II CONGRESSO NACIONAL DE ENGENHARIA MECÂNICA
II NATIONAL CONGRESS OF MECHANICAL ENGINEERING
12 a 16 de Agosto de 2002 - João Pessoa – PB

TELEOPERATION ARCHITECTURE FOR THE NAVIGATION THROUGH WEB OF THE NOMAD XR4000 MOBILE ROBOT

Sérgio Roberto Gonsalves Tourino
Departamento de Engenharia Mecânica
Faculdade de Tecnologia
Universidade de Brasília
CEP: 70910-900
Brasília/DF Brasil
tourino@graco.unb.br

Alberto José Álvares
Departamento de Engenharia
Faculdade de Tecnologia
Universidade de Brasília
CEP: 70910-900
Brasília/DF Brasil
alvares@alvarestech.com

***Abstract**. This work describes the implementation of a teleoperation architecture to the navigation of the Nomad XR4000 mobile robot using the Internet Protocol (TCP/IP) through WEB. The system developed is based on the client-server architecture, using a browser as a client. The graphical interface (GUI) is based on the Java programming language and HTML pages having programs developed on the C language as servers: video capture (video stream), pan-tilt control, obstacle detection (infrared and sonar sensors), dead-reckoning, battery level and motion control of the robot. A Fuzzy control for local sensorial data monitoring is used to minimize the effects of the communications delays from the TCP/IP protocol.*

***Keywords**:Teleoperation, mobile robot, Internet.*

1. INTRODUCTION

The existence of environmentally ill sites to humans, like nuclear power plants, brought the need to the development of remotely operated systems. The use of mobile robots to visualize and monitor dangerous places is growing due to the technological advances in computing and sensors. In these days there are several academical researches in the field of the use of Internet as a communication vehicle to remote control of machines, like mobile robots, as in the work of Nehmzow et all (1996) and Simmons (1998).

This works deals with the development of a teleoperation architecture to the Nomad XR4000 robot control through Internet by the use of a Java-based user interface in a ordinary WEB browser. Results are analysed and future improvements to the system are defined.

2. NOMAD XR4000 MOBILE ROBOT

The Nomad XR4000 robot is a fully integrated system designed to industrial or research purposes. It's features includes:

- Onboard dead reckoning system (odometry);
- Infrared sensors;
- Sonar sensors;
- Two PC Pentium computers running the Linux operating system (C language programming);
- Camera, pan-tilt unit and a famegrabber for vision tasks;
- Wireless network connection;

The internal odometric system has its calculations based in data got from encoders located at each actuator (motors in the wheels) and the kinematic model of the robot. The user can handle only the calculated data from the motion controller, and cannot get individual data from each encoder. The resolution of the odometric data is dependent mainly of the wheel slip in the floor. This data is accumulated over distance and the robot's odometric data error increases over its use.

Robot programming is done by using the XRDev architecture provided by the manufacturer. This multi-process architecture in composed mainly by three processes:

- Nrobot, the server process that communicates with the robot's hardware;
- Ngui, a graphical interface for robot's control;
- User processes, that communicating with the Nrobot can perform several tasks defined by the user.

The programming interface is composed by a set of C structures, as can be seen in the Fig. 1. These structures are manipulated by the user to command the robot to do its tasks, like go straight in the Y direction at 2 m/s or get the sonar sensors data.

The network connection is made by a Proxim RangeLan2 wireless Ethernet card adapter (http://www.proxim.com/). This system communicates with a bridge server that connects the robot with the local network in the laboratory. This bridge can handle up to 16 clients at a speed of 10 Mbps.

Figure 2 shows the network configuration for the XR4000 mobile robot. It can be seen that the brigde can establish several connections. In the laboratory there are two robots connected with the bridge, the XR4000 robot and the MRL 1.00 robot (Tourino, 1999).

```
struct N_RobotState
{
N_CONST long RobotID;
N_CONST char RobotType;
struct N_Integrator Integrator;
struct N_AxisSet AxisSet;
struct N_LiftController LiftController;
struct N_Joystick Joystick;
struct N_SonarController SonarController;
struct N_InfraredController InfraredController;
struct N_BumperController BumperController;
struct N_Compass Compass;
struct N_LasetSet LaserSet;
struct N_S550Set S550Set;
struct N_BatterySet BatterySet;
struct N_Timer Timer;
};
```

Figure 1. Main structure for programming the robot.

