2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Interactive Planning and Supervised Execution for High-Risk, High-Latency Teleoperation

Will Pryor, Balazs P. Vagvolgyi, Anton Deguet, Simon Leonard, Louis L. Whitcomb, and Peter Kazanzides

***Abstract*— Ground-based teleoperation of robot manipulators for on-orbit servicing of spacecraft represents an example of high-payoff, high-risk operations that are challenging to perform due to high latency communications, with telemetry time delays of several seconds. In these scenarios, confidence of operating without failure is paramount. We report the development of an Interactive Planning and Supervised Execution (IPSE) system that takes advantage of accurate 3D reconstruction of the remote environment to enable operators to plan motions in the virtual world, evaluate and adjust the plan, and then supervise execution with the ability to pause and return to the planning environment at any time. We report the results of an experimental evaluation of a representative on-orbit telerobotic servicing task from NASA's upcoming OSAM-1 mission to refuel a satellite in low earth orbit; specifically, to change the robot tool to acquire the fuel supply line and then to insert it into the satellite fill/drain valve. Results of a pilot study show that the operators preferred, and were more successful with, the IPSE system when compared to a conventional teleoperation implementation.**

I. Introduction

Ground-based teleoperation of robot manipulators for on-orbit servicing of spacecraft represents an example of high-payoff, high-risk operations that are challenging to perform due to high latency communications, with telemetry time delays of several seconds. In these operations, it is often difficult to recover from failure. For example, on Earth a dropped or broken tool may be retrieved or replaced, but in space a "dropped" tool may float away and a broken tool cannot easily be repaired or replaced. Small failures can cause the entire mission to fail, wasting many years and many millions of dollars. The situation is compounded by the inherent challenges of remote teleoperation, which include large communication delays and limited visualization of the remote environment. In particular, we focus on ground-based control of a robot in low earth orbit (LEO), which is subject to time delays of several seconds. This work is motivated by NASA's On-Orbit Servicing Assembly and Maintenance 1 (OSAM-1) mission (formerly called Restore-L) to demonstrate telerobotic refueling of a satellite in LEO [1], which is further described in Section II.

Based on observations of NASA flight operations, discussions with NASA robot operators, and prior experiments with these operators [2], we concluded that keyboard-based robot control was generally preferred over joystick control for high-risk operations over time-delayed telemetry, especially those in proximity of the satellite. This is partly due to the multi-second time delay, which leads operators to adopt a "move and wait" strategy that is well suited to a keyboard interface. The keyboard interface also enables operators to more precisely command motions. On the other hand, the keyboard interface is cumbersome for large motions, such as when moving back and forth to the tool stowage area. Although a joystick would make the task easier to perform, it is generally avoided due to the higher potential for commanding erroneous or imprecise motion. This scenario motivates the design of an Interactive Planning and Supervised Execution (IPSE) system (Fig. 1) that can facilitate both coarse and fine motions without increasing the risk of failure.

Johns Hopkins University, Baltimore, MD, USA,
{willpryor, balazs, sleonard, llw, pkaz}@jhu.edu

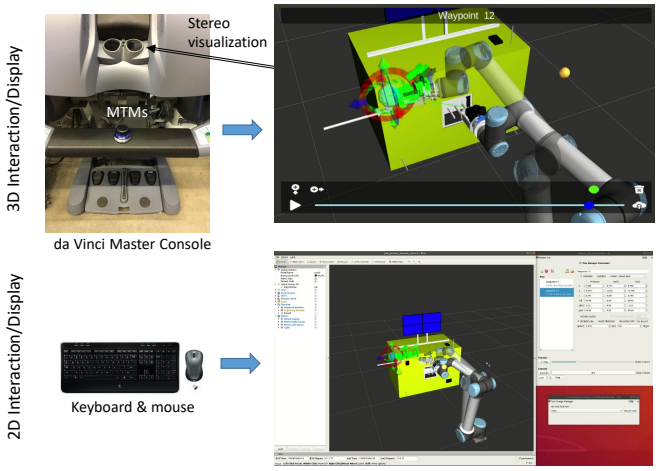

Fig. 1. Overview of Interactive Planning and Supervised Execution (IPSE) system, showing 3D interface (top) and 2D interface (bottom). The da Vinci Master Console includes two Master Tool Manipulators (MTMs) that enable 3D interaction (for example, to move the interactive cursors) and provides a stereo display for visualization. Both 3D and 2D interfaces are available simultaneously and visualize/update the same scene.

This paper is organized as follows: Section II presents background information about the motivating application in satellite servicing, followed by a review of related work in visual robot programming and interactive planning. Section III describes the system implementation and Sections IV and V present the experiments and results, respectively, of a ground-based evaluation of the IPSE system compared to the conventional keyboard/GUI interface and joystick.

