Controlling Robots via Internet

Aleksander Malinowski
Department of Electrical and Computer Engineering
Bradley University
Peoria, Illinois 61525, USA

Bogdan Wilamowski
Boise Graduate Center, 800 Park Blvd
University of Idaho
Boise, Idaho 83712, USA

ABSTRACT

This paper describes client-server systems for controlling robots over the World Wide Web. Robots perform robust real time manual control over the Internet connection characterized by varying bandwidth and latency. A Web server is used to provide the client application to the operator. Client uses custom TCP/IP protocol to connect to the server, which provides interface to the specific robotic manipulators. Sensors and a video camera provide the feedback to the client. Several implementations using various microcontrollers such as Motorola, Intel, Siemens, and Hitachi are described.

I. INTRODUCTION

The remote controlled operation started with Goertz and Thompson's demonstration of their first "master-slave" remote control in 1954 [1] and was followed by Ferrell and Sheridan [2] who focused more on the system control aspects. Recently tele-operation became the subject of intense research. Applications of such technology include inspection and exploration of hard-to-reach places and hostile or toxic environments [3,4]. Most current tele-operating systems are either basic extensions of direct manual control [3] or model-based supervisory control [2,5-9]. Direct control requires operator to send the steering signals to the remote manipulator continuously while it is moving. This approach has obvious drawbacks such as reduced stability of the control loops due to long delay caused by fully remote operation and dependence on the continuous connection between controller and controlled object [10,11].

It is desired that the remote manipulator would move autonomously and require only specifying its new desired destination or state. In this case there is no need for high-speed continuous communication. Data monitoring and control through a computer network or some other proprietary network is no longer prohibitively expensive and thus restricted only to industrial or luxurious products [12-15]. Continuously decreasing cost of microprocessors and network interfaces opened additional possibilities in the home automation applications.

The "Mercury Project" is the first successful implementation of the teleoperation via the Internet. In 1995 Goldberg et al. [16] developed a simple robotic manipulator with CGI program interface and video feedback. The model-based control for a complex system of multiple bulldozers was reported by Yeuk and Stark in 1999 [17].

This paper describes several client-server systems to control a moving robot over the World Wide Web, which were developed at Bradley University in years 98 to 01. Robot manipulators perform robust real time manual control over the Internet connection characterized by varying bandwidth and latency. A Web server is used to provide the client application to the operator. Client uses custom TCP/IP protocol to connect to the server, which provides interface to the specific robotic manipulators. Sensors and a video camera provide the feedback to the client.

II. SYSTEM ARCHITECTURES

Every system, regardless of the type of the networked appliance, has the following three hardware components:

- The controlled object – "robot".
- The embedded microprocessor either as a part of the controlled objects or as a separate device that interfaces the object to a personal computer or directly to the network.
- The operator – PC computer as a client on the computer network with a GUI interface or as a server being an interface between the controlled object and Internet or Intranet.

Many robotic manipulators may be connected at a time to the server through either serial or parallel ports of the computer. In case of autonomous robots the server passes the commands addressed to the robot. In case of simple robot server runs a separate process (a thread) that interfaces to the robot i.e. takes care of interpreting commands and direct control. At this moment it is not possible to remotely change either code of the interfacing process or reprogram the on board embedded system. However, such functionality may be also implemented in the future. At this moment only the whole server program can be changed remotely after transferring the new code to the server machine with temporary shutting down of the communication while it the server is being restarted.

Several possible strategies are possible. In the first strategy, shown in Fig. 1, robot is connected to a computer via an embedded system. In this case one computer controls several components but each of them requires its own microprocessor interface that is connected to a computer using for example a serial port.

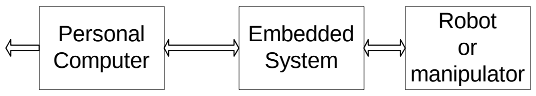

Fig. 1 Block diagram illustrating the first strategy

This strategy is suitable when each system is independent and microprocessor either is an integral part of the appliance or can be incorporated into it to enhance its functionality. In case of disconnecting the computer, each system becomes autonomous. The connection to the computer can be made via a serial, parallel or USB port, or via a wireless link.

