# Control System Architecture for a Minimally Invasive Surgical Robot

Kenneth Fodero II+ BSEE, H. Hawkeye King+ BACS, Mitchell J.H. Lum+ MSEE, Clint Bland+, Jacob Rosen+ Ph.D, Mika Sinanan++ M.D., Ph.D, Blake Hannaford+ Ph.D.

*+ Department of Electrical Engineering, ++ Department of Surgery,*
*University of Washington, Seattle, WA, USA*
*E-mail:<kfodero, hawkeye1, mitchlum, cbland, rosen, mssurg, blake>@u.washington.edu*

**Abstract.** As the field of surgical robotics continues to evolve, it is important to keep patient safety in mind. This paper describes a safety control architecture aimed at moving an experimental system in the direction of intrinsically safe operation. The system includes safety features such as: a small number of states, Programmable Logic Controller (PLC) state transition control, active enable, brakes, E-STOP, and a surgeon foot pedal.

**Keywords.** Surgical Robot, Safety, Teleoperation, Telesurgery, PLC

## 1. Introduction

Surgical robotics is revolutionizing the way in which clinicians deliver health care. As the field of surgical robotics continues to evolve, it is important to keep patient safety in mind. In this paper we look at moving an experimental system in the direction of intrinsically safe operation. Although the resulting system is several steps short of a level of safety suitable for human surgery, we hope to evaluate concepts for a safety system which could facilitate a human rated design.

Safety has been considered by several other teams working on surgical robotics [1,2]. As early as 1991, a system with an emergency stop, foot pedal, and brakes was developed by Taylor, et al. [3]. Rovetta et al. looked at safety in terms of relevant industry standards [4]. Currently, the DaVinci, an FDA approved and human rated surgical robot system, includes such safety features as redundant sensors, hardware watchdog timers, and real-time error detection [5].

In this study, our main focus is on ways to increase our confidence that a highly complex system of software and hardware will always "do the right thing" or "fail safe." Our approach will rely on the following assumptions:

- A system with a small number of well defined states is inherently safer.
- Programmable Logic Controllers (PLCs) are a highly reliable off-the-shelf technology that can easily and reliably be programmed for small numbers of states.
- PLC implementation is more reliable than implementation of equivalent functions in a computer.

*Medicine Meets Virtual Reality 14, Long Beach CA, January, 2006*
*J.D. Weswood et. Al. (Eds.), IOS Press, 2006*

- Most of the complex real-time control functions will be managed by software.

This paper describes the design of a safety system for a surgical robot now under development in our laboratory.

## 2. Architecture

Based on an extensive database of dissection and suturing tasks performed by 30 surgeons, it was determined that 95% of the time surgical tools operated in a conical range of motion with a vertex angle 60 degrees [6]. We used these results to design the arms of our surgical manipulator with a numerical optimization process [7]. The safety system described in this paper is being developed for highly reliable operation of this manipulator in planned animal surgical experiments.

The basic hardware architecture of the system consists of a PC running a real-time version of the Linux operating system (RTAI). The PC communicates with a USB 2.0 interface card, designed in our lab. The interface card passes commands and information to and from the Linux host to the sensors, power amplifiers, and the PLC every 125us. The surgeon's foot pedal may be connected directly to the PLC for local operation or passed in through the network and I/O Board for remote operation. This allows remote or local operation of the pedal. Brakes on each motor engage if power fails or in the Pedal up or E-stop states under direct control of the PLC. The brake system is independent of possible bugs in the control software. The hardware architecture is shown in Figure 1.

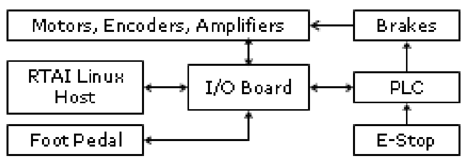


**Figure 1.** Hardware Architecture

The software for the system consists of a set of kernel modules under RTAI Linux. The Control Module contains the I/O, kinematics, control, and other functionality of the system. This module communicates with the system hardware through the USB 2.0 interface card. A function in the Control Module produces a 10Hz square wave for a watchdog timer function implemented in the PLC. User inputs are collected through engineer and surgeons interfaces. The networking module serves as an intermediary between these interfaces modules and the control module. The software architecture is shown in Figure 2. Allocation of software modules to computing hardware is flexible except for the I/O Board software and the PLC ladder logic.