II. Background and Motivation

The Exploration and In-Space Services (ExIS) Division at the NASA Goddard Space Flight Center (GSFC) is developing, evaluating, and demonstrating new technologies for telerobotic servicing of satellites on-orbit. ExIS is developing the OSAM-1 mission [1] to launch a servicing spacecraft equipped with two robotic arms and a satellite capture

mechanism for docking with the target satellite, Landsat 7, in low earth orbit (LEO). The rendezvous and capture procedures will be performed autonomously due to the high relative velocities, which precludes operator control, but NASA plans to use ground-based telerobotic control for the subsequent proximity operations, including refueling Landsat 7. These operations include cutting the multilayer insulation (MLI) that encases the satellite, cutting retaining wires, removing the cap on the fill/drain valve, attaching the refueling hose, transferring fuel, and then reinstalling the cap and applying an MLI patch. ExIS has designed several specialized robot tools for these operations, which necessitates navigation of the robot arm between the worksite on the target satellite and the tool stowage area on the servicing spacecraft, while avoiding collisions between the robot arm and structures on either spacecraft. Thus, the overall procedure requires both coarse motion to transit to/from the tool stowage area and fine motion to perform proximity operations.

The design of the IPSE system was primarily motivated by our previous study [2], in which we recruited five trained NASA robot operators to evaluate several teleoperation interfaces for the task of cutting the MLI "hat" that covers the satellite fill/drain valve. Specifically, we compared our recently-developed augmented virtuality (AV) visualization [3] to the conventional camera visualization and the da Vinci master console to the conventional keyboard/GUI control interface. The results showed that the NASA operators unanimously preferred the AV visualization, but only when paired with the conventional keyboard/GUI control interface. The da Vinci control interface was the least preferred; in fact, operators were willing to sacrifice the AV visualization in order to retain the keyboard/GUI interface, even though they consistently performed the task faster with the da Vinci.

Post-experiment interviews with the NASA operators revealed two primary reasons for their preferences. First, the task did not require the dexterity offered by the da Vinci master console, since the MLI cutting operation primarily involved moving in a straight line. They suggested that other operations, such as large motions between the worksite and the tool stowage area, would be likely to benefit from the ability to command more complex motions. Second, operators indicated that performing the task without error is more important than performing it quickly. Although task failures could be easily remedied in our ground-based setup, the NASA operators followed their training to avoid failure. The keyboard/GUI interface provided a simple method to preview a motion before executing it; specifically, a yellow ring was overlayed on the camera images to show the position the cutter would reach if the operator pressed the "Move" button. In contrast, the da Vinci interface only allowed the operator to move the robot or to adjust the virtual camera (for the AV visualization) – there was no ability to preview motions. This revelation led to the development of the IPSE capability, where the operator first plans in a virtual environment, previews the plan, makes adjustments if necessary, and then supervises execution of the motion with the ability to pause and replan/resume at any time. We implemented both 2D (keyboard/GUI) and 3D (da Vinci) interfaces.

The IPSE system is demonstrated in a refueling task that is representative of on-orbit telerobotic servicing of spacecraft. We assume that the robot has just removed the cap on the fill/drain valve and must first move to the tool stowage area to acquire the fill tool, then move to grasp the hose and mate it with the fill/drain valve.

The IPSE approach resembles offline robot programming, which has been used for decades [4]. However, offline programming is unable to respond to geometry which is unknown a priori, or to geometry which may change unpredictably during the operation. Another strategy that has been used to mitigate the effects of time delay is to have operators command the robot using high-level task goals, which generalize to variations in the environment, and the remote robot determines the exact details of the operation autonomously [5]. This relies on the autonomous system's ability to execute a high-level goal without error, which creates unnecessary risks with high penalties in a satellite servicing environment. Likewise, systems which predict the response of the remote robot and environment to simulate real-time feedback, such as in [6], [7], are reliant on accurate simulation of the robot and environment. Other approaches that rely on simulation include teleprogramming [8] and telesensor-programming [9], where the operator interacts with a simulated remote environment and teleprograms the remote robot through a sequence of elementary motion commands. Similarly, in [10] the simulated environment is used only for visualization, and inaccuracies are mitigated by adjusting the commanded path based on sensor feedback on the remote system. These strategies can improve execution safety but preclude human oversight on the final executed path.

More recent related work involves the use of mixed/augmented reality for visual programming of robot motions. In [11], a user can plan paths as a series of waypoints in an augmented reality (AR) environment, preview and edit the paths, and then execute them either autonomously or by allowing the user to control progress through the path. In [12], the user similarly builds a path out of primitives, visualizes the final path, and then executes it. The authors also evaluated the mixed reality interface against a 2D baseline interface among a non-expert population, finding that the mixed reality interface was preferred. Other studies have investigated interfaces that assist the user in selecting pose goals, which can be used to specify waypoints in such plans [13], [14]. Both studies found that, in general, users were more comfortable and more successful with interfaces that exposed fewer degrees of freedom and provided more automated assistance, with [13] additionally finding that the best-performing strategy varied by environment. Similarly, another study demonstrated that reduced-DOF interfaces performed better than full-DOF control during live teleoperation [15]. In our proposed system, we provide control in a task-aligned coordinate system, with the goal of allowing expert users to select an intuitive reduced-DOF subset of inputs while allowing full control when needed.

III. System Description

An operation using our proposed system begins with a registration and reconstruction procedure, to locate and build the objects in the virtual (simulated) environment. The operator then creates a motion plan using the interactive planning capability and executes the plan with supervised execution. The latter two steps are repeated until the task is complete. The system is implemented with a set of programs using the Robot Operating System (ROS) [16].

A. Registration and Reconstruction

The operator of our system views and manipulates the work-site via a computer simulation of the scene that contains the relevant objects, robots, and equipment. The simulation is based on design drawings, robot kinematics, camera images, and other sensor data. We did not include dynamics in our simulation because we assume that the scene contains rigid objects that do not collide with each other.

