understanding that is updated according to bandwidth availability. Evaluation of the SHARC framework through controlled lab experiments demonstrates that SHARC-VR enables novice operators to complete manipulation tasks in framerate-limited conditions (i.e., 0.1–0.5 frames per second) faster than expert pilots using a conventional topside controller. For both novice and expert users, the SHARC-VR interface also increases the task completion rate and improves sampling precision. The SHARC framework is readily extensible to other hardware architectures, including terrestrial and space systems.**

***Index Terms*—Shared Autonomy, Virtual Reality, Underwater Manipulation**

I. Introduction

ROBOTIC systems for scientific exploration and intervention in the deep ocean (beyond SCUBA diving depth) are vital for improved understanding of natural environments and effective management of regions altered by human activity [1]. However, inaccessibility remains a fundamental impediment to these deep ocean operations [1, 2]. Technological innovations in marine robotics provide a path forward to substantively improve the efficiency and geospatial precision of deep submergence operations [2], while also increasing societal engagement and understanding of oceanographic processes [1].

Currently, dexterous sampling tasks at depth are performed by tethered underwater remotely operated vehicles (ROVs) equipped with robotic manipulator arms. ROV pilots directly teleoperate these manipulators with a topside controller in a shipborne control room using a high-bandwidth, low-latency tether that supports the realtime streaming and display of numerous video and telemetry data. However, teleoperation has several limitations that sacrifice the effectiveness and efficiency of the tasks being performed. First, the tether significantly limits the ROV's maneuverability and increases the infrastructure requirements for operations [3]. Second, it places significant cognitive load on the operator, who must reason over both the high-level scientific objectives and low-level manipulator control, while simultaneously interpreting the diverse data streamed from the ROV. Third, operators typically exercise one joint angle at a time in a "joint-by-joint" teleoperation mode when using conventional control interfaces [4, 5], which restricts dexterity, limits efficiency, and can be error-prone. These problems are exacerbated when operating under bandwidth-limited, high-latency conditions [6]. Despite these limitations, direct teleoperation is still the standard approach for robotic arm manipulation tasks such as benthic sample collection and return with ROVs [2].

Unfortunately, access to ROVs for sampling remains prohibitively expensive for many researchers since their operation requires a surface support vessel (SSV) with a highly trained operations crew. At the same time, SSV space constraints limit the number of onboard participants. Reduced berthing for scientific crew is a particular concern for the U.S. academic research fleet [7]. Expanding shore-based access for remote scientists and technicians to observe and control robotic sampling processes can decrease the berthing requirements for technical personnel and increase the number of scientists engaged in the deployment, both onboard and remote, while reducing barriers to participation (e.g., physical ability, experience, or geographic location).

Recent advancements in marine intervention systems are driven by a desire to extend conventional ship-based teleoperation of ROVs beyond the constraints of high-bandwidth tethers. Offshore service companies are developing ships equipped to deploy ROVs controlled from shore-based centers via satellite communication links, while recent advancements have demonstrated hybrid acoustic and optical communication systems for tetherless ROV control [8, 9]. These technologies

This work was supported by the National Science Foundation National Robotics Initiative under Grants IIS-1830500 and IIS-1830660, and in part by the National Aeronautics and Space Administration, Planetary Science and Technology from Analog Research under Grant NNX16AL08G, and the Strategic Environmental Research and Development Program Grant W912HQ24P0024. Amy Phung would like to acknowledge support from the National Science Foundation Graduate Research Fellowship under Grant No. 2141064 and the Link Foundation.

All research involving human subjects was conducted in accordance with research methods reviewed and approved by the Woods Hole Oceanographic Institution's Office of the Deputy Director and Vice President for Science and Engineering. Each human test subject participant reviewed the study methods and granted consent prior to participation in the study.

(Corresponding author: Richard Camilli)

Amy Phung is with the Woods Hole Oceanographic Institution, Woods Hole, MA 02543, and also with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: aphung@mit.edu)

Gideon Billings is with the Australian Centre for Robotics, The University of Sydney, Sydney NSW 2006, Australia, and also with the Woods Hole Oceanographic Institution, Woods Hole, MA 02543 USA (e-mail: gideon.billings@sydney.edu.au)

Andrea F. Daniele and Matthew R. Walter are with the Toyota Technological Institute at Chicago, Chicago, IL 60637 USA (e-mail: afdaniele@ttic.edu; mwalter@ttic.edu)

Richard Camilli is with the Department of Applied Ocean Physics and Engineering, Woods Hole Oceanographic Institution, Woods Hole, MA 02543 USA (e-mail: rcamilli@whoi.edu)

aim to significantly reduce the ship-based infrastructure necessary for tether management, which in turn would enable deployment from smaller vessels, reduce operational costs, and streamline vehicle deployment and recovery. Furthermore, in marine robotics, the nascent shift from single-vehicle operations to multi-vehicle cooperative missions will require shore-based operation centers and tetherless communication modalities [10].

Shore-based operation of tetherless vehicles requires a combination of satellite, acoustic, and optical-based communications, which can adversely impact operational robustness and efficiency by introducing latency and reducing bandwidth. For direct low-level teleoperation, operators typically require less than half a second of latency (0.4 seconds) [11], and their performance declines significantly with greater latency (particularly beyond 2.6 seconds) [12, 13]. Conventional direct teleoperation becomes infeasible under bandwidth limitations or high latency [14], necessitating a different control method. At extremely low bandwidths, the effects of latency are diminished relative to the impact of bandwidth. For instance, if a given bandwidth supports a maximum data update rate of 0.1 Hz, data can be delayed by up to 10 seconds, which far outweighs the effect from latency due to satellite (e.g., $<$99 ms with Starlink [15]), through-water optical (µs-ms), or acoustic communications (e.g., 8 seconds round-trip acoustic travel time at 6000 m water depth). Consequently, developing systems capable of operating across extremely low-bandwidth connections will be crucial to unlocking the full potential of future shore-based operations and tetherless multi-vehicle collaboration.

Greater autonomy for marine intervention systems is a viable solution to the challenges of limited or degraded communication associated with tetherless and shore-based operations. Although methods for fully autonomous underwater intervention are advancing, contextual awareness in unstructured environments remains insufficient for such systems to operate reliably [16]. Supervisory control-based approaches, which integrate low-level robot autonomy with human input for high-level decision-making, are widely used to address challenges with latency and bandwidth in a variety of applications ranging from telerobotic planetary exploration [6, 17] to telesurgery [18, 19, 20]. For underwater infrastructure maintenance applications, the DexROV project implemented a supervisory control system that can autonomously avoid joint limits and obstacles in the workspace, and enables semi-autonomous control by remote users using an exoskeleton [21, 22]. The DexROV's shore-based interface incorporates prior models of the workspace with a 3D scene reconstruction generated by onboard processing of stereo camera imagery, which are also transmitted over the satellite link [23, 24, 25, 26]. Recent work has also explored learning-based approaches to infer human intent in order to increase an autonomous system's robustness to bandwidth limitations during remote teleoperation [27, 28, 29, 30]

While a system's ability to maintain safe operations in low-bandwidth environments is paramount, its usability among a broad user base with minimal training is also important. ROV pilots typically undergo extensive training to manage the high cognitive load imposed by conventional teleoperation interfaces. Pilot training has important ramifications for ROV operating costs and staffing requirements. Ongoing research aims to reduce the cognitive load placed on operators, which could improve the accessibility of operator interfaces for both expert and novice users alike. User studies comparing experimental virtual-reality (VR) interfaces to industry-standard control methods reveal that VR reduces task completion times while also reducing the cognitive load for operators [31, 32]. A recent study demonstrates that, even when the ROV control method is left unchanged, a 3D VR interface increases pilots' sense-of-presence over a conventional 2D visual interface and reduces task completion time by more than 50% [33].

Similar to how interface improvements can reduce an operator's cognitive load and task completion times, relaxing human proprioception and motor control requirements can further reduce cognitive demand. Natural language and gesture-based interfaces provide a succinct mechanism for high-level, goal-directed control. If made sufficiently expressive, these interfaces have the potential to increase task efficiency (i.e., speed and precision) by decoupling human operator dexterity from manipulator control [34]. Natural language and gestures are intuitive and thereby provide a means of command and control that is accessible to a diverse user base with little prior training.

Increasing robot autonomy can also improve a system's usability, especially among novices. Senft et al. [35] proposed a task-level authoring approach, which enables novice operators to control a semi-autonomous robot by specifying actions rather than direct motions with an augmented reality interface. Their user study results demonstrate that this approach significantly reduces the operator workload, improves usability, and reduces task completion times when compared with the manipulator's standard control interface. Lawrance et al. [36] developed a system for controlling a semi-autonomous underwater vehicle (sAUV), which incorporates automated path planning, generates a sonar-based 3D scene reconstruction, and estimates a user's skill level to provide variable assistance accordingly to better support novices.