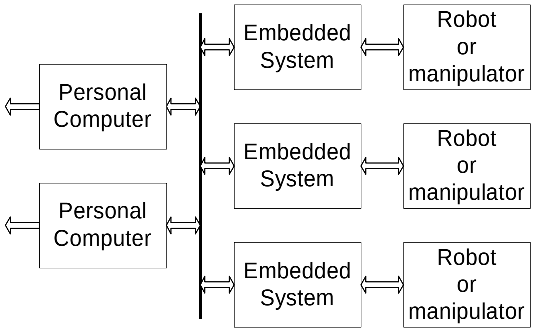

Fig. 2. Block diagram illustrating the second strategy

In case of the second strategy one computer can control several components and each of them requires a separate embedded system. However, the way each system is connected to the computer allows the components to interact with each other even if the computer is disconnected. This feature can be implemented using a proprietary network, a regular local area network (LAN) or using wireless network adapters, for example, such as Blue Tooth. In that case, the control could be performed from any computer that is connected to that internal network. This strategy is at the moment still more complex and a little more expensive. The embedded system software must implement some network protocols and be connected to an individual network interface card.

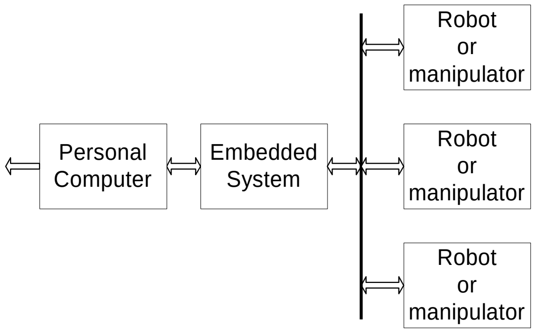

Fig. 3 Block diagram illustrating the third strategy

In case of the third strategy one embedded system is used to control several appliances. This approach is suitable when several appliances are already interconnected, adding a separate embedded system to each of them would increase cost significantly or there already is a proprietary network developed for several appliances that can be exploited.

Certainly all strategies can be mixed together by connecting several embedded systems using second strategy while some of those systems control several appliances using third strategy. Additional first type systems may be connected to a computer using for example a serial port. In addition, several systems may be interconnected via computers that are connected to a local Intranet. As long as there is no single standard for controlling appliances, the architecture possibilities are endless.

Each of the components listed above may be developed to various extend based on the nature of the application. For example, some manipulators can be allowed to freeze their state in case the remote control is lost while others need to carry on their essential functionality and respond to changes in the environment for instance to avoid their damage. Furthermore, for the sake of the modularity of the design, the complete system may consist of several specialized servers and clients per manipulator. For example: When live video feed is necessary but not required for the closed loop control it makes sense to use two separate client-server subsystems rather than make only one pair of server and client more complex.

The software components depend on the hardware architecture and usually include:

- The firmware for the embedded microprocessor.
- The software that interfaces a personal computer to the controlled object or to the embedded system.
- The graphical user interface for performing the control.
- The Internet server in case if the object is to be controlled via Internet or Intranet.

III. IMPLEMENTATION ISSUES

Our implementation of the robotic manipulators that that are the scope of this paper consists of several components that can be mapped to the general structure that was discussed earlier. In our case the most important features are:

- the direct real time control of the manipulator
- the low latency multimedia (at least video) feedback
- damage avoidance system to prevent damage to the manipulator due to collision or system overload
- low power in passive mode of operation
- autonomous return to the base (charging) station in case of low battery power

Those desired product prototype features had direct impact on the design of the system. The developed system consists of several components as outlined below:

- the manipulator movement control user interface (the client)
- the video feed receiver client
- the Internet as the communication media
- the manipulator control server (the control server) that interfaces the client to the semiautonomous controller and implements complex local control algorithms in case the connection is lost or impaired.
- the video feed transmitter (server)
- the Web server to allow for fast installation of the current version of the clients directly from the server computer
- the built-in robot controller that executes the operator commands and has built-in manipulator protection against overload or collision

In general, the client side software should run on any computer platform that is sufficient to run a Java compatible Web browser. In this particular implementation, however, the multimedia receiver was based on Microsoft NetMeeting ActiveX component and that limited significantly the possible platforms. All communication between the client and the server is handled by various Internet Protocols (IP) based protocols. Therefore any IP-compatible network may be utilized as a communications media.

