A Framework for Customizable Multi-User Teleoperated Control

Adnan Munawar, *Member, IEEE*, Jie Ying Wu, *Graduate Student Member, IEEE*, Russell H. Taylor, *Life Fellow, IEEE*, Peter Kazanzides, *Member, IEEE*, and Gregory S. Fischer, *Member, IEEE*

***Abstract*—Traditional teleoperation (leader/follower) systems primarily focus on one operator controlling one remote robot, but as robots become ubiquitous, there is an increasing need for multiple operators, including autonomous agents, to collaboratively control multiple robots. However, existing teleoperation frameworks do not inherently support the variety of possible collaborations, such as multiple operators, each with an input device (leader), controlling a robot and camera or different degrees of freedom of a single robot (follower). The same concept applies to teleoperating robots in a simulation environment through physical input devices. In this letter, we extend our novel simulation framework that is capable of incorporating multiple input devices asynchronously with a real-time dynamic simulation to incorporate a customizable shared control. For this purpose, we have identified and implemented a sufficient set of coordinate frames to encapsulate the pairing of multiple leaders, followers and cameras in a shared asynchronous manner with force feedback. We demonstrate the utility of this framework in accelerating user training, ease of learning, and enhanced task completion times through shared control by a supervisor.**

***Index Terms*—Medical robots and systems, simulation and animation, telerobotics and teleoperation.**

I. Introduction

A TYPICAL teleoperation system consists of a remote (follower) robot with one or more cameras to visualize the environment; these cameras may be mounted on the target robot, on a separate robot, or fixed to the environment. The camera feedback is displayed to the operator, who uses an input device (leader) to control the motion of the robot and possibly also of the camera. One compelling use case is minimally-invasive surgery, where the end-effectors of the remote robot and cameras are inserted through small incisions into the patient's body. The most popular telesurgical robot is the da Vinci® Surgical System (Intuitive Surgical Inc., Sunnyvale, CA). Both the robot and its simulator, which surgeons must train on, maintain an intuitive alignment between the patient and master console orientation. However, as a research community is growing around the open-source da Vinci Research Kit (dVRK) [1], researchers are exploring pairings of the patient console with different input devices, as well as with the master console controlling different robots.

More generally, while novel methods of visualizing surgery have improved the outcomes of various procedures [2], non-aligned scenes increase the mental task load for surgeons [3]. It is difficult to prototype and test different viewpoints, however, as current simulation software packages, such as Gazebo [4], do not focus on teleoperation. The relation between the input device, the telemanipulated robot, and the scene is often left for the user to constrain. There is a need to better understand how adjusting perspective affects task performance and how multiple users can share control, with the consideration that each user may not even have the same view while sharing the control of a common robot. This flexibility compounds the motion resolution of controllable components (camera and the robot) as the intermittent control of a shared camera (initiated by specific events) requires the computation of the correct mapping for each user as well as the mapping imposed by the changing camera viewpoint on the shared or unilaterally teleoperated robots. Moreover, ensuring continuously smooth teleoperation of the controllable components while accounting for user-triggered events is necessary and requires managing several coordinate frames and transformations. This work thus identifies a complete set of coordinate frames for a customizable control ranging from a single user, single camera and single follower robot to multiple users, shared cameras and shared robots and implements the underlying methods on a novel open-source simulator. The primary contribution is a systematic framework for the customizable pairing of input devices to simulated proxies, and defining coordinate frames and then automatically updating the transformations to maintain consistency for each user participating in the shared teleoperation environment.

Manuscript received October 15, 2020; accepted February 8, 2021. Date of publication February 26, 2021; date of current version March 22, 2021. This letter was recommended for publication by Associate Editor Z. Li and Editor P. Valdastri upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation (NSF) through the NRI 1637759, in part by AccelNet OISE under Grants 1927354 and 1927275, and in part by Johns Hopkins University internal funds, and an agreement between Johns Hopkins University and the Multi-Scale Medical Robotics Centre, Ltd. *(Corresponding author: Adnan Munawar.)*

Adnan Munawar, Jie Ying Wu, Russell H. Taylor, and Peter Kazanzides are with the Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: amunawar@jhu.edu; jieying@jhu.edu; rht@jhu.edu; pkaz@jhu.edu).

Gregory S. Fischer is with the Department of Robotics Engineering, Worcester Polytechnic Institute, Worcester, MA 01609 USA (e-mail: gfischer@wpi.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3062604, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3062604

II. Previous Work

A generic implementation of a shared multi-user control where multiple users can share the control of a common follower robot is a multi-fold problem. To the best of our knowledge, our

work is the first to explore and address the different aspects of this problem holistically. Previous works have identified different aspects of this problem which we review below.

In surgical scenarios, flexibility in camera viewpoint and intelligent input tool coordinate frame adjustment has been shown to improve user performance in surgical subtasks. Draelos *et al.* [3] found that in a mock surgery scenario, arbitrary viewpoints with automatic hand-tool alignment can reduce task completion time by 50% compared to a top-down view. Similarly, Omarali *et al.* compared different camera configurations' effects on recognizing objects in a virtual reality environment [5]. Koreeda *et al.* [6] show that by allowing study participants to vary the angle of the viewpoint from the endoscope camera, they can reduce errors while suturing. While they built a real setup with a joystick and suturing phantoms for their experiment, our system would allow them to test for the optimal viewpoint control in simulation.