Based on our research, we propose the SHared Autonomy for Remote Collaboration (SHARC) framework [37], which enables remote scientists to participate in shipboard operations via a VR or desktop interface across a low-bandwidth connection. This framework was previously used in a field demonstration to enable shore-based users without prior piloting experience to successfully collect seafloor samples across a satellite connection [37]. In this paper, we present the design, implementation, and evaluation of this framework. The contributions of this paper can be summarized as follows:

1) We present the design and implementation of the SHARC framework, which extends prior approaches to supervisory control by engaging multiple simultaneous shore-based users and integrating natural user interfaces, automated subroutines, and a 3D scene reconstruction into a VR-based display.
2) We demonstrate that SHARC enables users (regardless of piloting experience) to perform comparably to experienced pilots using a conventional controller under

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

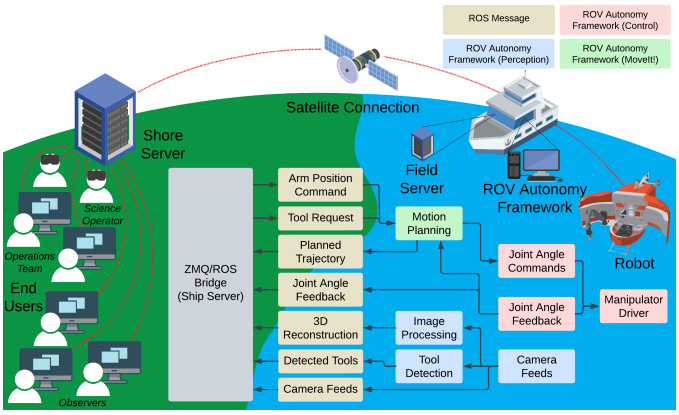

Fig. 1. Key components within the SHARC Framework. Vehicle data (e.g., camera feeds and manipulator joint position feedback) is sent via satellite communications from the field to the shore-side server that handles the distribution of data to remote users. SHARC utilizes components of the ROV autonomy framework described in Billings et al. [38].

full-bandwidth conditions (e.g., ship-side control across a tether).

3) We demonstrate that SHARC out-performs the conventional controller in low-bandwidth conditions (e.g., shore-side control without a tether) for both novice users and trained pilots.

II. SHARC Framework

The SHared Autonomy for Remote Collaboration (SHARC) framework consists of four primary components: the autonomy framework, the field server, the shore server, and the shore-side user interface. Figure 1 provides an overview of key components within this framework. The remainder of this section describes each of these components in detail.

A. ROV Autonomy Framework

On the vehicle side, SHARC extends the autonomy framework proposed by Billings et al. [38], which implements visual processes for tool detection and scene reconstruction, and exposes an API for the planning and execution of high-level manipulation tasks (e.g., tool pick-up, Euclidean space end-effector control). This autonomy framework is built on the Robot Operating System (ROS1) and leverages the MoveIt motion planning framework [39]. The implementation used in SHARC significantly improves upon the original framework by adding a PID controller on each manipulator joint, which effectively compensates for the hydraulic manipulator's hysteresis and control offsets reported by Billings et al. [38] with the addition of a tuned integral term.

The underlying autonomy framework was originally designed to support safe development, testing, and in-field deployment of automation methods on tethered ROV-manipulator systems via ship-side control [38]. SHARC's addition of a ship-to-shore data pipeline and end-user interface notably extends the framework's supervision and control capabilities to shore-based users.

The autonomy framework's vision system leverages known geometric features and AprilTag [40] fiducials within the structured ROV tool-tray to track tools and re-configurable vehicle components (e.g., door position and tool basket orientation), which can be robustly detected even in visually degraded conditions. However, computing a 3D reconstruction of the external working environment from stereo camera frames relies on natural features, which are sensitive to lighting quality and turbidity, as demonstrated by Billings et al. [38]. By providing users with an interactive 3D visualization of the natural scene reconstruction with projections of the known vehicle and object models, SHARC effectively leverages human perception to recognize gaps in the scene reconstruction. With SHARC, users can still safely plan and execute manipulation tasks in visually degraded conditions by relying on strong geometric cues from the structured workspace.

B. User Interface

SHARC's user interface design aims to reduce the bandwidth required for operation by enabling safe and effective operator control with low data update rates. While direct teleoperation with conventional controllers typically requires a data update rate exceeding 10 Hz, SHARC remains functional with update rates that are two orders-of-magnitude lower by leveraging the aforementioned autonomy framework to adopt a supervisory control approach, which is widely used in communication-limited applications [6, 17, 20]. Similar to Walker et al. [41], SHARC users preview manipulator trajectories by controlling a virtual surrogate of the robot (Figure 2). SHARC renders the robot arm's planned trajectory and intended actions prior to execution within the context of its surrounding environment (e.g., a 3D workspace reconstruction along with the location and label of detected tools), thereby making its behavior more predictable than conventional interfaces such as the topside controller. This provides users with a responsive interface, even when data updates are infrequent. Distributing control between the operator and the robot also enables users to focus on high-level scene understanding and scientific objectives, while offloading low-level control tasks to the robot, which should reduce operators' cognitive load and lower the amount of training required. This, in turn, enables safe operations by novice users and trained pilots alike.

Inspired by the ways in which scientists and pilots communicate, SHARC enables people to use natural language speech and hand gestures to convey high-level objectives to the robot. Complex commands that would be time-consuming and difficult to execute with conventional controllers can be succinctly communicated with language. Within a matter of seconds, users can specify a task that then takes the robot several minutes to execute. In addition to reducing the cognitive load required of the operator, the intuitive nature of natural language speech and gestures minimizes the training required for operation [34, 42] and makes SHARC accessible to a diverse population of users. These natural input modalities also have the benefit of remaining functional during intermittent, low-bandwidth, and high-latency communication [34], thereby reducing telemetry requirements and enabling participation from remote users with only rudimentary Internet access.

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

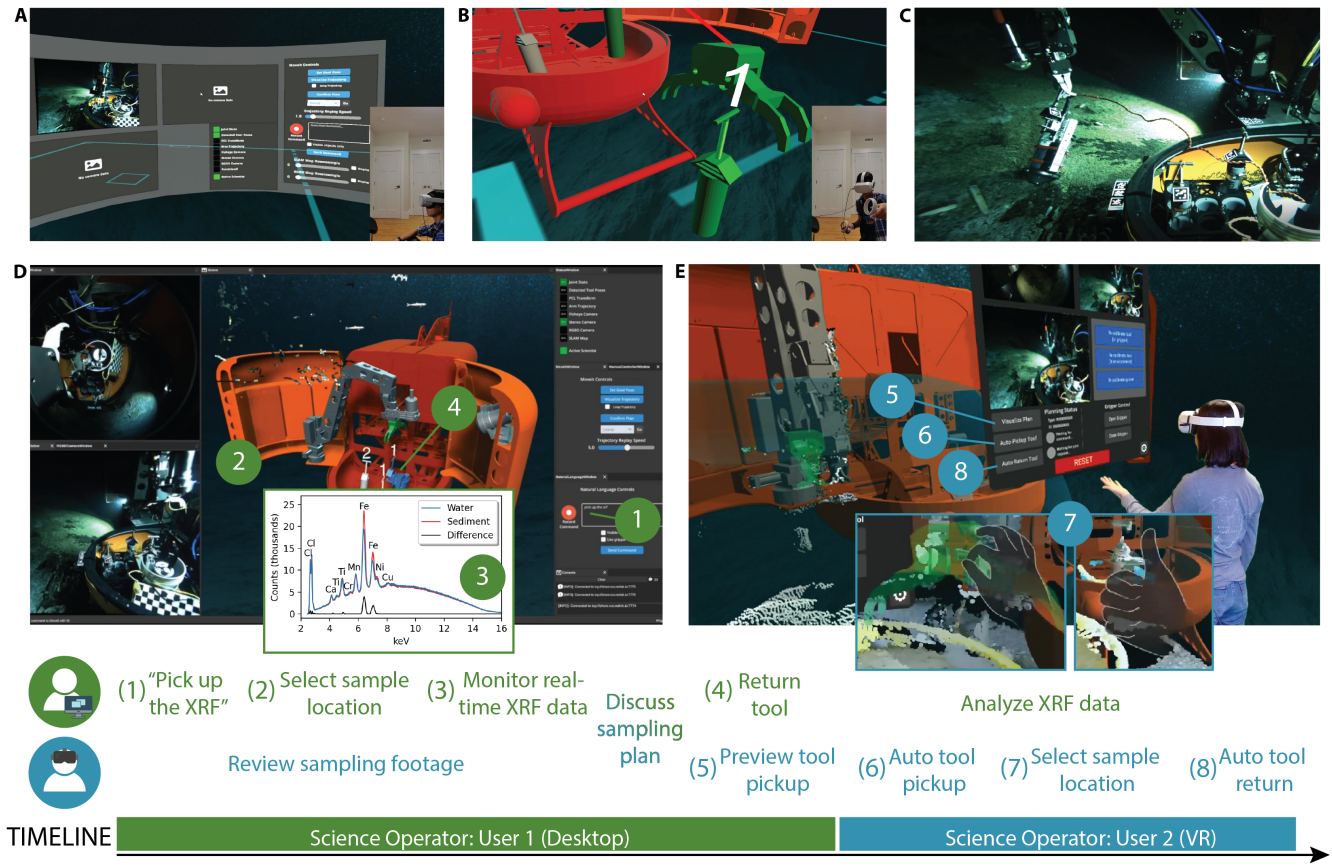

Fig. 2. SHARC interfaces and example workflow for scientific sampling. With SHARC, multiple tasks can be planned in parallel by different users via different interfaces, with execution occurring in series as tasks are completed. The (A, B) SHARC-VR and (D) SHARC-desktop interfaces enabled (C) remote scientists to direct ROV operations during sea trials at 1000 m ocean depth in September 2021 [37]. The SHARC-VR interface was revised based on user experiences during the sea trials. (E) This revised interface was used during the user study described in Section III.