The area of operation of robots is monitored by a single video camera WebCam II which works in the video capture mode. The image is buffered on the harddrive and transmitted to the remote apllet. It is also available for download using the generic Web server. Since one of the project's goals is to provide a complete Java solution (with the exception of the microprocessor systems embeded in robots) we are awaiting for Sun Microsystems to distribute libraries that would allow for direct access to the image capturing devices. When such library is available the third party video capturing software will be replaced by the new component buit in directly into WebBot server.

A. Client Side Components

There are two client-side software components: manipulator control user interface designed as Java applet and video receiver ActiveX plug-in. Those two software components are embedded in a Web page that is download from the remote Web server. This approach allows for the minimum maintenance of the system [18].

There are three threads in the control client software. One thread performs the task of listening to the user interface events and sending appropriate commands to the control server. The second thread requests transmission of the sensor readings. The third thread monitors feedback from the robot and updates user interface accordingly. The reason behind using a separate task to send the sensor reading updates is to achieve a better utilization of the connection bandwidth through the control of sensor reading update rate. The server would send the manipulator status and sensor reading regardless whether client requests an update or not.

B. Network Protocols

All network protocols used by the remote control system are Internet protocol (IP) based. This design decision was made because IP is a standard 3rd layer protocol that has an implementation in every major type of network. Certainly, if remote control is performed only within a local area network, another protocol, that is specific and native to that network might be used. Using custom protocols usually has some performance advantage and in case of extreme performance requirements may by the desired choice. In general, however, using commonly adopted standard cuts the development costs and offers more reliability because the software drivers for standard protocols are already developed and tested thoroughly.

The remote control system for the robotic manipulator uses the standard HTTP protocol for the Web server. The video transmission uses a protocol that belongs to H.323 standard of real-time multimedia communications and conferencing over packed based networks [19].

The real time robot control protocol can be performed thorough one or more virtual channels. Although all virtual channels are implemented in one network channel, there is some advantage to that strategy when certain control signals have more priority than other. In case of use of multiple virtual channels, some commands may be sent using reserved channel and thus bypass the others that are still in the outgoing queue at the client. The reserved channel may be prioritized in IP version 6 to achieve better quality of service [20].

The real time robot control protocol developed in this case study utilizes only one virtual channel and is based on connection oriented TCP/IP protocol. The reason for this design decision was relatively low bit-rate of the control commands sent and status reports received back. Since the traffic is bi-directional and continuous, the piggyback packet acknowledgement is used. Furthermore, TCP/IP also allows setting up a secure connection.

In order to allow fast and easy debugging, the developed protocol is text-based. A short one or two letter commands are followed by integer or real number parameters. A new line character is used as a end of command/new command indicator. The robot replies start with one character that indicates whether the reply is affirmative, informative (data), or an error message.

C. Server Side Components

Three main components provide the communication gateway between the robotic manipulator and the remote client that is connected via the computer network. The

Web server allows setting up the initial communication between the two computers and provides the client configuration Web page that request downloading and starting the applets. The project described in this paper utilizes Apache Web Server [21]. The second component is the multimedia (video) server. Microsoft NetMeeting is used for that purpose.

The next two most important components are the manipulator control server and Saphira robot client software. Saphira is multithreaded software that was developed by Active Media Robotics [22] that interfaces the high end user commands to the firmware low-level robot control server described in the next section of the paper. It connects to the robot microprocessor via RS232 serial port.

The control server is designed as a modular software application that consists of a simple command server and several additional modules. Those additional modules are dedicated to perform certain more complex actions. They are configured either to be overloaded by user or to overload the user. For example, the collision detection system overloads the movement command issued by the user.

D. Embedded Controller

Several different controllers were used in the series of tele-operation projects. The first, shown in Fig. 4, project used a controller that was simulated on a personal computer that was interfacing the motors and sensors directly via parallel data ports [23]. The manipulator does not have any controller; the circuit directly triggers motors or actuators. That approach turned out to be inefficient and not always reliable due to use of non-real-time operating system.

Figure 4 shows the interface between network-connected computer and wireless transmitter that send commands to the manipulator.