More generally, other works have considered different sets of alignments between the input device, the robot, and the camera and how the alignments affect task completion. We highlight selected works and refer to [7] for a more comprehensive review. Most works calibrate the coordinate systems between different users and end-effector frames for specific tasks. Specifically, Nudehi *et al.* [8] addresses it for shared control where a trainer and a trainee drive haptic devices that are aligned. Hiatt and Simmons discuss various orientations between the user coordinate frame and the robot coordinate frame for telemanipulation [9]. The three main choices were user-centric, robot-centric, and task-centric and they fix the mappings for each. They point out that humans switch between reference coordinate frames naturally (e.g., "Go north until you reach the river and then turn right") yet human-robot interactions do not. By defining a general set of coordinate frames rather than fixing transformations between these, this work aims to simplify switching between these views.

DeJong *et al.* [10] define three coordinate frames at the remote (robot) site. They are the site's world coordinates, the robot's control coordinates, and the camera view's coordinates. At the local (user) site, there are four coordinate frames: the global coordinates, the manipulandum's control coordinates, the display view's coordinates, and the human operator's view coordinates. They found that there is a heavy mental burden when people are teleoperating robots and have to manually align these coordinate frames. We believe that an important goal of the robot interface should be to provide intuitive alignments between these coordinate frames to reduce the mental burden. We expand upon this framework to define additional transformations between the robot base and tip, the input device's base, and tool and unite the two sites. This allows the user to model a more flexible combination of control devices and robots and easier prototyping.

III. Problem Formulation

We define multi-user control (or multi-user shared teleoperated control) as the pairing of a single teleoperated Simulated Dynamic End-effector (SDE) to multiple Input Interface Devices (IIDs), as shown in Fig. 1. The IIDs may be arbitrarily different and the users manipulating them may share a single camera or have independent cameras through which they perceive the simulation or the environment. This definition is also satisfied by multiple users teleoperating multiple SDEs, provided that at least one SDE is controlled by more than one user. Concerning multi-user control, the terminologies that are used throughout this manuscript are described in Table I. The overall problem of multi-user control is motivated by previous works [5], [9] respectively showing that different users may have different preferences for camera and controller alignments. We break this problem down into three components which are elaborated as follows.

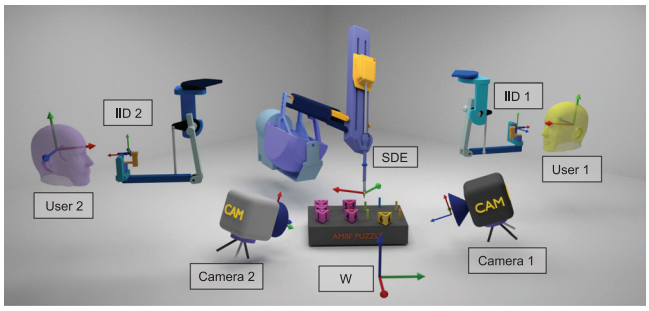


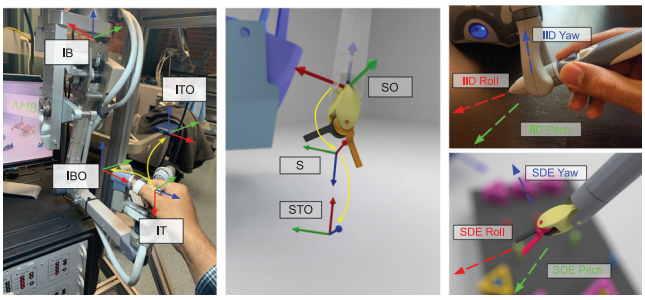
Fig. 1. An example scenario of two users controlling a single Simulated Dynamic End-Effector (SDE) in a shared fashion using their own input interface devices (IIDs). Each user has independently controllable viewpoints (cameras).

TABLE I
Table of Terminologies and Convention

Input Interface Device (IID)	This is a physical input device that can be used to interact with the simulation. The device may or may not have haptic feedback.
Simulated Dynamic End-Effector (SDE)	SDE is a simulated entity that acts as a proxy to the IID. SDE could refer to a single simulated body (link) or group of connected bodies (e.g., a palm link and two forceps links).
Clutch	A button (or an event) used to engage/disengage the control of an SDE, the Camera or reposition the IID.
Telerobotic Unit (TU)	This is a triplet of an IID, an SDE and a group of cameras. Each user has an exclusive TU with an exclusive IID but may have shared SDEs and cameras.
Frame Naming Convention	The first letter in a Frame indicates the component to which the frame belongs to. Thus (S), (I) and (C) indicate SDE, IID and Camera, respectively. Afterward, (B=Base), (T=Tip), (O=Offset), (BO=Base Offset), (TO=Tip Offset), (CL=Clutch), (PCL=Pre-Clutch), (R=Reference) and (RO=Reference Origin).

A. One-to-One Pairing of an IID-to-SDE

The general problem while pairing a single IID to an SDE can be understood from Fig. 2. The IID's tip (the part where a user may grasp it) is usually defined in relation to the IID's base coordinate frame. The SDE is by definition at a pose in relation to the simulation world's coordinate frame. From the **user's perspective**, the SDE's motions must be in relation to the eye (or the Simulated Camera paired to the user's view). The goal here is to align the motions mappings of the IID to the SDE and nullify any unintended linear and angular offsets.



(a) (b) (c)

Fig. 2. Identification of Frames for IID and SDE. The yellow arrows indicate that the initial origin of the frames coincide.

In terms of orientation, one may desire: (1) a fixed offset between the IID's and SDE's coordinate frame, and (2) an "angular motion mapping" (or "angular mapping") between the angular motion of an IID and the paired SDE. For example, the angular motion of an IID around the $\vec{X}$ axis may result in the angular motion of the SDE around $\vec{Y}$ (or any arbitrary axis). Depending on the origin of the mesh comprising the SDE, both the offset and the angular mapping need to be user specifiable.