SHARC's VR and desktop interfaces display a model of the manipulator's current pose with a 3D stereo reconstruction of the scene and 2D camera feeds in the background. Through these interfaces, users can collaboratively identify target sample sites based on real-time data while deferring low-level control of the manipulator to the automated system. To support a wide range of users without additional hardware, we developed a desktop interface and made it cross-compatible with commonly used operating systems (e.g., Windows, macOS, and GNU/Linux). We also developed a VR interface to complement the desktop interface, as an immersive 3D stereoscopic view of the workspace is known to enhance users' contextual awareness and overall scene understanding compared to a 2D monoscopic display [33]. Similar to the interface used by the DexROV project [23, 25], SHARC renders prior models (e.g., tools, the manipulator, and the vehicle) in context of the 3D reconstruction generated by the stereo camera. These models reduce the bandwidth required to communicate the position of structured elements within the workspace. Figure 2 (A, B, D) illustrates the interfaces used during our field demonstration [37].

Informed by our experiences with SHARC's VR interface during these field operations, we sought to improve the interface's usability for novices. Incorporating passthrough augmented reality (AR) provides users with continual awareness of their physical environment, which mitigates inadvertent collisions with objects in their personal space. Additionally, users encountered difficulties in recalling controller key mappings with the original interface, which prompted us to transition to hand tracking and gesture recognition for enhanced control. Alignment issues during tool pick-up and return were also prevalent with the original interface. Automating these structured tasks helped improve efficiency while reducing the frequency of errors. This revised interface (Figure 2E) was employed during user testing. Figure 2 illustrates example workflows using SHARC.

C. Data Pipeline

SHARC's architecture is designed to facilitate real-time collaboration among remote users and support multiple simultaneous operations. This can increase operations tempo by parallelizing sampling tasks and data analysis with different instrumentation, as illustrated in Figure 2. For example, during our field demonstration [37], one scientist operated the in-situ X-Ray Fluorescence (XRF) instrument and analyzed its data in real-time while another scientist (located in a different geographic region) cooperatively planned manipulator trajectories to potential push core sites. By involving shore-

based scientists, SHARC enables operators to take advantage of resources that are difficult for ship-based operators to access over a ship's satellite Internet connection, such as cloud-based speech and natural language processing services.

SHARC implements an efficient, bi-directional data transmission pipeline between the vehicle and the remotely connected users. The processing between the two sides of this network is segregated into two servers, the "field" server and the "shore" server, which are connected, generally via satellite, through an IP network. The field server interfaces directly with the vehicle's autonomy framework through a software bridge to its ROS system and encodes data messages with the high-performance messaging library, ZeroMQ (ZMQ), for transmission to the shore server.

ZMQ is used to handle field-to-shore communications rather than ROS since it is a transport-only middleware that leaves data marshalling to the user, which enables us to employ a better compression and representation format for the messages. Unlike ROS, ZMQ also allows for the creation of hierarchical network architectures. In this configuration, nodes in the network can only receive a subset of the messages available to their parent nodes. This allows us to integrate a permissions layer that prevents shore-based users from directly accessing the robot's internal communications, which would have serious security and safety implications. To improve the data transmission speed, the UDP communication protocol is used for relaying vehicle data. Meanwhile, we use TCP for relaying user inputs due to its reliability.

SHARC uses ZMQ for message passing between servers and client devices, and distributes bandwidth among independent data streams. This architecture ensures that smaller data packets (e.g., joint angle feedback) are not disproportionately delayed while waiting for the complete transmission of larger packets (e.g., camera footage). Field operators are also given control over throttling or pausing any data streams between the field and shore servers to limit bandwidth usage.

The shore server implements a star topology network to the connected user clients, enabling efficient scaling of the network to any number of users without increasing processing requirements on the field-side compute. It also implements an authentication system to handle incoming user requests. Any number of clients can be connected as "observers" to the shore server, which enables users from the general public to access live data streams without control privileges. Meanwhile, authenticated members of the operations team can delegate control authority, operate payload instruments, and generate task-level plans. One member of the operations team is designated as the current "science operator," who is granted control authority for issuing natural language commands, specifying end-effector goals, and executing task-level plans. This designation prevents conflicting commands while maintaining clarity of who holds current control, and can be seamlessly handed-off between members of the operations team. Figure 2 illustrates this control hand-off process. Specific details of the software architecture for the field and shore servers are provided in Appendix B.

The shore server also implements a natural language interface that translates recorded speech into commands for the various components of the system. For example, the designated science operator can issue the verbal command "pick up the push core on the left." An on-device speech-to-text module is paired with an instance of the Distributed Correspondence Graph (DCG) probabilistic language grounding model [43, 44] to infer the corresponding structured language command. When parsing natural language commands, SHARC considers the current 3D scene understanding to distinguish tools based on their relative positions and the tool types. These commands are then sent to the field server for actuation. It is worth noting that the field server does not distinguish between speech-issued commands, hand gestures in VR, or mouse clicks on a UI button. The end-user interface and the shore server converts raw inputs into commands that the field server can interpret and execute.

TABLE I
TEST GROUP SIZES

Test group	# of participants
ROV pilots using topside controller	6
ROV pilots using SHARC-VR	5
ROV pilots TOTAL	6
Novices using topside controller	8
Novices using SHARC-VR	9
Novices TOTAL	17

III. USER STUDY DESIGN

To quantitatively compare the SHARC framework and the conventional topside controller's suitability for tetherless and shore-based operations, we conducted a user study that focused on performance metrics under low-bandwidth conditions with expert as well as novice participants. Participants completed representative manipulation tasks using the SHARC-VR interface and the topside controller in a range of simulated bandwidth conditions. The study was conducted using a laboratory-based in-air testbed equipped with a robotic manipulator arm that is identical to the system used on the Nereid Under Ice (NUI) HROV (Hybrid ROV). During testing, we measured the task completion time, success rate, precision, and accuracy to quantify a participant's performance with each interface. This section outlines the study's design and procedures, with the overall results presented in Section IV.

A. Participants

Participants in the user study included ROV pilots with extensive experience operating underwater manipulators, and novice users without any prior experience. A subset of participants used both the SHARC-VR and topside controller interfaces while a second subset used only one of the two interfaces, depending on their availability. All participants were employees at the Woods Hole Oceanographic Institution (WHOI) and were recruited as volunteers over email and selected on a first-come first-serve basis. Table I provides the number of participants who operated each interface.

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

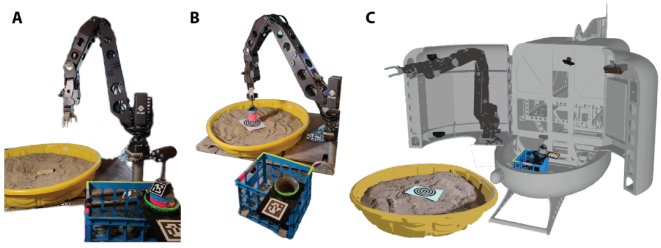

Fig. 3. The user study testbed for (A) the block retrieval task and (B) the push core sampling task. (C) The testbed has a hydraulic manipulator and is equipped with a basket for tools and samples. The manipulator, camera, tool basket, and workspace position on the testbed (colored) closely resemble the layout of the physical vehicle (displayed in monochrome, for reference)

B. Experimental Setup & Testing Procedure

The timed trials were performed using an in-air testbed setup with a fixed-base mounted seven degree-of-freedom hydraulic manipulator and complementary topside controller (Kraft TeleRobotics; Overland Park, KS) [38], which is identical to the hardware used on the NUI HROV operated by WHOI and used for our field demonstration [37]. The workspace consisted of a sandbox that approximates the reachable area of the manipulator mounted on the NUI vehicle [45]. Visualized in Figure 3, the testbed includes cameras and a tool basket with a push core, each positioned to mimic their placement onboard the NUI vehicle. Users were presented with these four camera feeds while using both interfaces. Following the approach of Billings et al. [38], AprilTags were affixed to the tools and tool basket for accurate camera-based localization. The system did not rely on fiducials in the environment outside the ROV tool-tray area, consistent with how it would be deployed in the real world. During testing, the field and shore server processes were run on the same computer with artificial software-limited data update rates to simulate particular bandwidth conditions. A second computer on the same network was used to run the SHARC-VR interface.

As a proxy for an underwater imaging system (e.g., stereo camera [46], laser scanner [47], or imaging sonar [48]), an Xbox One Kinect (Microsoft; Redmond, WA) was used to generate the 3D workspace reconstruction during the user study. The Kinect's workspace reconstruction is comparable to an underwater stereo imaging system in good lighting conditions. For a viewing range of 3 m, the metric pixel resolution for our underwater stereo system is 1.7 mm with a depth resolution of $\sim$3 cm [38], and the Kinect's metric pixel resolution is $\sim$4 mm per pixel with a depth resolution of $\sim$3 cm at the same range. While these values represent the theoretical limits of each sensor, the resolution of the reconstruction generated by both sensors tends to be lower in practice.

Participants completed a 15 minute tutorial before starting trials with the SHARC-VR interface. Before trials with the topside controller, novices completed a 6 minute tutorial followed by a 25 minute practice session with direct line-of-sight to the manipulator, while pilots were given a 10 minute "warm-up" period instead of a tutorial.