Fig. 5 shows an improved version of the “Lego Robot” that uses a small remote controlled car. The interface showed in the picture above was directly connected to the remote control device for this robot.
Fig. 6. shows the image transmitted by a camera placed on the remote controlled car that is shown in the Fig. 5. The explorer system illustrated in Fig. 5 and 6 is controlled with a parallel ports [23] of PC. Explorer system consists of seven main components:

- Remote control interface that runs inside a Web browser of a remote user's computer.
- Web server providing the remote interface for downloading as a Web page with embedded Java applet.
- Robot server written in Java, interfacing the remote clients connected using a custom TCP/IP protocol.
- Video monitoring system providing visual feedback. Currently only one view is available, either from the side or from camera mounted on the robot.
- Remote controlled robotic manipulator interfaced to the network-connected base PC.

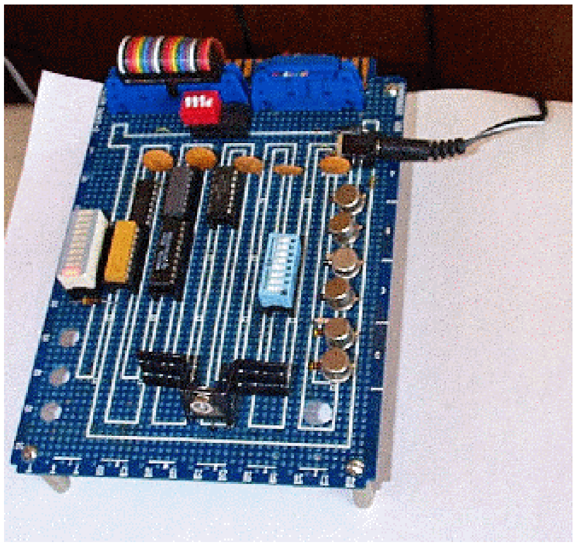
Fig. 4. The interface for wireless communication.

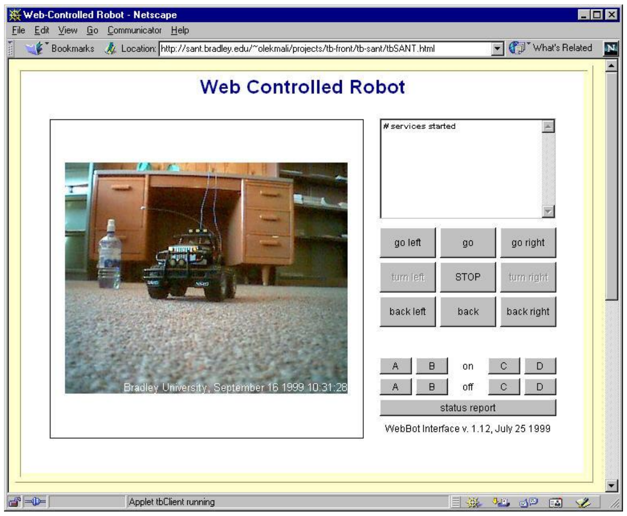
Web-Controlled Robot - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://sant.bradley.edu/~olekmali/projects/tb-front/tb-sant/tbSANT.html
What's Related
Web Controlled Robot
# services started
go left
go
go right
turn left
STOP
turn right
back left
back
back right
A B on C D
A B off C D
status report
WebBot Interface v. 1.12, July 25 1999
Bradley University, September 16 1999 10:31:28
Applet tbClient running

Fig. 5. The web page to control the explorer with the feedback through on side camera.

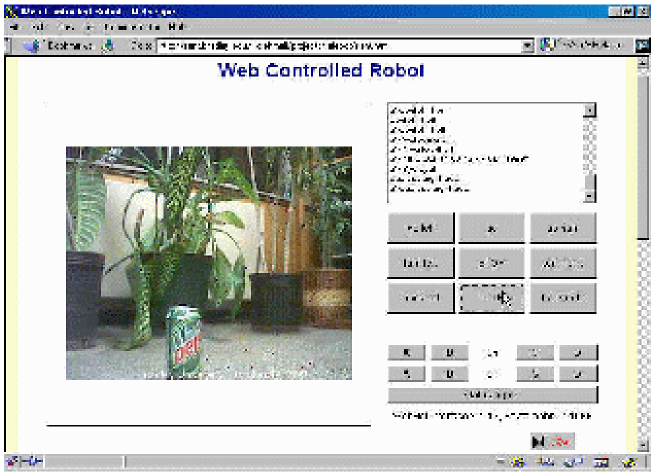

Fig. 6. The web page to control the explorer with the feedback through the camera on vehicle.