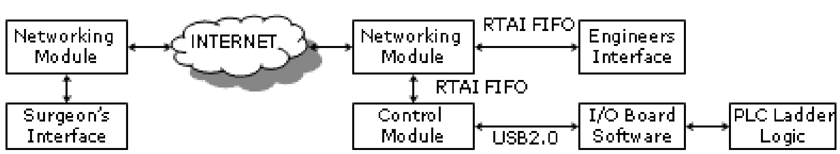


**Figure 2.** Software Architecture

The system operates in four states. These states are: Emergency Stop (E-Stop), Initialization, Pedal up, Pedal down. A Direct Logic 05 PLC (Figure 4) contains the state transition logic, and manages the transitions between the states. PLC state variables are connected to the USB I/O card and read by the Linux computer. The real-time Linux software (Control Module) will contain the same four states, but it transitions between states only in response to transitions of the PLC state machine. Transitions between pedal up and pedal down are controlled by surgeon inputs, either directly wired or relayed to the PLC. The state transition diagram for the system is shown in Figure 3.

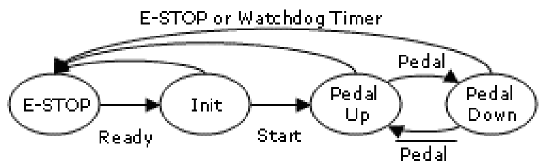


**Figure 3.** State Transition Diagram

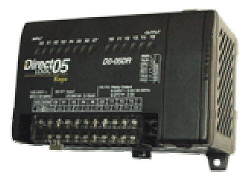

**Figure 4.** Direct Logic 05

The PLC contains watchdog timer logic which detects loss of the 10Hz square wave (in either 1 or 0 position) indicating a crash of the controller software, and initiates a transition to the E-Stop state, within 100ms. A faster watchdog timer would be desirable for more rapid shutdown, but reliable detection of signals faster than 10Hz was not possible with the selected PLC.

## 3. Conclusions

The control system architecture described in this paper is a crucial element in the overall surgical robot system under development in the University of Washington BioRobotics Lab. We expect our system will provide a level of predictability, reliability, and robustness sufficient for animal surgery evaluation.

## References

[1] Davies B.L. “The safety of medical robots.” In Proceedings of 29th ISR Conference on Advanced Robotics: Beyond 2000, Birmingham, April, 1998.

[2] Varley, P. “Techniques for Development of Safety-Related Software for Surgical Robots.” IEEE Transactions on Information Technology in Biomedicine, Vol. 3, No. 4, Dec 1999.

[3] Taylor, R., et. al., “Taming the Bull: Safety in a Precise Surgical Robot” IEEE International Conference on Advanced Robotics. Pisa, Italy, June, 1991.

[4] Rovetta, A, “Telerobotic Surgery Control and Safety,” IEEE International Conference on Robotics and Automation, vol. 4, pp. 2895-2900, 2000.

[5] Guthart, G. “The Intuitive™ Telesurgery System: Overview and Application,” IEEE International Conference on Robotics and Automation pp. 618-621, San Francisco, USA, April 2000.

[6] J.D. Brown, et. al., 'Design and Performance of a Surgical Tool Tracking System for Minimally Invasive Surgery,' Engineering Congress and Exposition: Advances in Bioengineering, BED, vol. 51, pp. 169-170, ASME, New York, Nov. 11-16, 2001.

[7] Lum, M., et. al., “Kinematic Optimization of a Spherical Mechanism for a Minimally Invasive Surgical Robot” 2004 IEEE International Conference on Robotics and Automation pp. 829-834, New Orleans, USA, April, 2004.

*Medicine Meets Virtual Reality 14, Long Beach CA, January, 2006*
*J.D. Weswood et. Al. (Eds.), IOS Press, 2006*