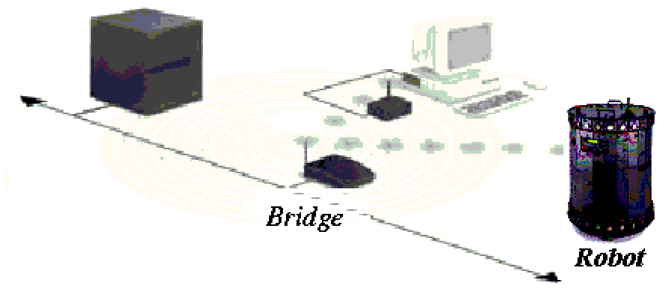


Figure 2. Network configuration for the XR4000 Mobile Robot.

3. TELEOPERATION ARCHITECTURE

The teleoperation architecture is based in the client-server architecture subset defined by Álvares (1998,1999a,1999b), that defines the following parts:

- the server, represented by the programs located at the mobile robot;
- the client, represented by the Java applets located at the user's browser;

This architecture can be seen in more detail in the Fig. 3.

Similar architecture was utilized in another work in the laboratory, the development of a mobile robot in Linux, in where Tourino (2000) presents the server-client architecture using as server a CGI-BIN program and a Java applet client.

The Internet represents the link between the servers (video and control) and the clients (video display, sensor feedback and commands from the user). The use of Internet as a communication device brings to the system some problems related to the data confidence and unexpected delays in the communication that will be discussed at a later section.

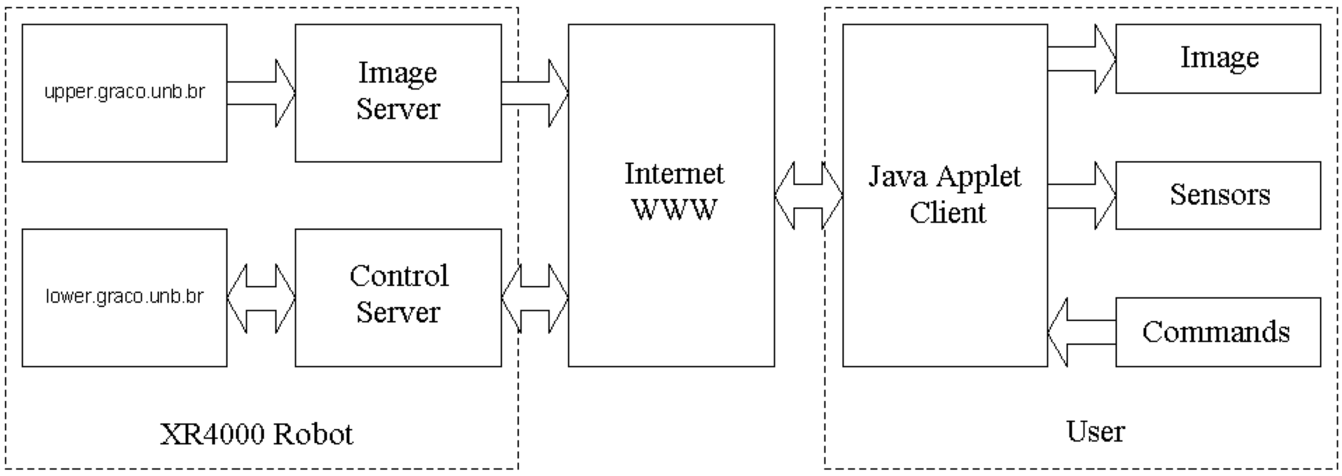


Figure 3. Teleoperation architecture used in the system.