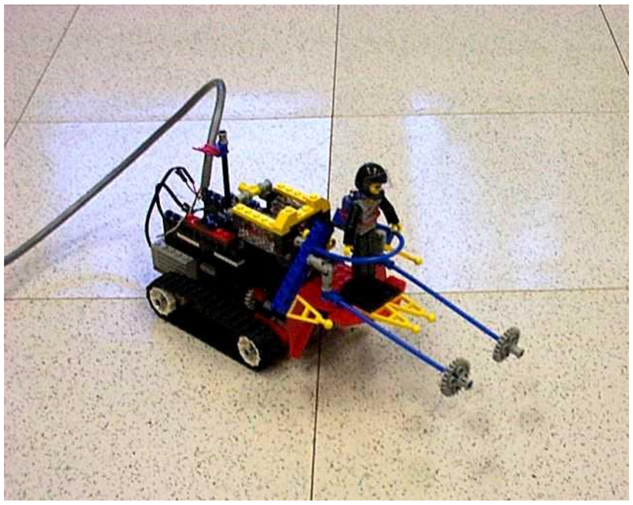
Fig. 7. Lego robot.

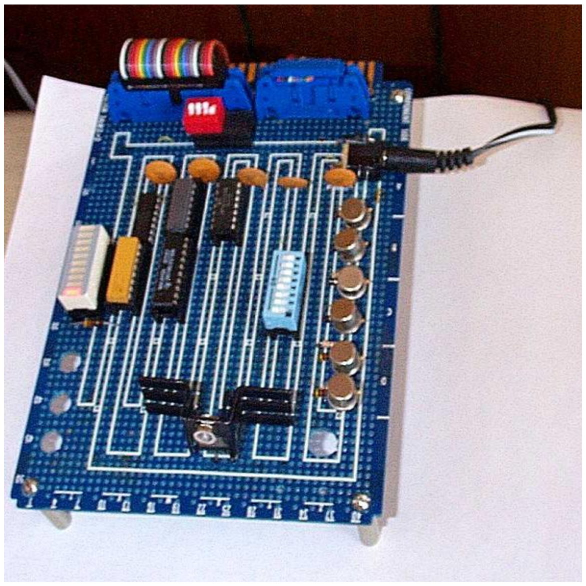
Fig. 8. The web side for the lego robot of Fig. 7.

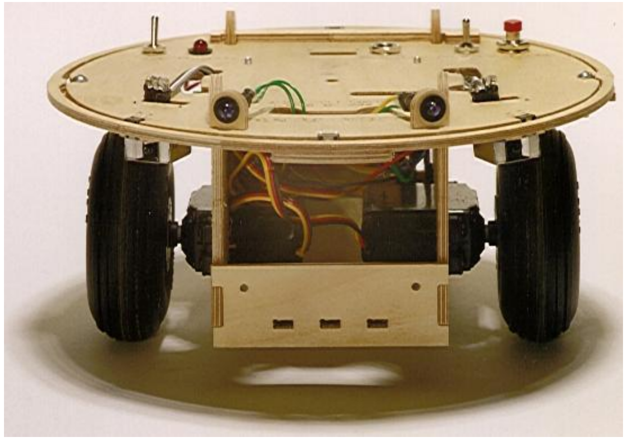
Fig. 9. “Florida robot which uses Motorola HC11 microcontroller on the board.

Fig. 7 shows the very first prototype, also referred to as the “Lego Robot”[25]. Fig. 8 shows the control screen on the web page for the prototype of Fig. 7. The lego robot controlled with a parallel port and with a stationary camera located off side [24].

Other project, shown in Fig. 9, utilizes a Motorola HC11 microprocessor that implemented motor controllers and collected data from sensors. HC11 connects to the server computer via a serial link [25]. It has several infrared sensors and bumper switches. Fig. 10 shows the robot with Intel 8051 controller, which communicates with PC through a serial port.