The manipulator testbed was located in an area separate from the participants for safety. During the initial topside controller training, participants stood in a designated area outside of the manipulator workspace with a direct line-of-sight to the physical arm. During testing, participants operated the arm from a separate room using either the VR or topside controller interfaces while an evaluator observed the physical arm from the training area. During trials using the SHARC-VR interface, the evaluator monitored the participant's headset view and the automated system's planned arm trajectories. The evaluator stopped the trial if the manipulator reached an unsafe configuration, an irrecoverable state was reached (e.g., dropping a tool outside the workspace), or the trial time exceeded 10 minutes. Each timed trial was labeled as "complete" when samples were successfully collected, "failed" when irrecoverable states were reached, "timed out" for incomplete attempts, or "crashed" for unsafe manipulator configurations.

The trials consisted of two different tasks: (1) picking up a wooden block and placing it in a basket, which is representative of a low-precision task (e.g., collecting rock samples); and (2) using a push core to punch a hole in a printed "bullseye" target and stowing the tool back in the tool-tray, which is representative of a higher precision task (e.g., sampling a heterogeneous microbial mat). These tasks were repeated with camera feeds and pointclouds updated at different frames-per-second (FPS) to simulate the effects of bandwidth constraints of different orders of magnitude. The block pick-up task was tested at 10 FPS, 1.5 FPS, 0.5 FPS, 0.2 FPS, and 0.1 FPS. The push core task was tested at 10 FPS, 0.5 FPS, and 0.1 FPS. In order to measure sampling precision, we recorded the distance between the center of the push core punch mark on the paper target and the printed "bullseye" for each trial.

IV. User Study Results

In contrast to conventional interfaces, SHARC enables users to operate with performance benchmarks (i.e., precision, accuracy, task time, and task success rate) comparable to that of trained pilots, regardless of their prior experience, even when faced with bandwidth limitations. We define the *Task Success Rate* at a given data framerate (FPS) as $R_{\text{FPS}} = \frac{\text{\# successes}}{\text{\# trials}}$. The *Task Success Rates* while using SHARC-VR were significantly higher than those obtained using the topside controller across nearly all tested framerates (Figure 4). This increase is most pronounced at low framerates—at 0.1 FPS, the success rate among pilots and novices were, respectively, 57% and 278% higher with SHARC than with the conventional topside controller. These results suggest that SHARC increases the probability of task success in operational settings, thereby minimizing time-consuming failures that can damage the vehicle platform, tools, or sensitive environments. Catastrophic failures that compromise platform survivability can jeopardize entire science campaigns, and thus SHARC may increase operations tempo while reducing risk.

It is notable that one pilot failed the first block pick-up trial with the topside controller, but succeeded in the final one, demonstrating that even trained pilots risk failure when not

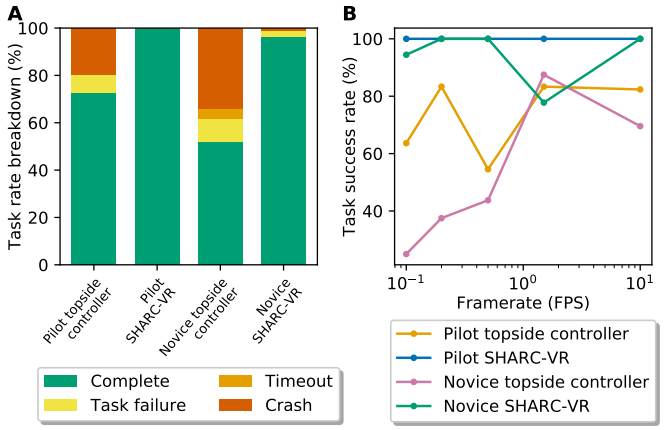

Fig. 4. *Task Success Rates* R_{FPS}. Task rate breakdown (A) among test groups and (B) expressed across framerates. Across most framerates, both trained pilots and novices had a higher *Task Success Rate* with SHARC-VR than with the topside controller.

fully familiar with a conventional controller's configuration settings. It is also worth noting that on average, trained pilots exhibited a higher success rate than novices with both interfaces, which indicates that operator experience still affects operational robustness with either interface.

Improved success rates with the SHARC-VR interface can be partially attributed to the integration of automated subroutines (e.g., tool pick-up and return). While 71% of participants (seven novices and three pilots) failed at the push core task during the pick-up or return in at least one of the trials with the topside controller, none of the participants failed at the push core task with the SHARC-VR interface. It is also worth noting that, with the topside controller, 36% of participants (three novices and two pilots) did not return the tool securely (i.e., the tool was not completely contained in the quiver at the end of the episode) in at least one of the trials. Although we still counted these instances as successful task completions, these improper returns could potentially lead to lost samples during transit in real-world field missions.

The gesture-based controls also proved to be less error-prone than keybinding-based input mechanisms (e.g., topside controller and handheld VR controllers). Although the conventional topside controller interface had buttons to "lock" the gripper in place to make it easier to hold an object after it has been grasped, most novices did not use the feature. One novice who attempted to use this feature confused the keybindings and accidentally dropped the tool outside the workspace shortly after taking it out of the quiver. Controller-based input also proved to be difficult for users in the original SHARC-VR interface, which uses handheld VR controllers. When using these controllers, users often forgot the keybindings and miskeyed commands during development testing. Meanwhile, during our user study with the gesture-based SHARC-VR interface, none of the users failed at a task due as a result of confusing the gestures. Although one error was caused by a user accidentally clicking the tool-pickup button when attempting to return the tool, this incident is more likely indicative of a visual interface design flaw rather than an issue with the gesture-based input. It is also worth noting that while this error caused the task to take longer, it did not result in an unrecoverable state (e.g., dropping the tool) like the topside controller did.

The frequency of crashes with the respective interfaces indicates that the 3D reconstruction and prior model rendering in the SHARC-VR interface improved users' spatial awareness over the 2D camera-based interface with the topside controller. Although 64% of participants (six novices and three pilots) crashed the arm with the topside controller, only one user crashed the arm with the SHARC-VR interface. This crash occurred early in testing, when the user was likely still becoming familiar with the interface. The other two failures with the SHARC-VR interface occurred when two novice users dropped the block outside the crate. In both cases, the users dropped the block on the edge of the crate, and the block bounced out. In contrast, five users (including two pilots) dropped the block outside the crate with the topside controller. Although further improvements could be made to the SHARC-VR interface to additionally improve the users' spatial awareness, these results demonstrate a significant improvement over the 2D camera-based interface typically used during ROV operations.

Figures 5A and 5B display the recorded *Task Completion Times* $T_{i,\text{FPS}}$ for block-pickup and push core trials across framerates. Figure 5C shows the *Expected Task Times* $\mathbb{E}[T_{\text{FPS}}]$ across framerates, computed as the average recorded *Task Completion Time* $\bar{T}_{\text{FPS}}$ divided by the *Task Success Rate*

$$\mathbb{E}[T_{\text{FPS}}] = \frac{\bar{T}_{\text{FPS}}}{R_{\text{FPS}}}, \quad \text{where} \quad \bar{T}_{\text{FPS}} = \frac{1}{n_{\text{FPS}}} \sum_{i=1}^{n_{\text{FPS}}} T_{i,\text{FPS}}.$$

Mathematically, this considers each trial as an independent event with a probability R_{FPS} of success, which implicitly assumes participants can retry failed tasks until successful. In Figure 5, we fit a power curve to the topside controller completion times, which increase exponentially with decreasing framerates. For SHARC-VR, we fit a linear trend to the completion times because the times remain relatively constant across framerates.

For statistical analysis, we consider results beyond two standard deviations (2σ, 95% CI) to be statistically significant. Given that our study involved 23 total participants, each participant represents roughly 4% of the total, and thus a 95% CI is the upper limit of what we can reasonably infer from our dataset.

At $10\,$FPS, there was no statistically significant difference (95% CI) between participants' expected time with both interfaces. At $0.5\,$FPS or lower, the expected times for both pilots and novices were lower with SHARC-VR than with the topside controller, and this difference increased as the framerate decreased. At $0.1\,$FPS, the expected time was $2\times$ faster for pilots with SHARC-VR than with the topside controller and $7.6\times$ faster for novices. In operational settings, these observed differences would likely be magnified since additional time would be needed to recover from failures, which occurred more frequently in trials with the topside controller than with SHARC-VR. Across all framerates, there is no statistically significant difference (95% CI) between the expected time for

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

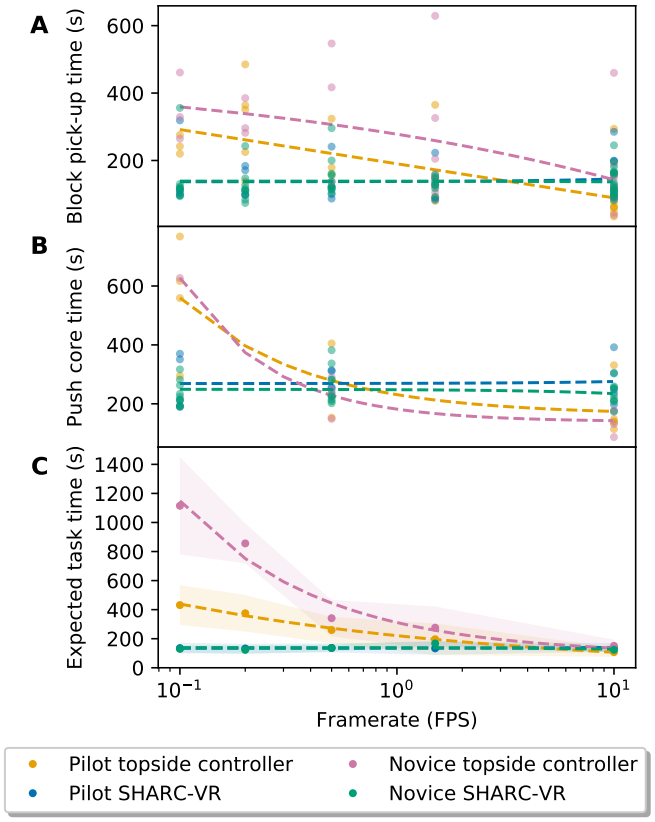

Fig. 5. Plots of *Task Completion Times*, $T_{i,\text{FPS}}$, for the (A) block pick-up and (B) push core tasks as well as (C) the *Expected Task Times*, $\mathbb{E}[T_{\text{FPS}}]$. A power and linear trend are fitted to the topside controller and SHARC-VR data, respectively. The 2σ distribution of points at each framerate is shaded in C. At lower framerates, pilots and novices complete both the block pick-up and push core tasks in less time with SHARC-VR than with the topside controller. This difference is more pronounced among the *Expected Task Times*, which factors in the *Task Success Rate*.

