A Surgical Assistant Workstation (SAW) Application for Teleoperated Surgical Robot System

Authors: Min Yang Jung, Tian Xia, Rajesh Kumar, Anton Deguet, Russell Taylor, Peter Kazanzides, Johns Hopkins University, Baltimore, MD

Background: The Surgical Assistant Workstation (SAW) modular software framework provides integrated support for robotic devices, imaging sensors, and a visualization pipeline for rapid prototyping of telesurgical research systems. Subsystems of a telesurgical system—the master and slave—and the visualization engine form a distributed environment that requires an efficient inter-process communication (IPC) interface connecting the various subsystems. The SAW framework is easily extensible to allow local and network communication interfaces providing seamless integration of IPC.

Methods: Using the proxy pattern, we extend the CISST library *provided/required* interface command patterns for inter-thread communication underlying the SAW such that they can perform IPC over a network. Defined proxies for this extension include: task/device proxy, interface proxy, command object proxy, and event object proxy. The proxy objects are setup after duplicating local command objects and deliver remote command information using Internet Communication Engine (ICE) as a communication medium. ICE provides a simple yet powerful method for IPC. These extended SAW interfaces provide user applications with a very convenient way to run subsystem over a network with minimal code changes. Our research da Vinci surgical system uses a separate computing engine for each master and slave pair (left and right), and another for the video processing and visualization subsystem. We use this distributed configuration to validate the utility and performance of our extensions.

Applications: Our current goal is to allow seamless transfer of servo-level information between master-slave engines, as well as the visualization engines. For example, the availability of such subsystem state allows for implementation of task-level control for tasks that require more than one instrument to accomplish (e.g. a two handed knot tying task). We model such tasks as a constrained optimization using the current state of the knot and the position of suture (detected from image processing of the endoscopic image), and the position of the instruments used for tying the knots. These elements of state are distributed over the three computing engines, and we utilize the communication infrastructure above to provide the needed state information to each of the subsystems, for generating appropriate real-time assistive constraints on the instrument motion.

Conclusions: SAW applications are envisioned to use multiple sensing and manipulation devices that may be distributed on separate computing engines, requiring appropriate an IPC mechanism. We demonstrate an extension of the SAW framework with ICE, providing sufficient bandwidth for servo-rate communication. The implemented architecture requires minimal code changes for converting an integrated framework into distributed components that can then be operated on separate computing platforms.