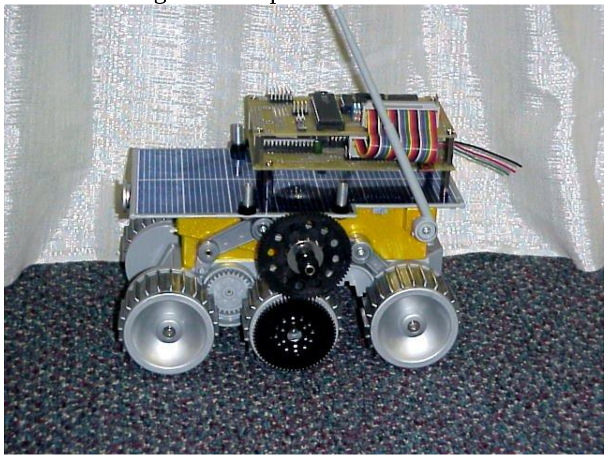
Fig. 10 The robot with 8051 controller on the board..

IV. DESCRIPTION OF SELECTED IMPLEMENTATIOS

Although, it is possible to develop and integrate all components under one, real time operating system partitioning components into two systems that communicate to each other was found beneficial. One of the systems, usually a simpler one, is solely dedicated to performing time sensitive control operations. All other tasks that can be run in less demanding environment can be run on the second computer. That approach significantly lowers the complexity of each component thus increases reliability and shortens the overall development time.

Several projects that involve remote control of different robotic manipulators were developed during 1998-2001. Some of them were described in the previous section others are shown in Fig. 11 to 14.

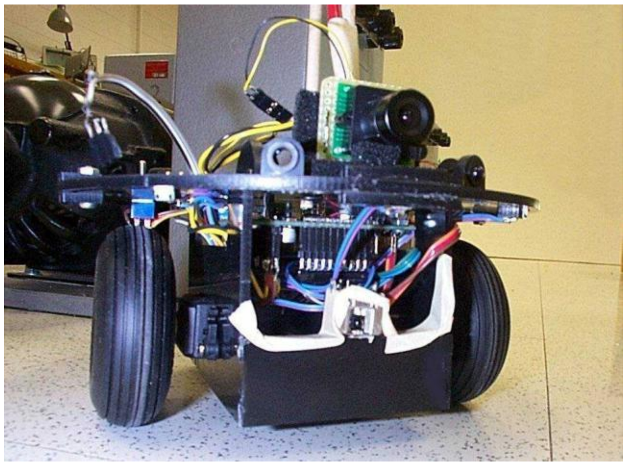
Fig. 11. Robot with a TV camera on the board

The robot shown in Fig. 11 has a small TV camera on the top. This robot was in operation in for about one and half years. The playground for this robot is shown in the Fig. 12.

The Magellan Robot, shown in Fig. 13, was purchase for development of a project founded by a local private company. The robot has its own Hitachi microcontroller that runs a real time operating system and is connected to a PC104 computer. The PC104 computer is connected to the Internet via the wireless network card. It also had a video frame grabber and good quality cameras for fast and sharp images. Please note abundant number of infrasound distance sensors, the infrared sensors and bumper switches mounted on every plate.

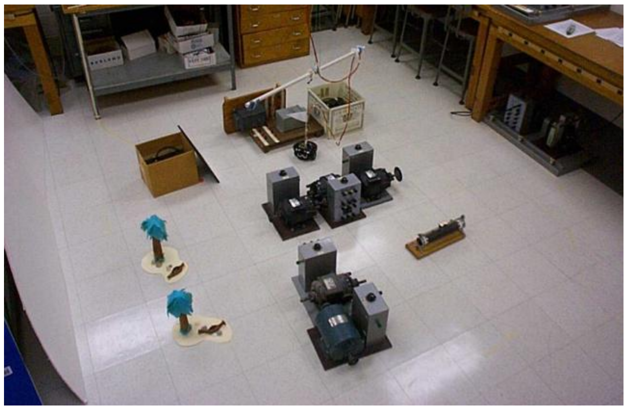
Fig. 12 Playground for the robot of Fig. 11.