Similar to angular mapping, "linear motion mapping" (or "linear mapping") affects the relationship between the linear motions of the IID and that of the SDE in the user's perspective. To generalize linear mapping, a new base coordinate frame (IBO) is required for all IIDs. The orientation of this frame alters the linear mapping as the tip velocities are now resolved in this differently oriented frame. However, altering the orientation component of this base frame imparts an offset on the tip frame which will be reflected at the SDE in the camera or the world frame and this must be accounted for using another coordinate frame (ITO).

B. Incorporating Multi-User Pairing and Shared Control

The pairing of multiple IIDs to a common SDE in shared or independent cameras and shared control has several requirements such as 1) a software API allowing customizable access to SDEs, cameras, and architecture to execute multiple Telerobotics Units (TUs), 2) specifying the ultimate reference frames for each user (such as the cameras), and defining mapping transforms for both the IID and the SDE in relation to the aforementioned reference frames, 3) ensuring the smooth operation of the SDE and the force feedback on the IIDs between intermittent clutching events by sharing users and 4) segregating collision-based forces and applying them to only the users that go beyond the initial collision of the SDE with an obstacle.

Regarding point (1), in simulators such as Gazebo [4] and VRep [11], IIDs can be incorporated via available plugin interfaces that are sequentially executed with the physics simulation. The underlying sequential architecture is unsuitable for incorporating a varying number of IIDs with real-time communication requirements alongside a real-time simulation. This is primarily due to the circular deterioration associated with carryover timing constraints.

As an alternative, we proposed an asynchronous architecture in our previous work [12] which separated each IID and the physics simulation into separate software threads. While this approach remedies the circular deterioration problem, it introduces new challenges such as accessing (reading and writing) common coordinate frames for shared cameras, SDEs, and IIDs without causing race conditions and accounting for IIDs with different update rates.

As simulators generally allow the camera basis (direction axes) to be defined arbitrarily, this means that the mapping frames in point 2) are camera dependent and thus need to be redefined. Moreover, choosing the camera as the reference for linear and angular motion is not always the case. Natural head motion and corresponding hand motion, and replicating this in simulation, is one such example.

Point (3) above is partly related to the pose computation equations and the control logic which is affected by the architecture of the software framework. This problem also requires handling more coordinate frames and transformations and is discussed in Section III-C.

Point (4) above is specific to shared control. One can imagine a scenario where two sharing users navigate the SDE to slightly contact an obstacle and stop. At this point, both the users should feel a gentle opposing force. Now, one user starts navigating their IID further into the direction of the obstacle and thus feels an increasing magnitude of force feedback which should not be felt by the other user. On the other hand, while operating in free space, the users should feel a force when their imparted motions to the SDE are in different directions.

C. Incorporating Shared Clutching

The purpose of clutching is to engage and disengage the motion of an IID to a paired SDE based on whether the clutch is released or pressed. Apart from providing a layer of safety, clutching allows for the repositioning of the IIDs within the user's comfort zone and/or away from the workspace boundaries. For multi-user control, clutching is equally, if not more, important. However, a clutch-engage by one user should not prevent other users from moving the paired SDE if the associated clutches are released. Secondly, in the event of releasing a clutch, the control of the SDE should be continuous and no jerks should be imparted by the corresponding IID nor a discontinuous feedback force be felt by the user as a result of the SDE moving while the clutch was pressed.

Similar to SDE clutching, the users may change the pose of their controllable camera by pressing a camera clutch and moving their IID. The challenges to camera clutching in multi-user control can be understood from the following example. Two users (***User A*** and ***User B***) share an SDE and a single camera. ***User A*** intends to move/reorient the camera and thus engage their camera clutch. This action disengages their SDE control and engages the camera control. Any active change in camera pose by ***User A*** should not prevent ***User B*** from continuously and smoothly teleoperating the SDE. However, the change in motion mapping must be continuously evaluated and applied to the motions of ***User B***.

TABLE II
COORDINATE FRAMES FOR MULTI-USER CONTROL AS SHOWN IN FIG. 1 AND FIG. 2. COLOR CODED TO MATCH FIG. 3

Idx.	Specification	Frame	Description
1	Implicit	W	Simulation world frame
2	Implicit	U	User perspective frame
3	Implicit	IB	IID base frame
4	Implicit	IT	IID tip frame
5	Explicit	IBO	IID base offset frame
6	Explicit	ITO	IID tip offset frame
7	Explicit	SO	SDE offset frame
8	Explicit	STO	SDE tip offset frame

TABLE III
RANSFORMATIONS COMPUTED FOR EACH TU. COLOR CODED TO MATCH FIG. 3

Idx.	Transformation	Description
1	T_I^W	IID's frame in W
2	T_{ICL}^W	IID's clutched frame in W
3	T_{IPCL}^W	IID's pre-clutch frame in W
4	T_S^W	SDE's frame in W
5	T_{SR}^W	SDE's reference frame in W
6	T_{SRO}^W	SDE's reference's origin frame in W
7	T_C^W	Camera's frame in W
8	T_{CPCL}^W	Camera's pre-clutch frame in W

While the linear mapping must be updated to maintain intuitive teleoperation, it may be preferable to leave the angular mapping unchanged. This stems from the fact that IID's pair with the SDE's, especially the ones modeling patient-side surgical robots, in a specific predefined orientation, as illustrated in Fig. 2(c). However, since this is not a general case, both forms of orientation mappings need to be supported which requires the computation of additional transformation matrices.

IV. METHODS

Based on the problem formulated in Section III a set of coordinate frames and transformations have been identified for a general-purpose implementation of a multi-user (including even a single user) control. These coordinate frames are described in Table II and Table III and most of them are visualized in Fig. 1, Fig. 2 and Fig. 3. Next, we discuss the association of a group of coordinate frames to each object, i.e. IID, SDE and Camera and the clutching-related coordinate frames.