4. ROBOT'S SERVERS

The robot's servers have the following purposes:

- control the robot's movement (RobServer);
- read and format the sensors data (SenServer);

- control the camera's pan-tilt (CamServer);
- provide the emergency stop button (StopServer);
- capture and format the captured images (JpgPush);

These servers communicate with the clients by a socket connection. Usually this type of connection is programmed by use of the BSD library (using commands like *socket*, *bind* and *accept*), but in this work the *inetd* server was used to cope with the connection and to make easy the programming effort, as stated by Arnett (1994). Its function can be understood by viewing the Fig. 4.

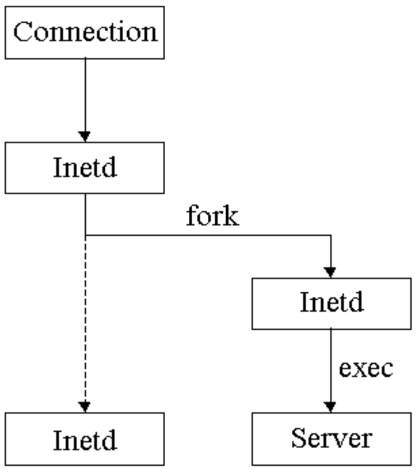


Figure 4. *Inetd* server system.

As can be seen in the picture, the *inetd* server waits for a remote connection in the defined service port and, when the connection arrives, it forks and sets the communications between the client and the specific server associated with that port. The server programming is based in the *stdin/stdout* syntax, simplifying the programming task (Stevens, 1993).

The Fig. 5 shows the relationship among the servers in the Nomad robot and the clients at the remote client (Netscape browser).

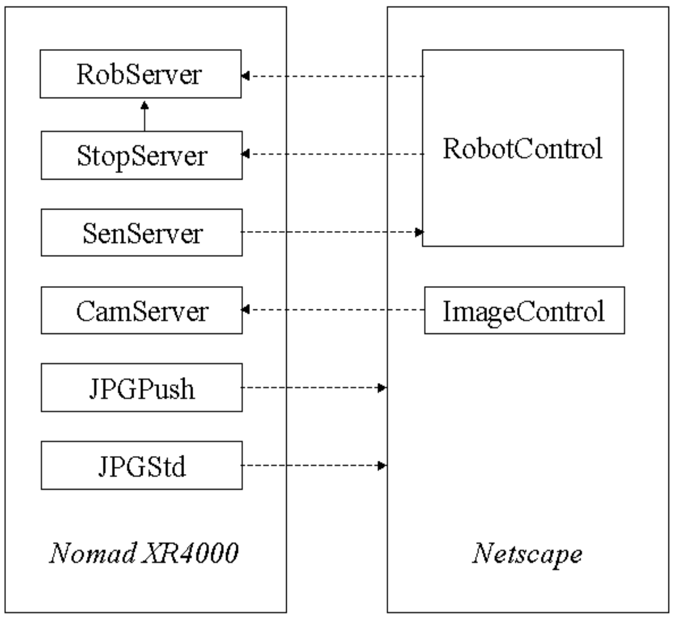


Figure 5. Servers and clients in the teleoperation architecture.

The following sections will define in detail the robot's servers.

4.1. RobServer

The RobServer server catches commands from the remote connection and drives appropriately the robot, using the vendor's provided library *Nclient*. The commands have the following general syntax:

command_id parameter

The *command_id* can be, for example, *distance*, and the *parameter* can be *1000*, indicating that the robot must walk for 1000mm in its next move.

The developed fuzzy controller (Saffiotti, 1999) gets data from the sonar sensors and the *distance* parameter to calculate the output speed (defining two input variables and one output variable). It was defined five fuzzy sets (triangular sets – see Fig. 6) to each variable, and twenty-five fuzzy rules were defined to generate the speed output. The centroid method of deffuzyfication was used, and the result is sent to the robot's controller after each calculation loop.

The RobServer also waits for a signal from another process (StopServer), that causes the immediate halt of the robot move.

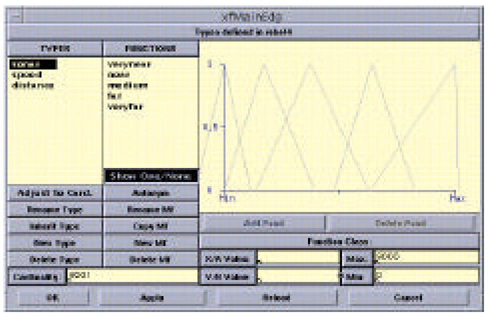


Figure 6. Fuzzy sets for the sonar variable.