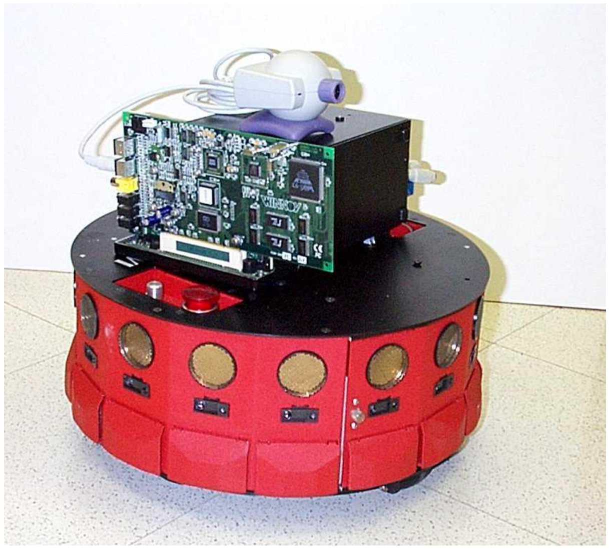
Fig. 13. Magellan robot manufactured by RWI.

Another company associated with RWI manufactured the Pioneer1 robot, shown in Fig. 14. It has a Hitachi microcontroller and several ultrasound distance sensors. Only a few sensors are present – this is a cost efficient robot. On the top there is a Sony Laptop – a cost efficient alternative for a PC104 computer for the lab environment. There is a PCMCIA network card in the laptop. Three lead acid batteries that are in the main robot compartment power all. That allows for operation between three and six hours.

The robotic manipulator used in the most recent projects utilizes Siemens 88C166 microprocessor that runs the Pioneer 2 Operating Systems (P2OS) from Active Media Robotics [22]. That system is interfaced to the on-board server computer via a serial link. P2OS provides low-level control for motors and actuators of the robotic manipulator, performs reading of sonar sensors, and prevents the motor overload. It also serves as a watchdog that stops the robot in case when serial communication is down for more than two seconds.

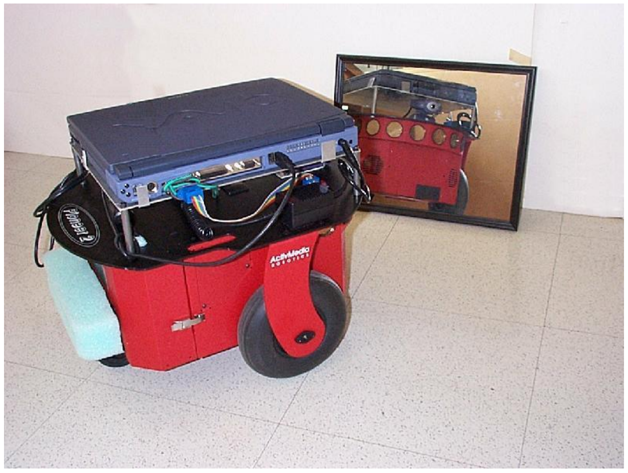
Fig. 14. Pioneer 1 robot with a laptop computer.

V. CONCLUSION

Several ideas for various projects that involve monitoring and control of a system using an embedded microcontroller and a computer were described. Some of those projects are expanded by adding additional component of controlling thorough the Internet.

Computer can be used directly to control up to eight actuators almost directly from a parallel printer port. The port by itself does not provide enough energy to control any actuator directly. A fast logic buffer gate circuit and external power supply is necessary. In case of controlling high voltage circuits, a reliable separation from the high voltage part of the circuit is mandatory. The current robotic manipulators have only basic functionality that allows only for movement. Two models were built using Lego sets and are powered by Lego motors. They are interfaced by a simple controller connected to a parallel port of the computer that runs the server.

Robots with serial ports require microcontrollers or microprocessors on the board. Such designs were validated with several experimental robots of various complexities that were developed as both student and sponsored research projects. Some of those projects are still available for testing to the public via internet.

The most recent project control server is located at http://pioneer1.bradley.edu/. One of the other, preliminary projects is still available at http://webbot.bradley.edu/. The tele-presence project is not available to the public. The home appliance system is available for a limited time at http://cdplayer.bradley.edu/. The projects are set up to work during the laboratory hours.

VI. ACKNOWLEDGEMENTS

Authors gratefully acknowledges participation and contribution of Bradley University senior students in several projects during 1998-2001: Pedro Febles-Cortez, Brian Davis, Theodore Konetski, Maria Perez, Jared Ramlal, Robert Harding, Gregory Harmon, John Kiolbasa, Jon Kujanski, Tim Booth, Jeffery Patrick, Sean Grady, Tim Booth, Sean Grady, Brian Huggins

V. REFERENCES

[1] Goertz, R., and Thompson, R., "Electronically controlled manipulator," *Nucleonics*, 1954.