We used our Vision Assistant software [3] to perform object registration and to reconstruct the geometry of unknown but relevant features of the object. The registration and reconstruction process requires the operator to perform a visual survey of the object by taking several images from multiple positions using the tool camera mounted on the robot's end effector. Visual landmarks that match reliably known features in the object's CAD model are used for registration, while landmarks that are different from the CAD model are reconstructed using 3D triangulation and added to the model.

The result of the registration and reconstruction process is a 3D simulation of the scene that accurately represents the physical location of the relevant objects, robots, and equipment in the remote work-site.

B. Interactive Planning and Supervised Execution (IPSE)

The IPSE component allows the operator to design a motion plan using one of multiple co-existent user interfaces. At any time, the operator can preview the resulting robot motion and edit the plan until the motion is satisfactory.

A motion plan consists of a series of waypoints, as shown on the left in Fig. 2. Each waypoint represents an intermediate destination in the motion plan. A motion planning engine, using the MoveIt planning framework [17], plans a trajectory to connect each waypoint's destination pose with the final configuration of the previous waypoint's trajectory, with the first waypoint connected to the robot's current configuration. The resulting trajectories are collision-free when possible and marked as invalid when a collision cannot be avoided. Invalid trajectories cause execution to be disabled and the waypoint marker (Fig. 3, see Sec. III-B.1) to be displayed in red.

The operator may configure each waypoint to use a straight-line path, which causes the end effector to follow a straight line in task space; to avoid obstacles, in which case the motion planner may select any collision-free path; or to follow the same task-space path that the operator followed to move the waypoint marker. Each waypoint also has an independent set of desired speeds, both linear and rotational. The trajectory

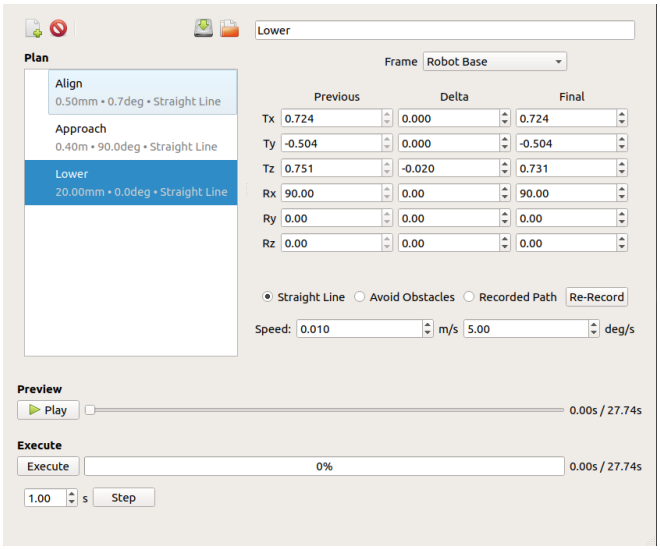

Fig. 2. The custom user interface for the 2D planning interface allows the operator to view and edit the waypoints as delta (relative to the previous waypoint) and final (relative to the fixed frame) values.

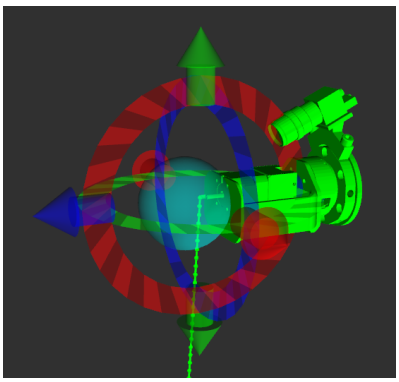

Fig. 3. A waypoint marker displays as a colored end-effector with additional controls to enable moving it in all six degrees of freedom. A dotted line shows the path the end effector will take to reach this waypoint.

is initially timed to obey pre-configured joint-level limits and then the duration of each trajectory segment is scaled up if necessary to ensure that the segment obeys both the linear and rotational end-effector speed limits.

The planner is independent from any user interface. It communicates with any number of interfaces simultaneously using a ROS topic API, and changes made in any interface are immediately reflected in all connected interfaces. We implemented two user interfaces, as shown in Fig. 1 and described below:

1) Interactive Planning: 2D Interface: The 2D interface uses a custom configuration of RViz, the standard robot visualizer bundled with ROS, in combination with a custom Qt interface called the Planner GUI (Fig. 2). This GUI displays the list of waypoints in the current plan and allows the operator to edit the selected waypoint. In addition to naming each waypoint, the operator can input the goal pose as either a "delta" from the start pose or as a "final" absolute pose, and modifying either column will update the other. The operator can choose the reference frame in which to display these

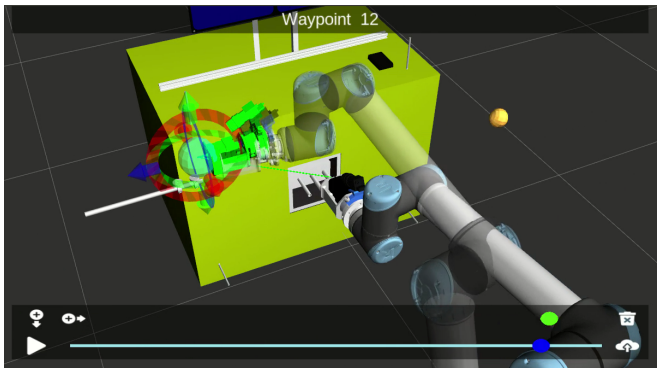

Fig. 4. The 3D Interface displays the virtual world in 3D, with a GUI overlay to replicate the most common features of the Planner GUI.

poses from a list, which can include any frame whose pose is known and static during an entire planning operation. If the pose of an object of interest is known, the operator can choose that frame and align each degree of freedom independently by entering 0 in the appropriate input of the Final column.