4.2. SenServer

The SenServer server gets data from the sensors (sonar and infrared), from the internal odometry system, and the battery charge status. Then it formats these data and send them to the remote client. The data is sent by request of the remote user (automatically handled by the Java applet). The commands have the following syntax:

get parameter

Where *parameter* is the sensor to be read (like *sonar*).

4.3. CamServer

The CamServer server is used to control the pan-tilt pose of the robot's camera, giving to the user the flexibility to change his viewpoint and gets better information from the remote environment. The server waits for commands from the remote client, that must have the following syntax:

command_id parameter

The *command_id* can be, for example, *direction*, and the *parameter* can be *up*, indicating that the pan-tilt unit must move to let the camera seeing to the top.

4.4. StopServer

The StopServer module is a simple server used to provide a secure command to stop the robot, when the user sees danger that the robot's autonomy could not cope with. The server simply sends a signal (through the *kill* call) to the RobServer when its service is requested.

4.5. JpgPush

The JpgPush server has three main purposes: get the image from the framegrabber, translate this image to the JPEG format, and format the resulting image to the HTTP-IMG format to be handled by the browser (Netscape).

The framegrabber is a Matrix Meteor card, whose Linux driver was developed by Sutton (1999), can get 3 images/sec in this implementation. The low frame rate is necessary due to problems noted with the server and the browser capabilities. The server uses the *ioctl_meteor* library, defining the image size, the image color depth and the type of grabbing (memory mapped grabbing).

The translation from rough image to the JPEG image format is done by the use of the *jpeglib* library from the Independent JPEG Group - IJG (1998). This library simplifies all the job and the server programmer just need to define the compression factor (it was used 30 in this implementation).

The final step to send the images to the client is to format them into the *push-server* format, from Netscape (1999). This is accomplished by the use of the following syntax:

Content-type: multipart/x-mixed-replace;boundary=JPEGImage

This sentence instructs the browser to wait for incoming images and display them over the older image, giving to the client user a type of image animation.

The program obtained is a new “webcam” server for the Matrix Meteor card in Linux systems that can send images to Netscape compatible browsers. The software can be downloaded at the address http://jazz.graco.unb.br/alvares/XR4000/webcam_meteor_ matrox.tar.gz

5. USER INTERFACE IN JAVA

The user interface (client side) was developed in the Java programming language, created by the Sun Microsystems (1999). The GUI consists in two Java applets (a special program to run in a browser): RobotControl, for controlling the robot's move, and ImageControl, for the control of the pan-tilt pose. The system can be accessed at the address http://xr4000.graco.unb.br.

5.1. RobotControl

The RobotControl applet consists in three main parts: the control buttons, the setup dialog box, and the feedback graphs.

The control buttons are used to control the robot's move: four of them to control the translation move, and two of them to control the rotation move. The central button, named Stop, is used to call the StopServer server.

The setup dialog box is used to define the rotational angle and speed to be moved in rotational moves, and to define the translational distance to move. There is not way to define the translational speed, since this data is automatically defined by the fuzzy controller in the RobServer server.

The feedback graphs are used to give to the user data feedback from the robot. In the upper there is a round ring representing the data received from the sonar or the infrared sensors, and in the lower there is a voltage meter indicating the voltage level from the robot's batteries. At the applet's bottom there is a text line indicating the actual position and orientation based on the internal odometric data.

The RobotControl applet can be viewed in the Fig. 7:

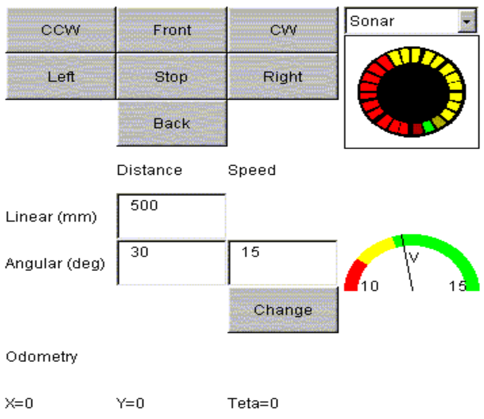


Figure 7. RobotControl applet.