A. IID Related Coordinate Frames

A new base offset coordinate frame (IBO) is defined in relation to the IID's base frame. We set the rotation of this coordinate frame to align the linear mapping with the simulation world frame ($R_{IBO}^U = R_W^U$) and its position to nullify the position offset of the tip. Another coordinate frame (ITO) is defined in relation to the IID's tip to nullify the tip's orientation offset in relation to the (IBO). An example of setting these frames is presented in Section V-B. The two coordinate frames (ICL) and ($IPCL$) are assigned to (ITO) and associated with clutching events. These are discussed in more detail in Section IV-C.

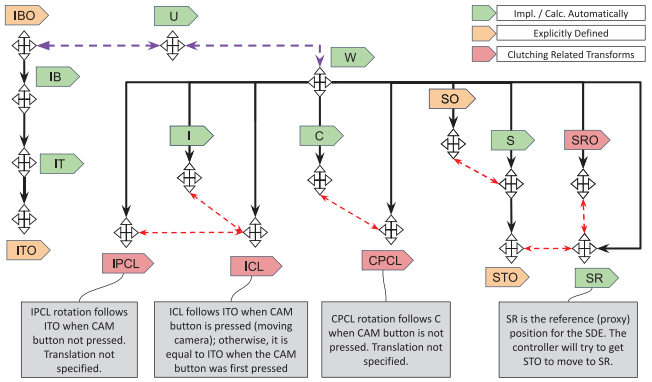


Fig. 3. Coordinate frames and transformations tree. The black arrows indicate the reference coordinate frame. The red dotted arrows indicate a caching relationship. The purple arrows indicate that relation has to be specified by aligning the frames in the user's perspective (See Section V-B1). For computation details, see Section IV-D.

B. SDE Related Coordinate Frames

A coordinate frame (SO) is required for matching the SDE's initial pose in the world and another coordinate frame (STO) is required to alter the angular mapping. Two additional reference coordinate frames (SR) and (SRO) are introduced which serve as a reference and a cached reference, respectively. These two coordinate frames enable smooth control of the SDE as a result of camera/SDE clutching for multi-user control and generalize to single-user control. The significance of these coordinate frames is discussed in Section IV-D.

C. Clutching Related Coordinate Frames

A typical teleoperation system supports at least the following three states for use of the IID: 1) Moving SDE (POS), 2) Moving Camera (CAM), and 3) Repositioning IID (not moving SDE or Camera). Often, this is implemented via two buttons or footpedals, so we adopt the terms "press" and "release" for the CAM and POS clutches.

In computer simulations, all the entities (rigid bodies, cameras, lighting, etc.) are defined in relation to a common coordinate frame (W) which can be considered as the world origin. For teleoperated control, the motions of the SDE are defined in relation to the camera's coordinate frame which is considered affixed to the user's perspective. Thus, a variation in the camera's pose alters the motion mapping of the SDE in the world coordinate frame (W), as discussed in Section III-C.

To implement the three states of clutching discussed above more coordinate frames are added to the IID and the camera, which are called (ICL), ($IPCL$) and ($CPCL$). The ($IPCL$) and ($CPCL$) are the $IID's$ and $CAM's$ poses before the corresponding clutch trigger while the (ICL) is the $IID's$ coordinate frame while the clutch is pressed. These coordinate frames ensure smooth transitions in a multi-user control setup.

D. Pose Computation Algorithm

The flowchart in Fig. 4 illustrates the pose computation algorithm that is executed repeatedly in each IID's update thread.

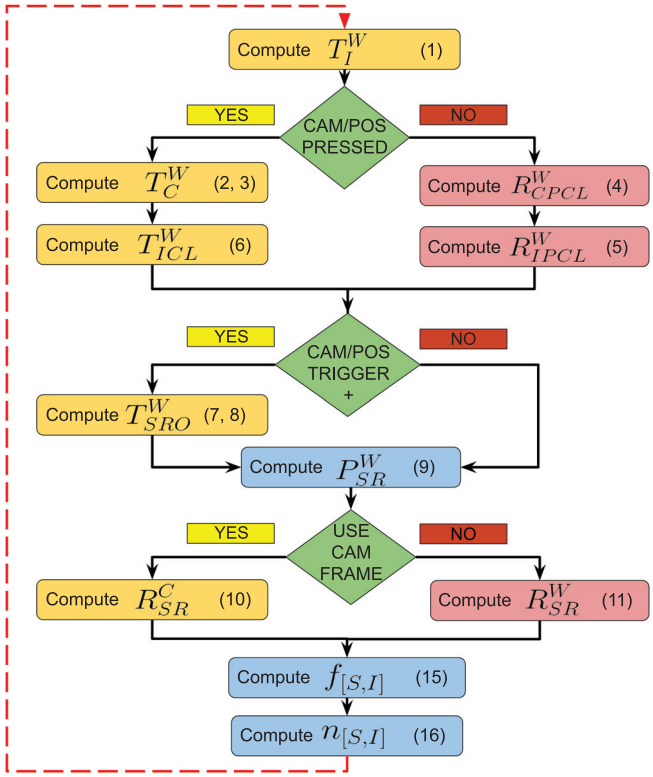


Fig. 4. The pose computation algorithm associated with each TU. The numbers in parentheses correspond to equations in Section IV-D. The TRIGGER+ evaluates to **True** only at the instance a button is pressed and then returns to **False**.