As waypoints are added and edited, each one is represented in RViz as a colored end effector, as shown in Fig. 3. The end effector is green when the plan is valid, red when it is invalid, and white when the plan has not been computed. The latter case occurs briefly whenever an edit is made as well as whenever a previous waypoint was invalid, as each waypoint's plan depends on the previous. Clicking each waypoint marker selects the corresponding waypoint. The selected end effector is surrounded by interactive markers which allow the waypoint to be translated along each axis (arrows), rotated about each axis (rings), or translated in the plane perpendicular to the virtual camera (sphere). When a trajectory has been computed between each pair of waypoints, it is shown as a dotted line tracing the path of the end effector tip.

The Planner GUI also allows for choosing between path types and setting the maximum speeds. Below the waypoint edit areas, the Play button and scrubber allow the operator to preview the current plan, either by playing it back in real-time or scrubbing through as in a typical media player. The "preview robot", a partially-transparent version of the actual robot, animates through the planned path (Fig. 1). The preview and execute functions use the same stored trajectory, so the operator can be assured that the execution will follow the exact same configurations as the preview. The GUI also includes the Supervised Execution controls (Sec. III-C).

2) Interactive Planning: 3D Interface: The planning system can also be controlled with a 3D interface, shown in Fig. 4, which uses the da Vinci master tool manipulators (MTMs) to control the same path. Each of these two MTMs is a 7 DOF compliant robot arm which is used to track the operator's hands in space and also includes a gripper which may be pinched with the user's thumb and forefinger to command actuation of some tool. In our system, each of the MTMs controls a 3D cursor, represented in the virtual world as a sphere. Pinching the MTM's gripper acts as a click. The operator looks into the da Vinci stereo viewer at the same virtual world as shown in RViz, here rendered in 3D. The 3D cursor can be used as a mouse to select and move waypoints in the same way as with the 2D interface, except that with a 6-DOF input device the sphere can be used to position and orient the waypoint anywhere in the workspace. The addition of 3D vision and 6-DOF manipulation greatly enhances the operator's ability to position the waypoints, which is especially important when recording paths.

The 3D interface includes an overlay GUI to give operators access to the most-used features of the Planner GUI without moving from the da Vinci console. When the 3D cursor moves behind one of the buttons visible at the bottom of Fig. 4, it transforms into a 2D cursor of the same color and allows clicking the buttons, which provides the ability to add a waypoint at the end of the plan, convert the current preview location to a new waypoint, delete the current waypoint, and preview and execute the plan. The 2D cursor also manipulates the preview scrubber, as illustrated by the lower interactive cursor in Fig. 1.

C. Supervised Execution

When the operator is satisfied with the planned trajectory, they may execute it on the remote robot. The Execute button in the Planner GUI (Fig. 2) transmits the current trajectory to the remote robot immediately, but due to the communication delay, the robot does not appear to move until one round-trip time (RTT). Once feedback is available, the progress bar fills to indicate progress through the planned trajectory and the opaque robot shown in the virtual world follows the robot telemetry feedback.

During execution, the operator may watch the robot's progress in an Augmented Virtuality (AV) visualization environment (Fig. 5), described in [2]. This visualization displays the robot and environment as in RViz, but also augments the virtual models with a projection of the image from the robot's tool camera. The registration procedure in Sec. III-A is sufficiently accurate that each area of the camera image is projected onto the virtual model of the corresponding object with minimal error. The projection improves the operators' situational awareness and ability to judge the completion of the task by transforming the 2D image into 3D textured objects; furthermore, it helps operators recognize inaccuracies in the simulation which informs them when more caution is required. In principle, this visualization could be unified with the visualization used during planning, but currently they are separate due to practical implementation concerns.

If the operator judges that the trajectory is inaccurate, they may pause the trajectory from the Planner GUI. However, due to the time delay, the robot will continue moving 1 RTT past its apparent position before the pause command is received. For cases where this delay could damage the mission, the operator can instead use the Step functionality to execute only a small portion of the trajectory at a time, allowing the operator to reassess the trajectory's success between each step. The step function truncates the trajectory to the specified time and sends only the truncated portion to the robot. The robot's progress through the trajectory is retained through

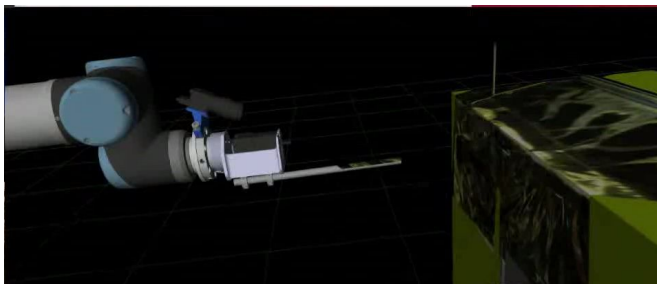

Fig. 5. Augmented Virtuality (AV): the tool camera image is projected onto the virtual environment (here, both the robot tool and satellite).