5.2. ImageControl

The RobotControl applet consists in two main parts: the control buttons and the feedback graphs.

The control buttons are used to control the camera's pose. The user can control only the pan move, only the tilt or both of them. The central button is used to go to the zero position, in which both the pan and the tilt values are set to zero degrees.

The feedback graphs are used to give to the user visual feedback from the pan-tilt unit. The client can see the pan and the tilt orientation relative to the robot, increasing the sense of the camera's pose relative to the robot.

The ImageControl applet can be viewed in the Fig. 8:

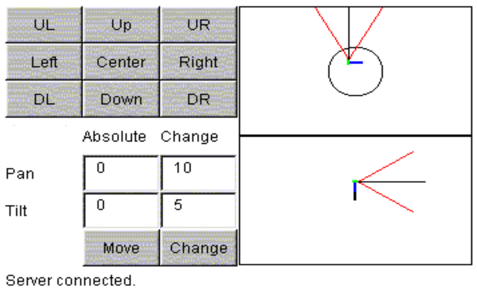


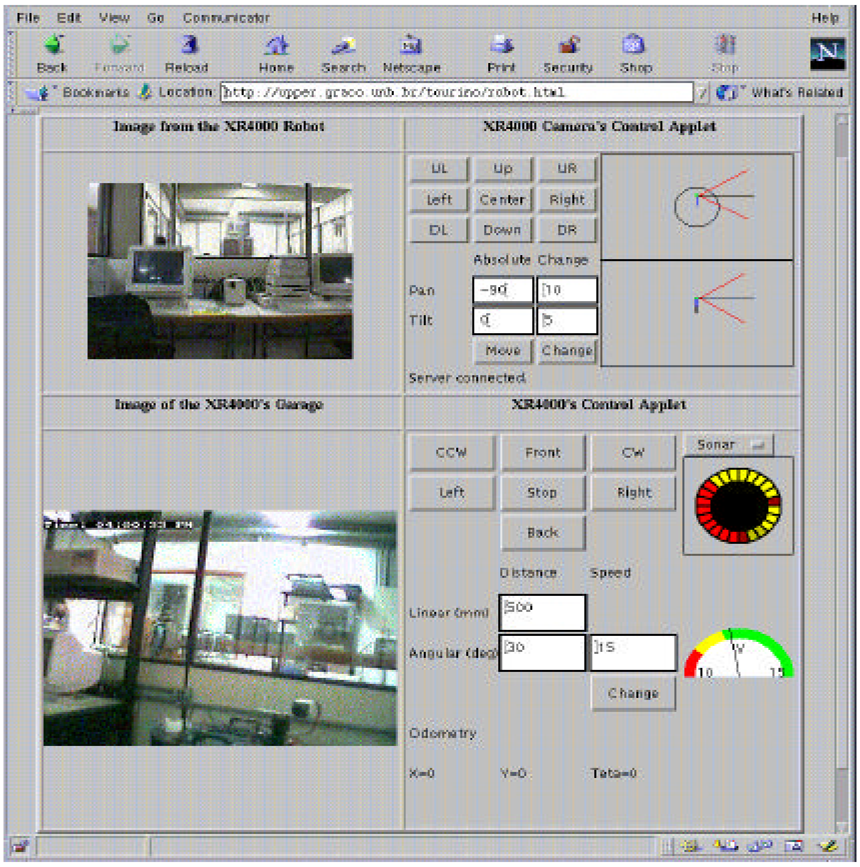
Figure 8. ImageControl applet.

6. RESULTS

The fully implemented interface of teleoperation can be viewed in the Fig. 9, captured from the Netscape browser at a Linux machine.