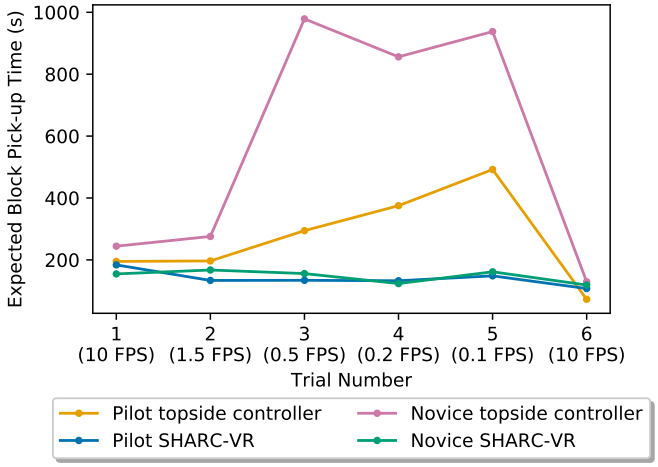

Fig. 6. *Expected Task Times* for the block pick-up task plotted by trial number ($\mathbb{E}[T_{\text{trial}}]$). Initial and final trials are conducted at 10 FPS to control for framerate. SHARC-VR times exhibit a slight negative correlation with trial number independent of framerate, while topside controller times appear to be framerate dependent.

pilots and novices when using SHARC-VR, and the variance in times with SHARC-VR is less than that of the topside controller for both groups.

All participants operated the arm in smooth continuous motions when using the topside controller in high-bandwidth conditions (i.e., 10 FPS). At 1.5 FPS, some participants switched to an incremental positioning approach, where they made a small movement and then waited for a data update to verify the arm's position before proceeding further. At lower framerates, all users adopted the incremental positioning approach. Naturally, as the framerate decreased, users spent more time waiting for data updates between movements, which in turn increased task completion times. Meanwhile, the SHARC-VR interface adopts a plan-then-execute approach. Although this approach is similar to incremental positioning, larger arm motions can be safely achieved between each data update due to the automated trajectory planning and manipulator control processes. This kept the SHARC-VR task completion times low, even in low-framerate settings. Meanwhile, large arm motions with the topside controller in low-framerate conditions were the primary cause of crashes.

In order to more closely examine the learning among pilots and novices during testing, we also compute the *Expected Task Time* for the block pick-up task based on the trial number instead of framerate, which we define as $\mathbb{E}[T_{\text{trial}}]$. Figure 6 presents this analysis. Both novices and pilots using SHARC-VR exhibited small but consistent speed improvements with each successive trial regardless of framerate, indicating that *Expected Task Times* are more strongly correlated with learning than framerate when using the SHARC-VR interface. In contrast, *Expected Task Times* for the topside controller interface increased exponentially as framerate decreased, with this effect dominating any improvement achieved through learning.

TABLE II
EXPECTED TASK TIME IMPROVEMENT METRICS

	Initial Time (s)	Final Time (s)	Absolute Change (s)	Relative Change
Pilot topside controller	195.0	72.7	122.3	63%
Pilot SHARC-VR	183.8	107.4	76.4	42%
Novice topside controller	244.2	129.3	114.9	47%
Novice SHARC-VR	154.7	118.8	35.9	23%

Table II reports the differences between the expected times of the first block pick-up trial ($\mathbb{E}[T_1]$) and the last trial ($\mathbb{E}[T_6]$). During the initial trial at 10 FPS, novices using SHARC-VR exhibited the fastest *Expected Task Time*. However, during the final trial at 10 FPS ($\sim$30 minutes into testing), pilots using the topside controller were the fastest, which is unsurprising given their familiarity with conventional manipulation systems. As the data illustrates, the differences between the *Expected Task Times* for the initial ($\mathbb{E}[1]$) and final ($\mathbb{E}[6]$) block pick-up trials at 10 FPS are greater when using the topside controller than when using SHARC-VR for both pilots and novices. This implies that the topside controller has an inherently steeper learning curve than SHARC-VR, and that operator performance is highly dependent on familiarity with the topside configuration (e.g., camera views, workspace layout, controller

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

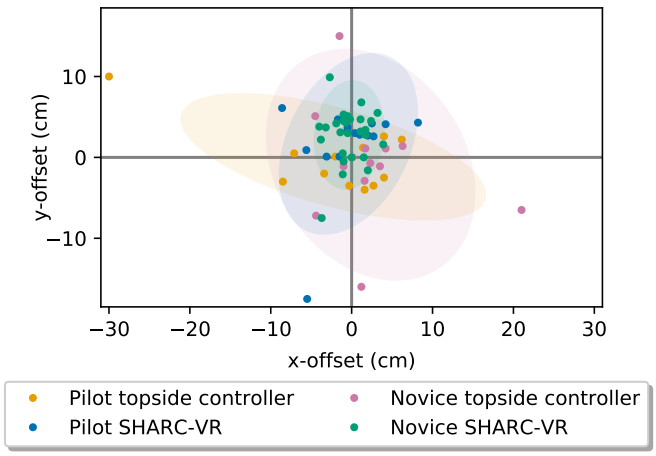

Fig. 7. Map of push core placement locations by test group relative to the target center. Confidence ellipses (2σ) are shown for each group. For both pilots and novices, push core locations achieved using the SHARC-VR formed a tighter confidence ellipse than those with the topside controller.

TABLE III
ENSEMBLE ACCURACY AND PRECISION OF PUSH CORE PLACEMENT

	↓ Accuracy (cm)	↓ Precision (cm)
Pilot topside controller	2.3	7.0
Pilot SHARC-VR	1.7	4.9
Novice topside controller	2.5	6.8
Novice SHARC-VR	2.8	3.3

settings, and manipulator arm response).

To quantify the participants' accuracy and precision with the two interfaces, the push core locations were recorded relative to the center of a target. Figure 7 illustrates this data, while Table III presents the average accuracy and precision for each group. As the data indicates, pilots using SHARC-VR exhibited the highest accuracy across all test groups, and the VR interface increased precision for both pilots and novices. Novices using SHARC-VR had the best precision, but the worst accuracy of all test groups. On average, using the VR interface instead of the topside controller decreased the variance in the participants' placement positions by $\sim$30% for pilots and $\sim$52% for novices. Pilots and novices using the topside controller had comparable accuracy and precision.

These results provide additional evidence that the 3D reconstruction and prior model rendering in the SHARC-VR interface improves users' spatial awareness compared to the 2D camera-based interface. In SHARC-VR, users viewed their sampling plan from multiple perspectives in the context of the 3D reconstruction before execution. Consequently, the accuracy with SHARC-VR interface is likely dependent on the 3D reconstruction accuracy. During the user study, objects in the reconstruction deviated as much as 3 cm from their actual position due to the Kinect's limitations. 50% of users (two novices and all five pilots) attempted to account for this discrepancy by cross-referencing the push core position with the camera, while the remainder relied exclusively on the 3D reconstruction.

Moreover, the automated tool-pickup process likely contributed to SHARC-VR's improved performance. With the topside controller, some users encountered challenges in aligning the tool with the gripper, which resulted in less precise sampling and tool return maneuvers. Notably, the improper tool returns discussed earlier were caused by misaligned grasps during pickup. In another case, one user returned the tool to the basket instead of the quiver since the tool's awkward angle within the gripper prevented proper alignment with the quiver. During the push core sampling task, two users missed the target entirely due to the misaligned grasp (for the dataset, these trials were still counted as successful task completions. These push core placements were estimated based on a top-down photo of the sandbox). In contrast, the auto-pickup subroutine integrated into SHARC-VR facilitated more consistent grasps, and all users were able to place the push core within target and return the tool successfully across the range of tested framerates with SHARC.

V. DISCUSSION

As our experimental results indicate, SHARC enables novice users to match the performance of trained pilots who use a conventional controller in tethered high-bandwidth conditions, while improving operational efficiency and robustness in low-bandwidth conditions for both pilot and novice users. The SHARC-VR interface renders prior models in the context of the current 3D reconstruction, which improved users' spatial awareness of the workspace and reduced the number of crashes, particularly in low-bandwidth conditions. The automated tool pickup and return processes improved tool grasps, which consequently improved task precision during push core sampling and tool return. The gesture-based input also improved usability among users, and reduced the frequency of miskeyed commands in comparison to the original controller-based SHARC-VR interface and topside controller. These results highlight SHARC's utility in enabling delicate operations in unstructured environments under bandwidth-limited conditions, which may be extensible to other sensitive domains where dexterity is required (e.g., nuclear decommissioning [49], deep space operations [50], and unexploded ordnance/disposed military munition remediation [51]).