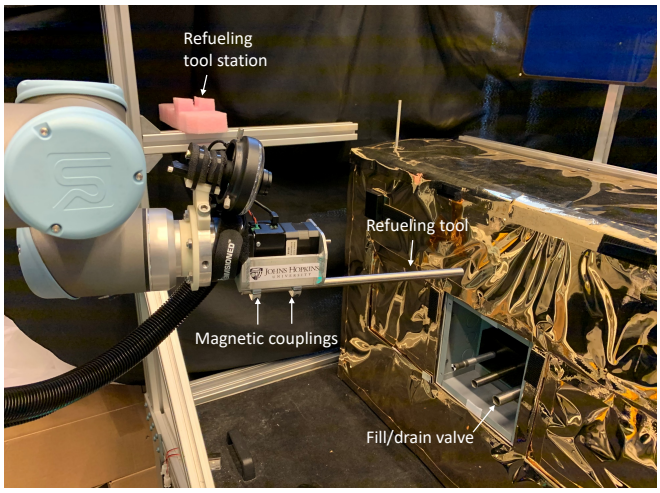

Fig. 6. The space-side setup includes a mock satellite (right), mock servicing robot (front left), and refueling tool station (back left).

multiple uses of the Step and Execute buttons so the operator may Step or Execute until the trajectory is complete.

When the trajectory requires correction, the operator may choose to start a new motion plan, at which point the trajectory progress is reset. If the difference is small, the operator may also choose to use a joystick to adjust the robot's pose, which is also subject to communications delay. The operator can then re-compute the trajectory for the remaining portion of the motion plan using the new start position and resume normal execution.

IV. Experiments

We performed a pilot study to investigate the effect of our proposed system on a representative satellite servicing task.

A. Experimental Setup

The experimental setup consists of a mock satellite, mock servicing robot, and operator station. The mock satellite, shown at the right of Fig. 6, is constructed from 80/20 aluminum bars and panels wrapped in a layer of Mylar [18]. Not including its solar panel, the satellite is a box of size $24 \times 24 \times 36$ inches. A set of three pipes of varying inner diameters (from left to right, 0.504", 0.555" and 0.584") is fixed within a cavity in one side of the satellite; the largest pipe is selected to represent the satellite fill/drain valve. A tool stowage area, visible behind the mock servicing robot in Fig. 6, is attached to the same structure as the mock satellite.

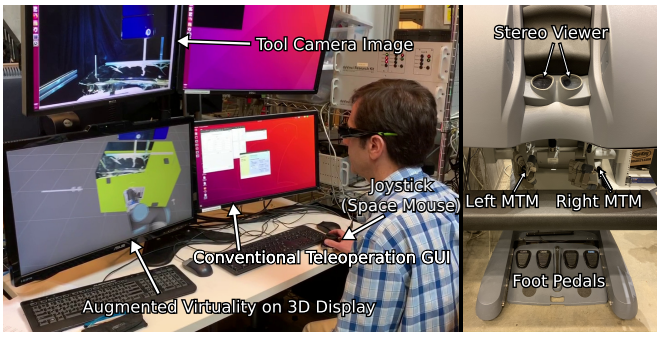

Fig. 7. Teleoperation control interfaces: keyboard/GUI with 2D interface (left) and da Vinci master console (right); left image also shows augmented virtuality (AV) visualization on 3D monitor (lower left).

The mock servicing robot, shown at the left of Fig. 6, is a Universal Robots UR-10. A custom tool including a camera, light ring, and tool mounting assembly is mounted to the UR-10. The tool mounting assembly includes a motor to which active tools can be mounted (not used in this experiment) and a set of magnets to which passive tools can be attached. The refueling tool, a section of pipe affixed to magnets, is shown attached to the tool mount. The outer diameter of the refueling tool is 0.500". At the base of the servicing robot, not visible, are two deck cameras with views of the pipe cavity on the mock satellite.

While operating the robot, the operator sits at an operator station that includes several monitors, a keyboard and (2D) mouse, a (6D) joystick (Space Mouse), and a da Vinci master console, shown in Fig. 7. A 3D monitor, used with active shutter 3D glasses, displays the Augmented Virtuality Visualizer (Sec. III-C). A standard 2D monitor displays the Planner GUI and RViz, (Sec. III-B.1), and an additional 2D monitor displays the unaltered tool camera images.

B. Experimental Task

An experiment begins with the mock satellite already registered using the Vision Assistant software described in Sec. III-A. The robot begins in a standard location not near the mock satellite or tool stowage area and the refueling tool begins in the tool stowage area. During the experiment, the operator must first command the robot to the tool stowage area and lower it onto the tool to engage the magnetic attachment. Once at least one magnet attaches, an experimenter records whether the tool was aligned correctly and manually aligns the tool if necessary so that the experiment may continue. The operator must then command the robot to move the tip of the refueling tool inside the pipe on the mock satellite that represents the fill/drain valve. For this pilot study all operators were instructed to use the largest pipe. The experiment is complete when the operator believes the tool tip is inserted at least 3 cm inside the pipe, or failed if the operator believes it is not possible to insert the tool tip into the pipe (e.g., if the tool is knocked off the magnetic mount).

