Paper Review

Hanlin Wan

March 17, 2011

Title: Surgical Assistant Workstation (SAW) Communication Interfaces for Teleoperation

Authors: Min Yang Jung, Gorkem Sevinc, Anton Deguet, Rajesh Kumar, Russell Taylor, Peter Kazanzides

Source: CISST Wiki (https://trac.lcsr.jhu.edu/cisst)

Summary

Many frameworks and packages exist for the development of robotic surgical systems. An important aspect of such packages involves the communication interfaces between various robotic devices. This paper discusses one such framework, the Surgical Assistant Workstation (SAW) developed at the Johns Hopkins University.

SAW is a component-based framework, providing interfaces for the hardware and software modules commonly used in robotic systems for surgery. Hardware components include robotic devices, imagers, and sensors, while software components include video processing, 3D user interfaces, and robot motion control. SAW is based on the cisst libraries, and the networking functionality utilizes the Internet Communication Engine (ICE).

A main advantage of SAW is the simple application layer interface (API) that SAW utilizes in order to make the software development portion very straightforward. Components can be controlled in the same way regardless of their physical locations. Whether two devices are running in the same process on the same computer or communicating through complex network loops, SAW handles the two situations in the same fashion. SAW achieves this simplicity by using a proxy pattern design.

The proxy pattern design is used to replace the direct connection between two objects (Object A and Object B) with a conceptual local connection over the network. From the objects' points of view, the connection remains unchanged. Object A sees Object B as a direct connection, and vice versa. The proxy pattern design works the same way when the objects are far apart and connected through the network. One object sees the connection to the proxy of the other object as a direct connection. This design is completely hidden from the application layer and is managed internally, which contributes to the simplicity of the API.

Each object can be thought of as a server task, which sends out commands, or as a client task, which executes the commands. The server task has a provided interface proxy, while the client task has a required interface proxy. Typical commands that can be sent between the tasks include void, write, read, and qualified read.

A global task manager is used to maintain a list of tasks and their access information. Server and client tasks are registered to the task manager, and a connection is established over the network.

The paper explained this framework very clearly and then went on to test the performance of SAW in terms of networking overhead and teleoperation.

For the networking overhead experiment, four setups were used: 1P, 2P, 1H, and MH. Each test involved 10,000 qualified read commands, and each test was repeated 5 times. 1P involved two threads in a single process on a single machine with no networking. Latencies were on the order of 1μsec. 2P involved two threads in two processes on a single computer using a local network. Latencies were about 100μsec, most of which came from the overhead of the ICE. 1H involved two threads in two processes on two computers using a single-hop network with wired Ethernet. Latencies of 300μsec were observed due to additional networking overhead. Finally, MH involved two threads in two processes on two computers networked between a wired system and a wireless system. Latencies were on the order of msec.

The second experiment performed involved the control of a slave device by the master device. The Sensable Phantom Omni and the Novint Falcon were used. These were haptic devices with positioning sensing and force feedback. The master device was tested under two circumstances: point-to-point motion and sinusoidal wave motion. The feedback control was sent over a wired network in order to move the slave device in the same manner. Test data show a signficant delay in the movement of the slave device compared to the master device.

The paper concluded that SAW framework provided a simple API to work with while maintaining high performance. The network latency experiment showed low latencies in optimal network settings, suitable for high bandwidth closed loop control systems. The teleoperation experiment showed that the delay of the slave device is acceptable for haptic update rates for teleoperation tasks. Further studies with the da Vinci robotic system and video streaming were proposed.

Critique

Overall, the paper did a thorough job explaining the architecture of the Surgical Assistant Workstation. The figures used clearly explained the connections between various components. The network latency test clearly showed the efficiency of the communication interface between various objects. However, the results from the teleoperation experiment were not overly convincing.

While the networking latency test showed a worst case scenario of a 5ms latency, the delay in the motion of the slave device is clearly much longer from the figures, although exact numbers were not given. This is likely due to the networking latency experiment using a single type of command, while the teleoperation experiment likely had a mixture of various types of commands. The network latency experiment should be performed using random commands with varying types and sizes of data being transmitted.

Another issue with the teleoperation experiment is that the motions were very predictable. There was a straight moving path and a sinusoidal moving path. However, under real conditions, movement of the robots are unpredictable. Therefore, random motions should also have been used to test the delay. While simple, predictable motions can be compensated for by modeling the behavior, random motions cannot, and would yield more realistic delay times.

Finally, while the paper states that the delay is acceptable for teleoperation tasks involving haptic feedback, it is difficult to justify without performing actual experiments with a human. Further experiments should have been performed where a human is operating the master device in order to control the slave device. Qualititative data would be taken to get the feedback of the user on how much the delay actually affects the usability of the device.

While these early experiments show great promise of a simple and efficient system, further experiments are needed to validate their performance in real world situations.

Relevance

This paper is highly relevant to our project. We are developing an iPad application to configure various settings of different robotic devices. Therefore, using the SAW framework would be perfect for our project. Since all the devices already implement the SAW framework, adding our iPad application to this system should be relatively easy to do given the simplicity of SAW's API.

Furthermore, since the iPad will be connected to the network via a wireless interface, latency times are expected to be high as demonstrated from the network latency experiment. However, our current project is focused on configuring settings such as lamp brightness, where latency times are inconsequential. Therefore, the delays shown in this paper should not affect our current project. In the long run, more functionality may be added to the iPad application including video streaming or haptic feedback. These scenarios may find the latency times to be problematic.