SHARC currently supports operations within semi-static environments, where dynamic changes in the scene are assumed to progress slowly relative to the data update delay induced by bandwidth and latency. Satellite-based communications with Starlink currently transmit at 8–25 Mb/s, with a <99 ms latency for marine applications [15]. Meanwhile, existing commercial through-water optical modems can transmit up to 10 Mb/s with µs-ms latency [52], and the median mobile Internet worldwide transmits 11.3 Mb/s with a 27 ms latency [53]. With an update packet size of 265 kb or less, which should be sufficient for robotic manipulation with framerate-limited feedback [38], the total data update delay from a tetherless robot to shore-side users should be on the order of a couple of seconds (for communications across a through-water optical modem and satellite, 8 Mb/s translates to an update rate of >10 FPS, and the combined latency adds up to <2 s). This should be sufficient to support SHARC-based operations, given that

participants using SHARC achieved good performance on representative sampling tasks with data update delays up to 10 seconds (10 seconds is the maximum delay with a 0.1 FPS update rate).

Meanwhile, lower bandwidth acoustic modems can transmit at 5.3 kb/s ($\sim$0.02 FPS) [54] with $\sim$8 second latency at full ocean depth. This translates to a total data update delay of up to a minute, which requires a semi-static environment. Future work could relax this semi-static scene assumption and enable operations across acoustic communications by incorporating dynamic replanning methods [55, 56, 57] to identify changes to the scene and adapt the reconstruction or manipulation plan as necessary, without human intervention.

SHARC uses the stereo camera setup described in Billings et al. [38] to perceive the environment, which is sensitive to lighting quality and turbidity. In its current form, SHARC leverages human perception to recognize gaps in coverage due to poor visibility conditions and to then adapt the sampling plan accordingly. Ongoing research directions include multi-sensor fusion [58] (e.g., optical and acoustic imaging) to reduce errors and uncertainty in scene reconstruction, natural object tracking for closed-loop visual servoing [59], and semantic mapping supported by natural language queues provided by the human operators [60]. These capabilities will build on the complementary strengths of human perception and contextual awareness with machine processing and control to enable greater automation of tasks while lowering operational risks within unstructured environments despite low-visibility conditions.

SHARC is platform-independent and can be readily integrated into other underwater systems equipped with at least one robotic manipulator, a workspace imaging sensor, and a data link to one or more operators. SHARC's current implementation supports single-manipulator platforms, but future work could extend the framework to multi-manipulator systems [61] distributed across one or more vehicles with more than one concurrent operator [62]. Coordinated manipulation could enable vehicles to manipulate objects too large or too heavy for one manipulator to handle alone [63, 64], complete tasks that require higher dexterity or redundant degrees-of-freedom [64], and operate more efficiently by parallelizing tasks. For a single operator, a dual manipulator setup can potentially reduce cognitive load since human operators are intuitively familiar with bi-manual control [64].

VI. Conclusion

Currently, deep-ocean exploration requires costly, highly specialized infrastructure, and limited crew berthing on ships restricts the number of onboard participants during ROV field operations. This presents multiple barriers to access for those who may lack the resources, time, or physical ability required for at-sea participation in oceanographic research. In this paper, we present the design and evaluation of the SHARC framework, which extends prior supervisory control frameworks by integrating natural user interfaces (e.g., language and hand gestures), VR, and automated processes to enable collaborative operations by multiple simultaneous remote users. As our field demonstration [37] and user study results indicate, SHARC enables shore-side novice users to conduct manipulator operations across a bandwidth-limited satellite connection using only a basic Internet connection and consumer-grade hardware, which provides a promising avenue for increasing access to deep-sea research. Future work could improve operations with SHARC by further reducing the bandwidth requirements to enable operations across acoustic communications, incorporating multi-sensor fusion to enable operations in low-visibility conditions, and extending SHARC to multi-manipulator systems for bi-manual control.

References

[1] A. G. Glover and C. R. Smith, "The deep-sea floor ecosystem: current status and prospects of anthropogenic change by the year 2025," *Environmental Conservation*, vol. 30, no. 3, pp. 219–241, 2003.

[2] A. J. Jamieson, B. Boorman, and D. O. Jones, "Deep-sea benthic sampling," *Methods for the study of marine benthos*, pp. 285–347, 2013.

[3] J. Hegde, I. B. Utne, and I. Schjølberg, "Applicability of current remotely operated vehicle standards and guidelines to autonomous subsea IMR operations," in *Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering (OMAE)*, vol. 56550, 2015.

[4] P. M. Pilarski, M. R. Dawson, T. Degris, J. P. Carey, and R. S. Sutton, "Dynamic switching and real-time machine learning for improved human control of assistive biomedical robots," in *Proceedings of the IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012, pp. 296–302.

[5] A. Thobbi, Y. Gu, and W. Sheng, "Using human motion estimation for human-robot cooperative manipulation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 2873–2878.

[6] T. B. Sheridan, "Space teleoperation through time delay: Review and prognosis," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 592–606, 1993.

[7] J. Austin, J. Bernhard, B. Blomquist, S. Carbotte, G. Cutter, A. DeSilva, A. Doyle, R. Keil, S. Z. Kelety, N. Rabalais, E. Roth, and J. Swift, "U.s. academic research fleet improvement plan 2019 update," 2019.

[8] N. Farr, A. Bowen, J. Ware, C. Pontbriand, and M. Tivey, "An integrated, underwater optical /acoustic communications system," in *OCEANS'10 IEEE SYDNEY*, 2010, pp. 1–6.

[9] D. Centelles, A. Soriano, R. Marin, and P. J. Sanz, "Wireless hrov control with compressed visual feedback using acoustic and rf links," *Journal of Intelligent & Robotic Systems*, vol. 99, no. 3, pp. 713–728, 2020.

[10] A. J. Dalpe, S. Suman, M. V. Jakuba, and A. Bowen, "Teleoperation of remotely operated vehicles: Development, challenges, and future directions," in *OCEANS 2022, Hampton Roads*, 2022.

[11] D. Lester and H. Thronson, "Human space exploration and human spaceflight: Latency and the cognitive scale

of the universe," *Space Policy*, vol. 27, no. 2, pp. 89–93, 2011.

[12] B. J. Mellinkoff, M. M. Spydell, W. Bailey, and J. O. Burns, "Quantifying operational constraints of low-latency telerobotics for planetary surface operations," in *Proceedings of the IEEE Aerospace Conference*, 2018.

[13] J. O. Burns, B. Mellinkoff, M. Spydell, T. Fong, D. A. Kring, W. D. Pratt, T. Cichan, and C. M. Edwards, "Science on the lunar surface facilitated by low latency telerobotics from a lunar orbital platform-gateway," *Acta Astronautica*, vol. 154, pp. 195–203, 2019.

[14] A. D. Bowen, D. R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J. C. Kinsey, M. Heintz, G. McDonald, D. B. Peters *et al.*, "The Nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth," in *OCEANS 2008*, 2008.

[15] Starlink, "Starlink for boats," 2024, accessed on Feb 27, 2024. [Online]. Available: https://www.starlink.com/boats.

[16] E. Simetti, "Autonomous underwater intervention," *Current Robotics Reports*, vol. 1, pp. 117–122, 2020.

[17] K. Hambuchen, J. Marquez, and T. Fong, "A review of nasa human-robot interaction in space," *Current Robotics Reports*, vol. 2, no. 3, pp. 265–272, 2021.

[18] T. Sheridan, J. Thompson, J. Hu, and M. Ottensmeyer, "Haptics and supervisory control in telesurgery," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 41, no. 2, 1997, pp. 1134–1137.

[19] T. T. Blackmon and L. W. Stark, "Model-based supervisory control in telerobotics," *Presence: Teleoperators & Virtual Environments*, vol. 5, no. 2, pp. 205–223, 1996.

[20] G. Gonzalez, M. Balakuntala, M. Agarwal, T. Low, B. Knoth, A. W. Kirkpatrick, J. McKee, G. Hager, V. Aggarwal, Y. Xue *et al.*, "Asap: A semi-autonomous precise system for telesurgery during communication delays," *IEEE Transactions on Medical Robotics and Bionics*, vol. 5, no. 1, pp. 66–78, 2023.

[21] A. Birk, T. Doernbach, C. A. Mueller, T. Łuczyński, A. G. Chavez, D. Koehntopp, A. G. Kupcsik, S. Calinon, A. K. Tanwani, G. Antonelli, P. D. Lillo, E. Simetti, G. Casalino, G. Indiveri, L. Ostuni, A. Turetta, A. Caffaz, P. Weiss, T. Gobert, B. Chemisky, J. Gancet, T. Siedel, S. Govindaraj, X. Martínez, and P. Letier, "Dexterous underwater manipulation from onshore locations: Streamlining efficiencies for remotely operated underwater vehicles," *IEEE Robotics & Automation Magazine*, vol. 25, no. 4, pp. 24–33, 2018.

[22] T. Łuczyński, P. Łuczyński, L. Pehle, M. Wirsum, and A. Birk, "Model based design of a stereo vision system for intelligent deep-sea operations," *Measurement*, vol. 144, pp. 298–310, 2019.

[23] A. G. Chavez, Q. Xu, C. A. Mueller, S. Schwertfeger, and A. Birk, "Adaptive navigation scheme for optimal deep-sea localization using multimodal perception cues," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7211–7218.