C. Conditions

Each operator performs the experimental task three times, with one trial in each of the following configurations:

1) Conventional: The operator directly moves the robot with a simple GUI or by using the joystick. The GUI allows the operator to enter only a desired pose, expressed relative to the base of the servicing robot, and end effector speed. The only preview capability is provided by a small marker that indicates the commanded pose in the Augmented Virtuality visualization. An *Execute* button commands the robot to move towards the commanded pose in a straight line, and if needed the operator may *Abort* the motion. This interface was designed to resemble an interface currently used for robot control at NASA.

2) IPSE-2D: The operator uses the 2D interface, which includes the joystick, for interactive planning and for supervised execution.

3) IPSE-3D: The operator uses the 3D interface (da Vinci master console) for interactive planning and is only allowed to use the 2D interface for features that are not currently supported by the 3D interface, such as supervised execution.

In all three configurations, the operator looks at the Augmented Virtuality environment on the 3D monitor and the unaltered tool camera image for feedback.

Although the IPSE system was designed to enable the operator to switch between the 2D and 3D interfaces at will, we decided to evaluate them in separate trials for two reasons: (1) to compare their effectiveness in performing the task, and (2) to ensure that each interface was actually used (otherwise an operator could choose to only use one of them).

V. Results

Six operators participated in the user study (JHU HIRB 00000701). Operators were recruited from a population familiar with using the da Vinci surgical system for teleoperation, to reflect the fact that this task would be performed by trained operators. Three of the operators rated themselves as "Experienced" and two as "Familiar" with remote teleoperation systems. All operators performed all three tasks in the same order: Conventional teleoperation, IPSE-2D, then IPSE-3D. Before each trial, the operators were introduced to the system and then allowed to practice the task until they chose to begin the trial.

In a high-risk task such as an on-orbit operation, task success is the most relevant metric. We categorized the results for the two tasks, tool pickup and refueling, into three categories: Full success, partial success, and failure. Partial success was defined as an attached but improperly aligned tool in the tool pickup task, and as a tool inserted less than the desired 3 cm in the refueling task. For both tasks, failure was indicated when the operator believed it was no longer possible to complete the task. We also measured the pose difference between the final tool pose at the end of each task and the nominal tool pose. It should be noted that this nominal tool pose is subject to registration error, and so some small amount of deviation from the nominal pose may indicate a more accurate tool placement. Large deviations, however, indicate misplacement. Position and orientation error are reported as average values over all successful trials.

TABLE I

Pickup Task Success and Pose Error, mean (stdev)

	Conv.	IPSE-2D	IPSE-3D
Full Success	67%	100%	100%
Partial Success	33%	0%	0%
Failure	0%	0%	0%
Position Error, mm	23.2 (12.9)	5.7 (7.3)	9.5 (4.9)
Orientation Error, deg	4.3 (2.4)	1.8 (1.8)	4.2 (2.1)

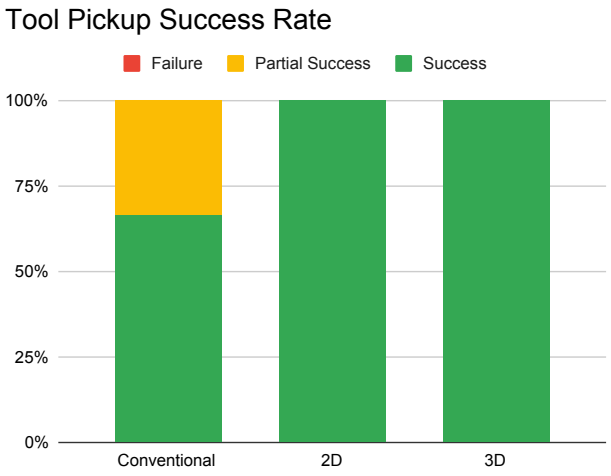

Fig. 8. In the tool pickup task, no operator failed in any condition and both IPSE-2D and IPSE-3D cases saw complete success.

Success metrics for the pickup task are shown in Table I and Figure 8. This task had no failures under any of the three conditions, which we believe reflects the fact that the magnetic mount is sufficiently strong to attach even across a fairly large distance. However, attaching at a distance increases the probability of a misaligned tool, which was evident in the partial success rate. Using conventional teleoperation, two of the operators misaligned the tool. The displacement metrics indicate that operators were able to position the tool much more accurately to the nominal pickup position using IPSE, which may be due to the ability to preview the commanded end effector pose using a model of the end effector itself, rather than a small stand-in indicator. The orientation error is similar between the conventional and IPSE-3D interfaces, but lower with the IPSE-2D interface, which may indicate the influence of being able to place a waypoint with respect to the nominal tool pickup pose using the IPSE-2D interface (i.e., by setting the `Frame` in Fig. 2 to the tool pickup pose).