The labeled equations in Fig. 4 are provided in this section. The transformation matrix notation is similar to [13]. The first step is to resolve the IID in W using Eq. (1).

$$T_I^W := T_{ITO}^{IBO} = (T_{IBO}^{IB})^{-1} * T_{IT}^{IB} * T_{ITO}^{IT} \tag{1}$$

The camera position is updated based on the IID motion, scaled with a user-defined scalar term (S_C).

$$\vec{P}_C^W = \vec{P}_C^W + R_C^W * \delta \vec{P}_I^W / dt * S_C \tag{2}$$

The camera orientation is updated as:

$$R_C^W = R_{CPCL}^W * \left(R_{IPCL}^W\right)^T * R_I^W \tag{3}$$

It is worth pointing out that each IID is capable of moving its paired camera. This is unlike the da Vinci which uses two-handed (IID) camera control. When the camera is not being moved (CAM not pressed) or the Position clutch is not activated, the pre-clutch rotations for the CAM and IID are continuously updated as follows (which also causes the above equation to maintain a constant camera orientation):

$$R_{CPCL}^W = R_C^W \tag{4}$$

$$R_{IPCL}^W = R_I^W \tag{5}$$

CAM or POS button press updates the clutched IID pose:

$$\vec{T}_{ICL}^W = \vec{T}_I^W \tag{6}$$

When a trigger event (button press) occurs, the SDE reference (proxy), SR, is saved as SRO.

$$\vec{P}_{SRO}^W = \vec{P}_{SR}^W \tag{7}$$

$$R_{SRO}^W = R_{SR}^W \tag{8}$$

The $(\vec{P}_{SRO}^W)$ allows force nullification based on clutching, smooth resumption of control after a clutching event, and collision-based force segregation as discussed in Section III-B. The $(\vec{P}_{SR}^W)$ is computed from $(\vec{P}_{SRO}^W)$ as:

$$\vec{P}_{SR}^W = \vec{P}_{SRO}^W + S_I * \left(R_C^W * (\vec{P}_I^W - \vec{P}_{ICL}^W)\right) \tag{9}$$

The S_I is a workspace scaling term. For variable orientation mapping (i.e., clutch based rotation), the R_{SR} is calculated as:

$$R_{SR}^C = R_{SRO}^W * R_C^W * \left(R_{ICL}^W\right)^T * R_I^W * \left(R_C^W\right)^T \tag{10}$$

For fixed orientation mapping (in relation to the simulation World), the R_{SR} is calculated as:

$$R_{SR}^W = R_{SO}^W * R_{STO}^S * R_I^W * (R_{STO}^S)^T \tag{11}$$

While the pose computation algorithm is executed in individual IID threads, some transforms are shared outside; these include the camera transform (T_C^W), the SDE's transform (T_S^W) and its reference (T_{SR}^W). These reference transforms are then used in the control algorithm discussed next.

E. Control Algorithm

The reference coordinate frames computed from Section IV-D can now be used to move the SDE and provide a haptic response to the IID. The input to the haptic IID is a wrench (force and moment) while the input to the SDE may either be a wrench or a twist (linear and angular velocity). Thus, two independent control laws are used for the SDE and the IID and each produces a 6 degrees of freedom (DOF) Cartesian command $\vec{F} = [\vec{f}_{[S,\,I]}, \vec{\eta}_{[S,\,I]}]^T$. The notation $[S,\ I]$ is shorthand for $[SDE,\ IID]$ and implies two distinct equations expressed together. For example $\vec{f}_{[S,\,I]}$ represents the linear command $\vec{f}_S$ and $\vec{f}_I$. We use discrete PD control laws to derive an action and a reaction Cartesian command for the SDE and the IID.

$$\Delta \vec{P}_n = \left(\vec{P}_{[S,\,I]}^W - \vec{P}_{ref}\right); \quad \Delta R_n = (R_{[S,\,I]}^W)^T * R_{ref} \tag{12}$$

$$[\overrightarrow{ax}_n, \Delta\theta_n] = ToAxisAngle\,(\Delta R_n) \tag{13}$$

The angular damping is computed in Eq. (14):

$$\Delta^2 \vec{\theta} = \left((\overrightarrow{ax})_n * (\Delta\theta)_n) - ((\overrightarrow{ax})_{n-1} * (\Delta\theta)_{n-1})\right) \tag{14}$$

The error terms are then used in the control law:

$$\vec{f}_{[S,\,I]} = \vec{K}_L * \Delta\vec{P}_n + \vec{B}_L * (\Delta\vec{P}_n - \Delta\vec{P}_{n-1})/dt \tag{15}$$

$$\vec{\eta}_{[S,\,I]} = R_{[S,\,I]}^W * (\vec{K}_{A_{[S,\,I]}} * (\overrightarrow{ax_n} * \Delta\theta_n) + \vec{B}_{A_{[S,\,I]}} * \Delta^2\vec{\theta}/dt) \tag{16}$$

```
input devices: [MTM-LEFT, MTM-RIGHT]

MTM-LEFT:
    hardware name: MTML
    workspace scaling: 5
    simulated multibody: "Gripper.yaml"
    # root link: palm_link
    haptic gain: {
        linear: [0.05, 0.05, 0.05],
        angular: [0.0, 0.0, 0.0]
      },
    controller gain: {
      linear: {
        P: [200.0, 200.0, 200.0],
        D: [20.0, 20.0, 20.0]
      },
      angular:  {
        P: [20.0, 5.0, 10.0],
        D: [2.0, 0.5, 1.0]
      },
    cameras: [left-cam, right-cam]
```

```
MTM-RIGHT:
  hardware name: MTMR
  workspace scaling: 5
  # simulated multibody: ""
  root link: palm_link
  haptic gain:
      linear: [0.05, 0.05, 0.05],
      angular: [0.0, 0.0, 0.0]
    },
  controller gain: {
    linear: {
      P: [200.0, 200.0, 200.0],
      D: [20.0, 20.0, 20.0]
    },
    angular:  {
      P: [20.0, 5.0, 10.0],
      D: [2.0, 0.5, 1.0]
    },
  cameras: [right-cam]
```

Fig. 5. Setting up multi-user control using ADF.