[24] T. Łuczyński, T. Fromm, S. Govindaraj, C. A. Mueller, and A. Birk, "3D grid map transmission for underwater mapping and visualization under bandwidth constraints," in *OCEANS 2017-Anchorage*, 2017.

[25] C. A. Mueller, T. Doernbach, A. G. Chavez, D. Koehntopp, and A. Birk, "Robust continuous system integration for critical deep-sea robot operations using knowledge-enabled simulation in the loop," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1892–1899.

[26] P. A. D. Lillo, E. Simetti, D. De Palma, E. Cataldi, G. Indiveri, G. Antonelli, and G. Casalino, "Advanced ROV autonomy for efficient remote control in the dexrov project," *Marine Technology Society Journal*, vol. 50, no. 4, pp. 67–80, 2016.

[27] A. K. Tanwani and S. Calinon, "A generative model for intention recognition and manipulation assistance in teleoperation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 43–50.

[28] M. J. Zeestraten, I. Havoutis, and S. Calinon, "Programming by demonstration for shared control with an application in teleoperation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1848–1855, 2018.

[29] I. Havoutis and S. Calinon, "Learning from demonstration for semi-autonomous teleoperation," *Autonomous Robots*, vol. 43, pp. 713–726, 2019.

[30] T. Yoneda, L. Sun, G. Yang, B. Stadie, and M. Walter, "To the noise and back: Diffusion for shared autonomy," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[31] A. Singh, S. H. Seo, Y. Hashish, M. Nakane, J. E. Young, and A. Bunt, "An interface for remote robotic manipulator control that reduces task load and fatigue," in *Proceedings of the International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2013, pp. 738–743.

[32] M. Wonsick and T. Padir, "A systematic review of virtual reality interfaces for controlling and interacting with robots," *Applied Sciences*, vol. 10, no. 24, 2020.

[33] A. Elor, T. Thang, B. P. Hughes, A. Crosby, A. Phung, E. Gonzalez, K. Katija, S. H. D. Haddock, E. Martin, B. E. Erwin, and L. Takayama, "Catching jellies in immersive virtual reality: A comparative teleoperation study of ROVs in underwater capture tasks," in *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST)*, 2021.

[34] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 25–55, 2020.

[35] E. Senft, M. Hagenow, K. Welsh, R. Radwin, M. Zinn, M. Gleicher, and B. Mutlu, "Task-level authoring for remote robot teleoperation," *Frontiers in Robotics and AI*, vol. 8, p. 707149, 2021.

[36] N. Lawrance, R. DeBortoli, D. Jones, S. McCammon, L. Milliken, A. Nicolai, T. Somers, and G. Hollinger, "Shared autonomy for low-cost underwater vehicles," *Journal of Field Robotics*, vol. 36, no. 3, pp. 495–516,

2019.

[37] A. Phung, G. Billings, A. F. Daniele, M. R. Walter, and R. Camilli, "Enhancing scientific exploration of the deep sea through shared autonomy in remote manipulation," *Science Robotics*, vol. 8, no. 81, 2023.

[38] G. Billings, M. R. Walter, O. Pizarro, M. Johnson-Roberson, and R. Camilli, "Towards automated sample collection and return in extreme underwater environments," *Field Robotics*, vol. 2, 2022.

[39] S. Chitta, I. Sucan, and S. Cousins, "MoveIt!" *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[40] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4193–4198.

[41] M. E. Walker, H. Hedayati, and D. Szafir, "Robot teleoperation with augmented reality virtual surrogates," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2019, pp. 202–210.

[42] D. Fogli, L. Gargioni, G. Guida, and F. Tampalini, "A hybrid approach to user-oriented programming of collaborative robots," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102234, 2022.

[43] T. M. Howard, S. Tellex, and N. Roy, "A natural language planner interface for mobile manipulators," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6652–6659.

[44] I. Chung, O. Propp, M. R. Walter, and T. M. Howard, "On the performance of hierarchical distributed correspondence graphs for efficient symbol grounding of robot instructions," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10 2015, pp. 5247–5252.

[45] A. D. Bowen, D. R. Yoerger, C. C. German, J. C. Kinsey, M. V. Jakuba, D. Gomez-Ibanez, C. L. Taylor, C. Machado, J. C. Howland, C. L. Kaiser, M. Heintz, C. Pontbriand, S. Suman, L. O'Hara, J. Bailey, C. Judge, G. McDonald, L. L. Whitcomb, C. J. McFarland, and L. A. Mayer, "Design of Nereid-UI: A remotely operated underwater vehicle for oceanographic access under ice," in *2014 Oceans-St. John's*, 2014.

[46] S. Ishibashi, "The stereo vision system for an underwater vehicle," in *Oceans 2009-Europe*, 2009, pp. 1–6.

[47] A. Palomer, P. Ridao, D. Youakim, D. Ribas, J. Forest, and Y. Petillot, "3D laser scanner for underwater manipulation," *Sensors*, vol. 18, no. 4, 2018.

[48] T. Guerneve and Y. Petillot, "Underwater 3D reconstruction using blueview imaging sonar," in *OCEANS 2015-Genova*, 2015.

[49] C. Pohl, K. Hitzler, R. Grimm, A. Zea, U. D. Hanebeck, and T. Asfour, "Affordance-based grasping and manipulation in real world applications," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 9569–9576.

[50] K. K. Babarahmati, C. Tiseo, Q. Rouxel, Z. Li, and M. Mistry, "Robust high-transparency haptic exploration for dexterous telemanipulation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 10 146–10 152.

[51] H. F. Morrison, A. Becker, T. Smith, and E. Gasperikova, "Detection and classification of buried metallic objects," in *Procfeedings of the EAGE Conference & Exhibition*, 2002.

[52] B. Alexander, S. Felix, and M. Igor, "Practical applications of free-space optical underwater communication," in *Proceedings of the Underwater Communications and Networking Conference (UComms)*, 2021.

[53] Ookla, "Speedtest global index," 2024, accessed on Feb 27, 2024. [Online]. Available: https://www.speedtest.net/global-index.

[54] S. Singh, S. E. Webster, L. Freitag, L. L. Whitcomb, K. Ball, J. Bailey, and C. Taylor, "Acoustic communication performance of the WHOI micro-modem in sea trials of the Nereus vehicle to 11,000 m depth," in *OCEANS 2009*, 2009.

[55] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite RRTs for rapid replanning in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 1603–1609.

[56] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 1478–1483.

[57] T. Lai and F. Ramos, "Ltr*: Rapid replanning in executing consecutive tasks with lazy experience graph," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 8784–8790.

[58] G. Billings, R. Camilli, and M. Johnson-Roberson, "Hybrid visual SLAM for underwater vehicle manipulator systems," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6798–6805, 2022.

[59] S. Sivčev, M. Rossi, J. Coleman, G. Dooly, E. Omerdić, and D. Toal, "Fully automatic visual servoing control for work-class marine intervention ROVs," *Control Engineering Practice*, vol. 74, pp. 153–167, 2018.

[60] M. Walter, S. Patki, A. Daniele, E. Fahnestock, F. Duvallet, S. Hemachandra, J. Oh, A. Stentz, N. Roy, and T. Howard, "Language understanding for field and service robots in a priori unknown environments," *Field Robotics*, vol. 2, no. 1, 2022.

[61] Y. Li, L. Chen, K. P. Tee, and Q. Li, "Reinforcement learning control for coordinated manipulation of multi-robots," *Neurocomputing*, vol. 170, pp. 168–175, 2015.

[62] S. Sirouspour and P. Setoodeh, "Multi-operator/multi-robot teleoperation: an adaptive nonlinear control approach," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 1576–1581.

[63] Z. Xian, P. Lertkultanon, and Q.-C. Pham, "Closed-chain manipulation of large objects by multi-arm robotic systems," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1832–1839, 2017.

[64] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal,

P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation—A survey," *Robotics and Autonomous systems*, vol. 60, no. 10, pp. 1340–1353, 2012.

[65] V. Andriyanov, "Runtime editor," 2019, accessed on Aug 9, 2021. [Online]. Available: https://assetstore.unity.com/packages/tools/modeling/runtime-editor-64806.

[66] P. Hintjens, *ZeroMQ: Messaging for many applications*. O'Reilly Media, Inc., 2013.

[67] C. Bormann and P. Hoffman, "Rfc 8949 concise binary object representation (cbor)," *Internet Engineering Task Force (IETF)*, 2020.

[68] A. Grönholm, "cbor2," 2016, accessed on Aug 27, 2021. [Online]. Available: https://pypi.org/project/cbor2/.

APPENDIX

A. User Interface

SHARC's end-user interfaces (Figure 8) were created with Unity (Unity Technologies; San Francisco, CA) for nontechnical end-users. Table IV lists the features of these interfaces. The desktop interface was built using the Runtime Editor Unity package [65] and supports cross-platform operation, which was tested on GNU/Linux, macOS, and Windows machines. The VR interface only supports Windows and was developed and tested with an Oculus Quest 2 (Meta; Menlo Park, CA). No software development environment was needed for users to operate either of these interfaces. In Unity, the Oculus Integration package with the Interaction SDK was used to interface with the VR headset, enable the AR/passthrough mode, and implement hand tracking and gesture recognition. Implemented gestures are listed in Table V. The reachable workspace visualization volume was computed based on an approximation of the arm's actual reach. Since the desired end-effector orientation changes the reachable workspace, the arm's reachable volume was computed with the gripper facing directly downwards (this orientation is typically used to collect samples and pick up tools). The interface is provided with the geometric tool and manipulator models in advance, and renders them in context of the 3D reconstruction given tool pose updates and joint angle feedback. The interfaces also display the live video feeds from the cameras. Users can employ these interfaces to send recordings of natural language speech, tool pickup/return requests, pose requests, and gripper open/close requests to the shore server.