[2] Ferrell, W.R., Sheridan, T.B., "Supervisory Control of Remote Manipulation," *IEEE Spectrum*, vol. 4, 1967, pp. 81-88.

[3] Bejczy, A.K., "Sensors, Controls and Man-Machine Interface for Advanced Teleoperation," *Science*, vol. 208, 1980, pp. 1327-1335.

[4] Ballard, R.D., "A Last Long look at Titanic," *National Geographic*, vol. 170, December 1986.

[5] Stark, L., Kim, W., Tendick, F., Ellis, S., Hannaford, B., et al., "Telerobotics: Display, Control, and Communication Problems," *IEEE Journal Robotics & Automation*, vol. RA-3, 1987, pp. 67-75.

[6] Yoerger, D.R., and Slotline, J.R., "Supervisory Control Architecture for Underwater Teleoperation," *Proceedings of IEEE International Conference on Robotics and Automation*, vol. RA-3, 1987, pp. 2068-73.

[7] Moray, N., Ferrell, R., Stassen, H.G., Yoerger, D.R., et al., "Supervisory Control: 30 Years and Counting," *Proceedings Of IEEE International Conference on Systems, Man and Cybernetics*, 1989, pp. 1185-86.

[8] Sheridan, T.B. *Telerobotics, Automation and Human Supervisory Control*, MIT Press, Cambridge, MA, 1992.

[9] Blackmon, T.T., and Stark, L.W. "Model-Based Supervisory Control in Telerobotics," *Presence*, vol. 5, 1996, pp. 205-223.

[10] Kim, W.S., Tendick, F., Ellis, S., and Stark, L., "A Compariosn of Position and Rate Control of Telemanipulation with Consideration of Manipulator System Denamics," *IEEE Journal on Robotics and Automation*, vol. RA-3, 1987, pp. 426-436.

[11] Buttolo, P., Kung, D., and Hannaford, T.B., "Manipulation in Real, Virtual, and Remote Environments," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-5, 1995, pp. 4656-61.

[12] Sweet W., Geppert L., "http:// It has changed everything, especially our engineering thinking", *IEEE Spectrum*, January 1997, pp. 23-37.

[13] Kaplan, G., "Ethernet's winning ways," *IEEE Spectrum*, January 2001, pp. 113-115.

[14] Bretz, E.A., "The car: just a Web browser with tires," *IEEE Spectrum*, January 2001, pp. 92-94

[15] Horowitz, E., "Migrating Software to the World Wide Web," *IEEE Software*, May/June 1998, pp. 18-21.

[16] Golberg, K., Mascha, M., Gentner, S., Rothenberg, N., Sutter, C., Wiegley, J., "Desktop Teleoperation via the World Wide Web," *Proc. of IEEE International Conference on Robotics and Automation*, May 19-26, 1995, Nagoya, Japan.

[17] Yeuk, F.H., Stark, L.W., "Design Simulation of a Web-Based Supervisory Control System," *Proceedings of International Conference on Web-Based Modeling and Simulation 1999*, 1999.

[18] Horowitz, E., "Migrating Software to the World Wide Web," *IEEE Software*, May/June 1998, pp. 18-21.

[19] Kumar, V., Korpi, M., Sengodan, S., *Telephony with H.323: Architectures for Unified Networks and Integrated Services*, John Wiley & Sons, 2001.

[20] Widjaja, I., Leon-Garcia, A., *Communication Networks: Fundamental Concepts and Key Architectures*, McGraw-Hill, 2000

[21] The Apache Software Foundation, http://www.apache.org/

[22] *Pioneer 2 Mobile Robots with Pioneer 2 Operating System Servers: Saphira Manual*, ActiveMedia Robotics, 2000.

[23] Malinowski, A., Konetski, T., Davis, B., Schertz, D., "Web-Controlled Robotic Manipulator using Java and Client-Server Architecture," *Proceedings of the Annual Conference of the IEEE Industrial Electronics Society (IECON'99)*, San Jose, CA, November 29-December 3, 1999, vol. 2, pp. 827-830.

[24] Malinowski, A., Dahlstrom, J., Febles Cortez, P., Dempsey, G., Mattus, C., "Web-based Remote Active presence," *Proceedings of the 2000 ASEE Annual Conference*, Saint Louis, MO, June 91-21, 2000.