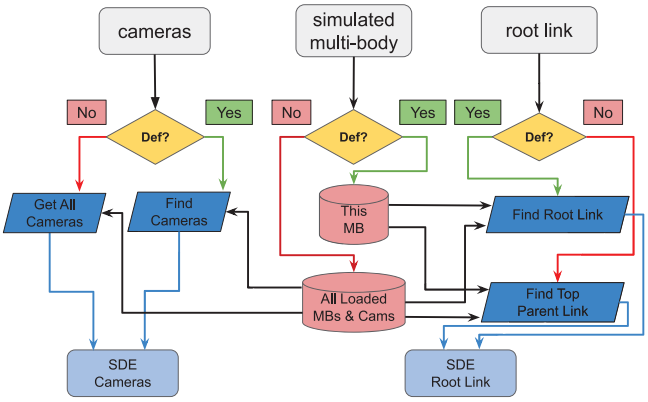


Fig. 6. Mechanism for loading the "cameras," "simulated multibody" and its "root link" specified via the ADF file.

In the equations above, $\vec{K_L}$ and $\vec{B_L}$ are linear stiffness and damping gains, which are 3×1 vectors usually with co-equal components. The $\vec{K_A}$ and $\vec{B_A}$ are the angular stiffness and angular damping gains and the individual components may not necessarily be identical. This is to compensate for the principal inertia of the IID or SDE if $\vec{\eta}_{[S,I]}]^T$ is used as a moment. The $\vec{ax}_n$ and $\Delta\theta_n$ terms are the axis and angle representation of ΔR_n and n is the sample number for discrete control. The linear and angular controller gains for the SDE and the haptic IID are independent of each other.

If the SDE belongs to an articulated set of links, the twist may be used to compute the joint velocities to drive the prior links of the SDE. It should also be noted that this PD control law can be replaced with any model-based control law or a cascade of control laws.

V. Implementation

The methods discussed in the manuscript have been implemented in the Asynchronous Multi-Body Framework (AMBF) [14]. The interface to the setting/changing coordinate transforms in AMBF is via dynamically loadable configuration files which are written in YAML[1] and called AMBF Description Format (ADF) files. The ADF files are the backbone of AMBF and model everything from rigid-bodies, joints, soft-bodies, sensors, actuators, world parameters and even IIDs.

A. Pairing a Camera to an IID

Fig. 5 shows an example of a YAML configuration file for multi-user control of two IIDs, called *MTM-LEFT* and *MTM-RIGHT*. The IID *MTM-LEFT* can control two cameras, called "left-cam" and "right-cam". The motions of any IID are always mapped to the first camera in the list of "cameras". The *MTM-RIGHT* IID has only one camera in its list called "right-cam". As *MTM-LEFT* also pairs "right-cam," this camera is shared between the two IIDs (i.e., both the IIDs can move the camera using their appropriate camera clutches). The cameras and their corresponding properties (i.e., Location, Orientation, Field-View Angle, etc.) are specified in a separate YAML file which is not covered in this manuscript but can be found at simulators web page[2]

B. Pairing an SDE to an IID

The ADF format simplifies the pairing of an IID to an SDE by allowing the specification of a "simulated multibody" and a "root link" as shown in Fig. 5. The field "simulated multibody" is the *filepath* to a robot/mechanism defined using a separate ADF file. This field is optional and, if set, the corresponding ADF file is loaded before the IID. Afterward, the "root link" of this loaded robot/mechanism is mapped to the IID for teleoperated control based on the Section IV-D and Section IV-E. This "root link" can be specified using the corresponding optional field. In the example of *MTM-LEFT* in Fig. 5, the field "root link" is commented out, which results in the automatic allocation of a root link. The automatic allocation algorithm searches a link with the least number of parents and in case the "simulated multibody" is a single rigid-body, it is the "root link" itself.

For the traditional teleoperated control of multiple users with independent SDEs, one may simply specify a "simulated multi-body" and optionally a "root link" for all the IIDs. On the other hand, for multi-user control, one can specify the "simulated multibody" field for only the first IID and the remaining IIDs should only specify the desired "root link" belonging to the previously loaded multibody. This algorithm for the camera and SDE pairing in AMBF using the ADF files is depicted in Fig. 6. As can be observed, ADF files simplify the setup for multi-user control.

1) An Example of IID-SDE Pairing: Here we demonstrate an example of IID-SDE pairing as shown in Fig. 7. The simulated robot is modeled after the Galen Head and Neck Surgery robot [15]. The SDE in this case is a simulated endoscope with the given linear and angular offset from (W) at home position. Similarly, the (IT) has a linear and an angular offset from (IB). The first step is to align the linear mapping between (I) and (W) with the placement of (IBO) which maps the (W) towards (anterior), right and up (superior) to that of the IID's. The origin

[1][Online]. Available: https://yaml.org/

[2][Online]. Available: https://github.com/WPI-AIM/ambf

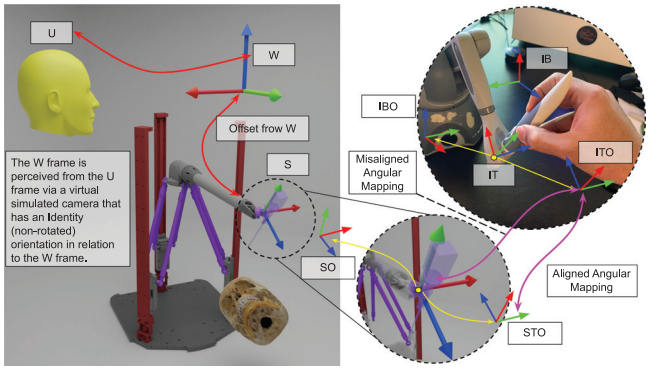


Fig. 7. One-to-one pairing between an IID and an SDE. The yellow arrows indicate the original position of the frames. The simulation world in this example has $\vec{X}$ towards the user, $\vec{Y}$ towards the right and $\vec{Z}$ upwards.