TABLE IV
SHARC INTERFACE FEATURES

Feature	Desktop	VR (Field Demo)	VR (User Study)
Live Data Visualizations			
1) 3D workspace reconstruction		✓	✓
2) Live video feeds	✓	✓	✓
3) Tool detection	✓	✓	✓
4) Robot trajectory visualization	✓	✓	✓
5) Manipulator position feedback	✓	✓	✓
6) Inbound data monitor	✓	✓	
End-User Request Capabilities			
7) Speech interface	✓		
8) Automated tool pickup	✓	✓	✓
9) Automated tool return			✓
10) Manipulator pose request	✓	✓	✓
11) Gripper control	✓	✓	✓
Other Interface Features			
12) Reachable workspace visualization		✓	✓
13) Hand tracking			✓
14) Gesture-based controls			✓
15) AR/passthrough mode			✓

TABLE V
SHARC INTERFACE GESTURE-BASED CONTROLS

Gesture	Action
Point palm	Move cursor for UI interactions
Point palm & Pinch	Click UI buttons
Pinch & drag	Move end-effector cursor
Thumbs-up	Confirm request to move arm
Open palm upwards	Open UI menu
Closed fist	Close UI menu

B. Field and Shore Servers

An overview of SHARC's back-end implementation is illustrated in Figure 9. The shore server runs a user authentication script and a data management script that use the high-performance messaging library ZeroMQ (ZMQ) [66] to receive end-user input and distribute data from the field server to all online end-user interfaces. On the shore server, the authentication script checks each incoming request from users (e.g., natural language commands, and tool, pose, and gripper requests) against known credentials. This script forwards planning preview requests from all authenticated users to the field server, but manipulator movement requests are only forwarded if they come from the current designated science operator.

The field server runs a ROS/ZMQ Bridge node, which uses ROS1 to exchange data with the vehicle and the various components of the ROV autonomy framework (e.g., the trajectory planner). This node uses ZMQ to exchange data across a satellite connection to the shore server. To reduce bandwidth usage and delay on the satellite communication link between the field and shore servers, we convert ROS messages to Concise Binary Object Representation (CBOR) objects [67] using the `cbor2` Python library [68].

In our implementation during the field trials, the field server and ROV autonomy nodes were run on a computer onboard the ship. However, this is not a strict requirement for SHARC, as these processes can also be run onboard a vehicle computer. Although onboard integration introduces additional bandwidth limitations and latency due to the communication delay between the robot and the field server, our user study results suggest that operations should be feasible if the effective update delay is $\sim$10 seconds.

C. Autonomy Framework

The ROV autonomy, 3D reconstruction, MoveIt trajectory planning, AprilTag tool detection, and hardware (i.e., manipulator and camera) driver nodes are largely derived from Billings et al. [38]. The ROV autonomy node takes in end-user requests and detected tool poses as input, and outputs end-effector pose requests for the manipulator arm. For the automated tool pickup and return procedures, the autonomy

This article has been accepted for publication in IEEE Transactions on Robotics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TRO.2024.3431830

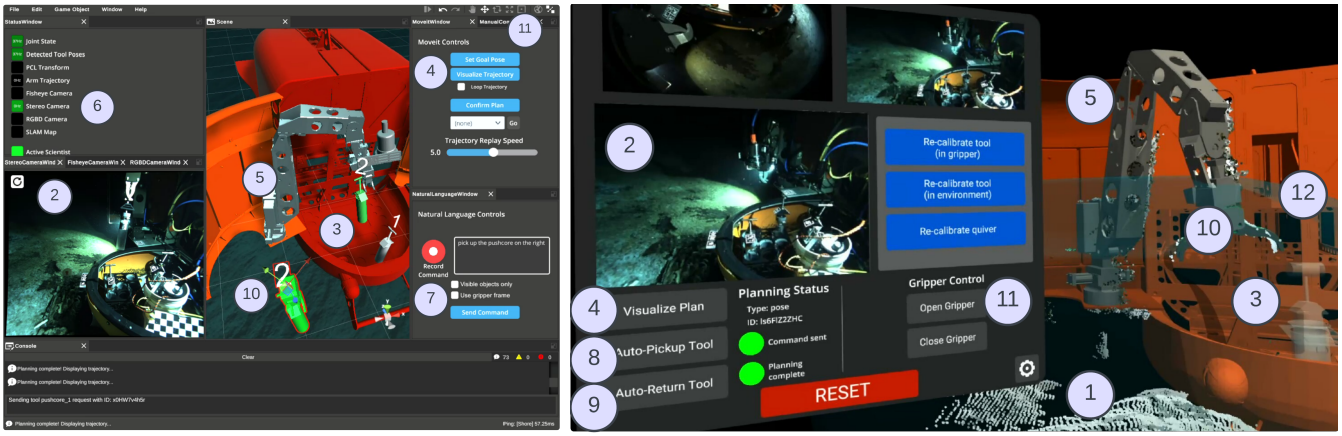

Fig. 8. SHARC (left) desktop and (right) VR interfaces. Annotated features are listed in Table IV.

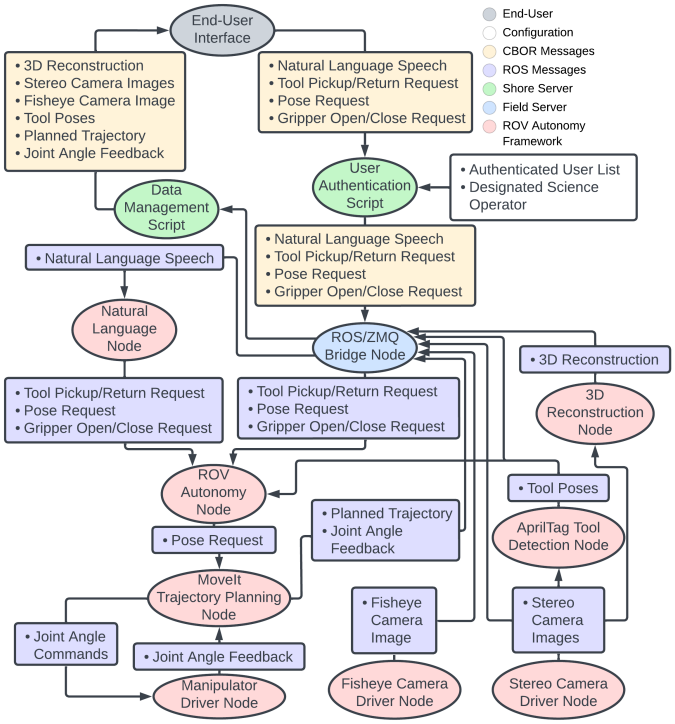

Fig. 9. SHARC software architecture block diagram.

node sends a series of pose requests to the MoveIt trajectory planning node based on the current location of the arm and tools. For example, to return a tool, this node first sends a pose request to move the tool up and away from the workspace, then sends a pose request to align the tool with the detected quiver, and finally sends a pose request to place the tool in the quiver. The 3D reconstruction node matches detected features in the workspace across each of the stereo camera images to reconstruct the workspace. The AprilTag tool detection node computes tool poses based on the known position of the tags relative to the tool. The hardware driver nodes are a mix of open- and closed-source, hardware-specific packages used for the interfacing with the cameras and manipulator arm.

D. Natural Language Interface

SHARC's implementation of the natural language speech interface is also largely derived from Billings et al. [38]. Given a natural language utterance, the interface first uses an off-the-shelf recognition module to convert the speech to free-form text. It then instantiates a probabilistic graphical model in the form of a Distributed Correspondence Graph (DCG) [43, 44] that relates words or phrases from the natural language text to their corresponding referents in a symbolic representation of the environment (e.g., tools) and action space (e.g., picking up or stowing a tool). We train this model on 308 additional examples of natural language commands paired with their corresponding groundings (Table VI).

TABLE VI
EXAMPLE NATURAL LANGUAGE COMMANDS WITH THE ASSOCIATED ACTION GROUNDING TYPES USED TO TRAIN DCG MODEL

Natural Language Command	Grounded Action Type
"Switch to view four" "Show the fisheye camera" "Show the stereo stream" "Show birds-eye viewpoint"	Adjust UI visualization
"Shift the gripper to port" "Move the arm to starboard" "Move up the arm" "Pitch the arm down"	Adjust end-effector position
"Pick up the pushcore in the tray" "Pick up the XRF in the basket" "Grasp the pushcore on the left of the XRF" "Replace the pushcore in the quiver"	Plan and execute Pick & Place action

Since the DCG model can generalize beyond the specific examples present in the training data, this additional data significantly broadens the original implementation's capabilities to include a wider variety of tools, actions, and identifying descriptors. While we use English for our experiments, SHARC supports any natural language as long as a speech-to-text module and a syntactic parser are available.

One of the challenges we faced while integrating off-the-shelf speech-to-text modules is their tendency to prefer commonly used words compared to more uncommon technical terms. For example, while the phrase "push core" is part of the module's vocabulary, the words "push cart" and "push car" were often generated instead. This could be mitigated by using a system that can be fine-tuned to the application's context.