For the refueling task, successful execution necessarily

TABLE II

Refueling Task Success and Orientation Error, mean (stdev)

	Conv.	IPSE-2D	IPSE-3D
Full Success	50%	83%	50%
Partial Success	17%	17%	0%
Failure	33%	0%	50%
Tool Misalignment	50%	17%	33%
Orientation Error, deg	2.6 (0.9)	1.9 (1.4)	1.7 (0.7)

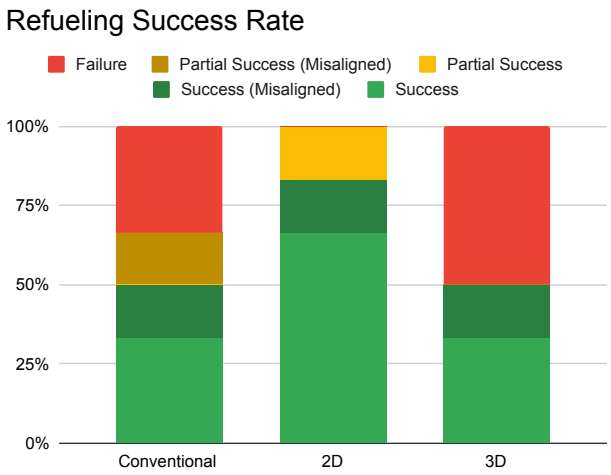

Fig. 9. In the refueling task, only the IPSE-2D interface saw zero failures. In this task, it was possible for operators to misalign the tool and still complete the task, shown here in darker color.

constrains the refueling tool to be within a pipe of only slightly larger diameter (a tolerance of 2.13 mm), and so position deviations are all within the margin of registration error. However, this task affords the opportunity to dislodge the magnetically attached tool without knocking it off entirely, and the rate of such misalignment is also reported in Table II. Tool misalignments are reported as the percentage of (fully or partially) *successful* tasks for which the tool was misaligned.

Success rates for the refueling task (Table II and Figure 9) were much lower, demonstrating the significantly higher requirement for precision in this task. Of the five failures across all conditions, four were due to the operator knocking the tool off the mount by contacting an obstacle in the environment. Of these, three were caused by contacting the refueling tool holder, which was visible when the operators were introduced to the task but was not modeled in the virtual environment. In addition, two of the four instances of operators knocking the tool out of alignment occurred on the same geometry. For this reason, the promise of the virtual environment and/or collision detection may have been detrimental to overall performance because operators expected that every collision would be visible in the virtual environment or detected by the IPSE system.

In another example of operators failing the task due to placing trust in the virtual environment, the operator who achieved a partial success in the refueling task using the IPSE-2D interface had correctly entered the command to insert the tool by exactly 3 cm. However, due to a 1.4 mm registration error, the refueling tool only entered 2.86 cm into the fuel pipe. This failure mode was not readily available in the IPSE-3D interface because it did not provide the capability to enter numerical motion commands.

The combined success rate for the IPSE-3D interface (50%) was the lowest of the three conditions, followed by the conventional interface (67%). Only the IPSE-2D interface had a 0% failure rate, and it also had the highest full success rate of the three. While the IPSE-2D interface improved task performance, the IPSE-3D interface led to worse results than the conventional interface. Multiple operators reported that they did not use the camera view, or used it much less, while using IPSE-3D due to the effort of switching between the 3D planning and 2D execution interfaces. Other operators reported that the preview feature was more difficult to activate using the IPSE-3D interface, leading them to use it less. It was also noted that the IPSE-3D interface did not offer the advantage of specifying precise movements with respect to a defined frame in the environment, such as the tool mount frame or the refueling pipe frame, which are known as a result of the 3D scene model constructed from the camera survey. It is likely that a fully-featured IPSE-3D interface, which does not require exiting to the 2D interface to access some functionality, would produce different results.

Orientation error appears to once again be lower using IPSE, although in this task the IPSE-3D interface had the lowest average error. However, since these numbers are the average of only trials that were completed successfully with no misalignment, the conventional and IPSE-3D cases are each derived from only two data points. The IPSE-2D case, which is derived from five data points, shows a similar alignment error to the pickup task.

The workload for each interface was measured using a version of the NASA Task Load Index (TLX) which reports values on a scale from 1 to 7, with 1 representing the lowest possible workload. The results correlate with the performance measures: The conventional interface was rated an average of 3.7 (standard deviation 0.82), the IPSE-2D interface imposed the lowest workload with 2.2 (0.71), and the IPSE-3D interface was significantly more difficult, with an average rating of 4.5 (1.41). We also asked the participants to rate the difficulty on a scale of 1 to 5 (where higher numbers indicate greater difficulty) with similar results: 3.5 (0.55), 1.7 (0.82), and 4.5 (0.55), respectively.

TABLE III

TASK DURATION, MINUTES:SECONDS, MEAN (STDEV)

	Conv.		IPSE-2D		IPSE-3D	
Tool Pickup	11:31	(4:17)	6:34	(4:01)	10:59	(4:38)
Refueling	17:35	(7:44)	12:18	(5:27)	30:22	(4:01)

Although execution time is significantly less important than success rate, we also recorded the average time to successful completion of each task, where applicable, which is shown in Table III. The results for the conventional and IPSE-2D cases show that the IPSE-2D interface allowed operators to complete the task faster, which we attribute to the lower difficulty and the operators' increased confidence in their ability to safely execute longer motions. The results for the IPSE-3D interface, however, indicate that in the less-constrained tool pickup it was comparable to the conventional interface, but in the severely constrained refueling task it required much longer execution times than the other interfaces. At least some of this time difference was likely due to the need to frequently switch between the 3D planning and 2D

execution, as well as the difficulty of using features such as the path preview with the da Vinci manipulators.