TABLE IV
MODES OF SHARED CONTROL AVAILABLE BY ENABLING (E) OR DISABLING (D) LINEAR/ANGULAR HAPTIC/CONTROLLER GAINS

	User 1		User 2	
Mode	Control Gains Lin. or Ang.	Haptic Gains Lin. or Ang.	Control Gains Lin. or Ang.	Haptic Gains Lin. or Ang.
SISO	E	E	E	E
SIAO	E	D	E	E
AISO	D	E	E	E
AIAO	D	E	E	D
SINO	E	D	E	D
AINO	D	D	E	D

of (IBO) is placed at the home position of (IT). The (ITO) may be placed at the same location as that of (IT) with a fixed order rotation (RPY) of $(\pi/2, 0, \pi/2)$ from (IT) to cancel the orientation offset between (IBO) and (ITO).

The (SO) sets the initial pose and (STO) sets the angular mapping of the SDE. If not specified, (SO) is taken from (S), and if specified, (S) is relocated to (SO). The (STO) is oriented around (S) such that the SDE's initial mapping of $\vec{X}$, $\vec{Y}$ and $\vec{Z}$ along (ITO)'s $\vec{Y}$, $\vec{X}$ and $-\vec{Z}$ axes can be remapped to its $\vec{X}$, $\vec{Y}$ and $\vec{Z}$.

C. Various Modes of Shared Control

In Fig. 5, the field "haptic gain" sets the linear and angular IID controller that influences the (***OUTPUT***) commands and the values in the field "controller gain" set the linear and angular SDE controller which regulates (***INPUT***) commands. The corresponding IID and SDE controllers were discussed in Section IV-E. By treating the values of each set of gains (linear and angular) as a binary value, where zero indicates one state (***DISABLED***) and a value greater than zero as the second (***ENABLED***), one can come up with various interesting control modes for linear and angular control. These are summarized in Table IV and discussed below:

1) ***Symmetric Input Symmetric Output (SISO):*** In this mode, the sharing users can control the SDEs and sense the haptic feedback.

2) ***Symmetric Input Asymmetric Output (SIAO):*** The users share control (***INPUT***) of the SDE, but not all users sense the haptic feedback.

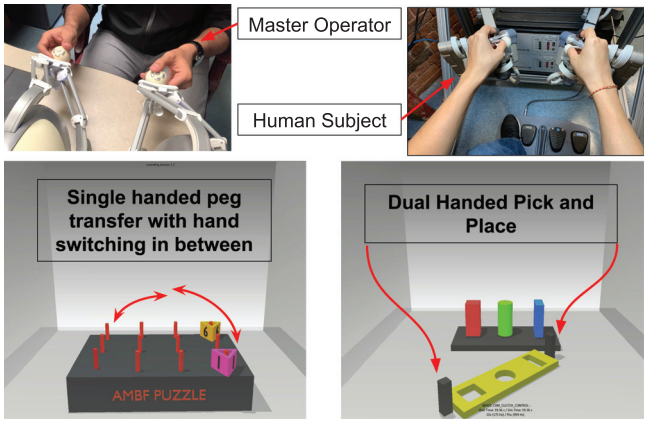


Fig. 8. Each operator has independent viewpoints (cameras). The first task (lower left) involves single handed picking, hand switching and placement. The second task (lower right) involves bimanual pick and place of the yellow puzzle in the base with matching extrusions.

3) ***Asymmetric Input Symmetric Output (AISO):*** All users can sense the force feedback but not all can control the SDE.

4) ***Asymmetric Input Asymmetric Output (AIAO):*** An example of this mode is where one user can only control the SDE while another user can only sense the force feedback.

5) ***Symmetric Input No Output (SINO):*** A mode that is similar to SISO except there is no force feedback.

6) ***Asymmetric Input No Output (AINO):*** A mode that is similar to AISO except there is no force feedback.

Between position and orientation control, any combination of the above control models is possible. For example, a specific scenario may involve SISO position control and AISO orientation control. Moreover, by utilizing the vector-based formulation of linear and angular gains in Section IV-E, the breakdown of control modes can also be done per axis.

VI. RESULTS

A preliminary user study was conducted to evaluate the application of the framework in a multi-user control setup. The study (IRB-20-0028) was approved by WPI's Institutional Review Board (IRB) and constituted 5 human subjects.

Fig. 8 shows the study setup in which the participants were seated in front of the dVRK's surgeon console. A qualified user (Master Operator) was in charge of aiding the study participant. The IIDs assigned to the Master Operator were the Novint Falcons (Novint Technologies, Inc., Albuquerque, NM) which have 3 D.O.F position control and force feedback. The study participants had both positional and orientation control but only force feedback. 3 of the 5 study participants had $\leq$2 hours of prior teleoperation experience with the dVRK. Each study participant performed a set of qualifying tasks for familiarity with the setup and was allowed additional attempts after successful qualification. An average of 2 additional attempts was performed by all users.

Two different tasks were used for evaluation that are shown in Fig. 8 and the participants were allowed training on each. For evaluation, three types of control modes were used:

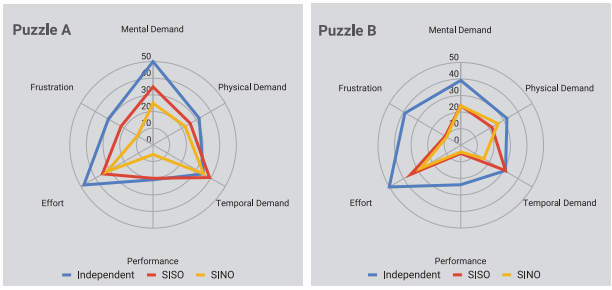


Fig. 9. NASA-TLX results for study of multi-user control.

TABLE V
RESULTS FOR MULTI-USER CONTROL USER STUDY. THE ABBREVIATIONS ARE DEFINED AS IND. = INDEPENDENT CONTROL, CF. = CLUTCHING FREQUENCY, DL = DISTANCE TRAVERSED FOR LEFT IID, DR = DISTANCE TRAVERSED FOR RIGHT IID, SD = STANDARD DEVIATION

Task 1 (Study A)								
Type	Time(s)		CF.		DL(m)		DR(m)	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Ind.	57.32	12.57	3.6	0.89	0.45	0.13	0.35	0.14
SISO	60.56	21.14	3	1.22	0.38	0.16	0.37	0.19
SINO	41.51	5.71	2.2	1.48	0.54	0.65	0.38	0.75
Task 2 (Study B)								
Type	Time(s)		CF.		DL(m)		DR(m)	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Ind.	57.34	19.89	6.8	2.58	1.21	1.18	1.6	2.11
SISO	39.91	2.96	4.8	1.30	0.67	0.25	1.28	1.3
SINO	28.52	4.1	2.8	1.78	0.48	0.31	0.80	0.86

1) Independent control with haptic force feedback where the Master Operator had no role, 2) Positional SISO control and 3) Symmetric Position ***INPUT*** with no force feedback (SINO). Evaluation criteria included 1) task completion time in seconds, 2) positional clutching frequency 3) distance traversed by MTML (DL) and MTMR (DR) in meters and the Reduced NASA Load Index (TLX) [16].

The study outcomes are presented in Fig. 9 and Table V. The participants preferred multi-user control in almost all of the Reduced NASA TLX [16] metrics and SINO control fared better than SISO control. Other noticeable metrics are the reduction in traversed path length (DL and DR) and task completion times (Time) except for SISO control for Study A. In this case, there was a higher standard deviation between the study participants.

As mentioned previously, this study was intended as a demonstration of the system and not as a conclusive study of its effectiveness which is a goal of our future work.

VII. CONCLUSION

We investigated the formulation of a generic framework for single and multi-user shared teleoperation, identifying a set of coordinate frames (discussed in Section IV-A and Section IV-B) that allow the pairing of different IIDs to different SDEs in a one-to-one or multiple-to-one fashion in independent or shared view coordinates (cameras). The identified set of coordinate frames modularize the pairing between the components of a TU by removing the interconnection between them. This simplifies and streamlines the inclusion of new IIDs and SDEs in the framework. Moreover, our methods of pose and control law computation (Section IV-D and Section IV-E) allow interesting shared control scenarios such as SISO, SIAO, AISO, AIAO, SINO and AINO (Section V-C). Lastly, our experiment with multi-user control (Section VI) demonstrates preliminary results indicating the feasibility of our implementation on improving the task performance and reducing the task load of simple teleoperation tasks by novice users.

REFERENCES

[1] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *Proc. Int. Conf. Robot. Automat.*, 2014, pp. 6434–6439.
[2] L. Qian, J. Y. Wu, S. DiMaio, N. Navab, and P. Kazanzides, "A review of augmented reality in robotic-assisted surgery," *IEEE Trans. Med. Robot. Bionics*, vol. 2, no. 1, pp. 1–16, Feb. 2020.
[3] M. Draelos, B. Keller, C. Toth, A. Kuo, K. Hauser, and J. Izatt, "Tele-operating robots from arbitrary viewpoints in surgical contexts," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2549–2555.
[4] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2004, pp. 2149–2154
[5] B. Omarali, B. Denoun, K. Althoefer, L. Jamone, M. Valle, and I. Farkhatdinov, "Virtual reality based telerobotics framework with depth cameras," in *Proc. IEEE Int. Conf. Robot Hum. Interactive Commun.*, 2020, pp. 1217–1222.
[6] Y. Koreeda *et al.*, "Development and testing of an endoscopic pseudo-viewpoint alternating system," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 10, no. 5, pp. 619–628, 2015.
[7] M. Shahbazi, S. F. Atashzar, and R. V. Patel, "A systematic review of multilateral teleoperation systems," *IEEE Trans. Haptics*, vol. 11, no. 3, pp. 338–356, Jul.–Sep. 2018.
[8] S. S. Nudehi, R. Mukherjee, and M. Ghodoussi, "A shared-control approach to haptic interface design for minimally invasive telesurgical training," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 4, pp. 588–592, Jul. 2005.
[9] L. M. Hiatt and R. Simmons, "Coordinate frames in robotic teleoperation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 1712–1719.
[10] B. DeJong, J. Colgate, and M. Peshkin, "Mental transformations in human-robot interaction," in *Mixed Reality Hum.-Robot Interact.*, 2011, pp. 35–51.
[11] E. Rohmer, S. P. N. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1321–1326.
[12] A. Munawar and G. S. Fischer, "An asynchronous multi-body simulation framework for real-time dynamics, haptics and learning with application to surgical robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2019, pp. 6268–6275.
[13] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ, USA: John Wiley & Sons, 2020.
[14] A. Munawar, Y. Wang, R. Gondokaryono, and G. S. Fischer, "A real-time dynamic simulator and an associated front-end representation format for simulating complex robots and environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1875–1882.
[15] C. He, K. Olds, I. Iordachita, and R. Taylor, "A new ENT microsurgery robot: Error analysis and implementation," in *IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1221–1227.
[16] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research," in *Adv. Psychol.*, 1988, pp. 139–183.