VI. Discussion and Conclusions

We developed an Interactive Planning and Supervised Execution (IPSE) teleoperation system to allow operators to accurately command satellite servicing robots from the ground in the presence of multi-second communication delays and non-ideal camera views. The Interactive Planning component allows operators to define a motion plan as a series of waypoints and the Supervised Execution component enables them to observe the plan's execution and respond if failure is imminent. We implemented two interfaces to IPSE, one using RViz and a custom GUI in 2D, and one using the da Vinci to allow planning in 3D.

The system was evaluated with six operators, most of whom rated themselves as Experienced or Familiar with teleoperation systems. We found that operators were more successful with the IPSE-2D system than with a conventional teleoperation interface, and rated the 2D interface as easier to use. We also found that the IPSE-3D interface was, in most measures, equal to or worse than the conventional interface in both success metrics and difficulty. Some operators indicated specific limitations of the IPSE-3D interface that influenced their success, such as having to leave the da Vinci console to view camera feedback or the inability to specify precise motions with respect to certain defined task frames. We hypothesize that the IPSE-3D interface may have been effective for some parts of the task, but not for the entire task. Future work will include improvements to the IPSE-3D interface as well as experiments that allow the operator to freely switch between the 2D and 3D interfaces.

Acknowledgments

This work is supported by NASA NNG15CR66C. We thank Brian Roberts, Billy Gallagher and the ExIS team at NASA GSFC. Adarsh Malapaka assisted with the experimental setup and Srishti Dhamija assisted with the implementation of the supervised execution capability. The da Vinci Research Kit (dVRK) is supported by NSF NRI-1637789.

References

[1] NASA GSFC, "On-Orbit Servicing, Assembly and Manufacturing 1 Mission." [Online]. Available: https://nexis.gsfc.nasa.gov/OSAM-1.html

[2] W. Pryor, B. P. Vagvolgyi, W. J. Gallagher, A. Deguet, S. Leonard, L. L. Whitcomb, and P. Kazanzides, "Experimental evaluation of teleoperation interfaces for cutting of satellite insulation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2019, pp. 4775–4781.

[3] B. Vagvolgyi, W. Pryor, R. Reedy, W. Niu, A. Deguet, L. L. Whitcomb, S. Leonard, and P. Kazanzides, "Scene modeling and augmented virtuality interface for telerobotic satellite servicing," *IEEE Robotics & Automation Letters*, vol. 3, no. 4, pp. 4241–4248, Oct. 2018.

[4] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," in *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*. VDE, 2010, pp. 1–8.

[5] M. D. Johnston and K. J. Rabe, "Integrated planning for telepresence with time delays," in *IEEE Intl. Conf. on Space Mission Challenges for Information Technology (SMC-IT'06)*, July 2006.

[6] A. K. Bejczy, W. S. Kim, and S. C. Venema, "The phantom robot: predictive displays for teleoperation with time delay," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 1990, pp. 546–551.

[7] Y. Tsumaki and M. Uchiyama, "Predictive display of virtual beam for space teleoperation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Nov 1996, pp. 1544–1549.

[8] J. Funda, T. S. Lindsay, and R. P. Paul, "Teleprogramming: Toward delay-invariant remote manipulation," *Presence: Teleoperators & Virtual Environments*, vol. 1, no. 1, pp. 29–44, 1992.

[9] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl, "Sensor-based space robotics-ROTEX and its telerobotic features," *IEEE Trans. on Robotics and Automation*, vol. 9, no. 5, pp. 649–663, 1993.

[10] M. Oda, N. Inaba, Y. Takano, S. Nishida, M. Kayashi, and Y. Sugano, "Onboard local compensation on ETS-VII space robot teleoperation," in *IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics*, 1999, pp. 701–706.

[11] C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. M. Van der Loos, and E. Croft, "Robot programming through augmented trajectories in augmented reality," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 1838–1844.

[12] S. Y. Gadre, E. Rosen, G. Chien, E. Phillips, S. Tellex, and G. Konidaris, "End-user robot programming using mixed reality," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2019, pp. 2707–2713.

[13] A. E. Leeper, K. Hsiao, M. Ciocarlie, L. Takayama, and D. Gossow, "Strategies for human-in-the-loop robotic grasping," in *ACM/IEEE Intl. Conf. on Human-Robot Interaction (HRI)*, NY, NY, USA, 2012, p. 1–8.

[14] D. Kent, C. Saldanha, and S. Chernova, "A comparison of remote robot teleoperation interfaces for general object manipulation," in *ACM/IEEE Intl. Conf. on Human-Robot Interaction (HRI)*, New York, NY, USA, 2017, p. 371–379.

[15] C. E. Mower, W. Merkt, A. Davies, and S. Vijayakumar, "Comparing alternate modes of teleoperation for constrained tasks," in *IEEE Intl. Conf. on Automation Science and Eng. (CASE)*, 2019, pp. 1497–1504.

[16] Open Robotics, "Robot Operating System." [Online]. Available: https://www.ros.org

[17] I. A. Sucan and S. Chitta, "MoveIt." [Online]. Available: http://moveit.ros.org

[18] B. Vagvolgyi, W. Niu, Z. Chen, P. Wilkening, and P. Kazanzides, "Augmented virtuality for model-based teleoperation," in *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, Sep 2017.