In the upper-left section of the interface there is the images captured from the JpgPush server, showing the remote environment to the user. In the lower-left section there is an image captured by an external camera, showing the Nomad XR4000 robot at its docking place. At the upper-right section can be viewed the CameraControl applet, that permits to the user control the pan-tilt system to get a better understanding of the remote site. At the lower-right section is viewed the RobotControl applet, that is responsible for the control of the robot's move and feedback data to the user.

The system was tested through the use of the factorial design method (Box, 1978) to detect the important variable effects over the fuzzy controller response. There were sixteen runs for a 2^3 full factorial design (replicated) including the following variables: type of environment (indoor/ outdoor), sonar sensor (on/off) and acceleration (low/high). It was verified that the sonar state (on/off) is the main source of speed variation of the fuzzy output. This occurs due to the unexpected wave reflections in the environment, that causes a stochastic data readings from the sensor. In spite of this problem, the fuzzy controller is robust and permits to the user a secure teleoperation control. At this time there is not reported collisions of the robot in its environment.


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks
What's Related
Image from the XR4000 Robot
XR4000 Camera's Control Applet
UL Up UR
Left Center Right
DL Down DR
Absolute Change
Move Change
Image of the XR4000's Garage
XR4000's Control Applet
CCW Front CW
Left Stop Right
Back
Sonar
Distance Speed
Change
10 15

Location: http://upper.graco.unb.br/tourino/robot.html

Pan -90 10

Tilt 0 5

Server connected

Linear (mm) 500

Angular (deg) 30 15

Odometry

X=0 Y=0 Teta=0

Figure 9. System´s homepage.

7. CONCLUSION

The teleoperation technology using Internet is in your beginning, and it have few applications due to its low data flow nowadays. In this work was presented the development of a guiding system to the Nomad robot. This teleoperation system uses as communication tool the Java programming language and the robot's servers. The system presents to the user images from the remote site as well as feedback from the robot's sensors, and permits to him the control of the robot and the pan-tilt unit. Due to the communication speed restrictions, the images are compressed to the JPEG format, reducing the bandwidth use and improving the user's view. The need to a secure control system and due to the unexpected delays in the data transfer required the development of an speed controller based on the fuzzy logic. This controller is the intelligence core of the system, using environmental data captured from the sonar sensors. The fuzzy controller was tested by the fractional design method, and was determined that the main cause of problems is the stochastic data received by the sonar sensors. In spite of this problem, the system shows robust and no collisions were reported up to now.

The implemented interface has been tested by several months in the laboratory, and shows up to now very robust and operational. The problems related to the sonar sensors could be handled by the use of a type of data filter, like Kalman filtering or another useful algorithm. Future improvements in the system includes static and dynamic mapping of environment to permit a full autonomous navigation of the robot, reducing the user assistance.

8. BIBLIOGRAPHY

Álvares, A. J. Romariz, L. S. 1998. "Desenvolvimento de um manipulador com dois graus de liberdade controlado remotamente via Internet". V CEM-NNE98. Fortaleza.

Álvares, A. J. 1999a. "Telerobótica : Metodologia Para o Desenvolvimento De Sistemas Robóticos Teleoperados Via Internet", Águas de Lindóia, SP, XV COBEM

Álvares, A. J. 1999b. "Telerobotics: A Methodology for the Development of Through-the-Internet Teleoperated Robotic Systems" Telemanipulator and Telepresence Technologies V – SPIE Boston, USA

Arnett, M.F., Dulaney, E., Harper, E. 1994. "Inside TCP/IP". New Riders Publishing. Indianapolis. USA.

Box E.P., Hunter W.G., 1978. "Statistic for Experimenters: An introduction to design, data analysis, and model building." ,Wiley, New York, USA

Nehmzow, U. Bühlmeier, A. Dürer, H. 1996. "Remote Control of Mobile Robot via Internet" Manchester.

Netscape. 1999. "An Exploration of Dynamic Documents". Available in Internet by WWW. URL: http://www1.netscape.com/assist/net_sites/pushpull.html

Saffiotti A., Ruspini E., Konolige K. 1999. "Using Fuzzy Logic for Mobile Robot Control". Int. Practical Applications of Fuzzy Technologies. pp 185-206.

Simmons, R. 1998. "Xavier: An Autonomous Mobile Robot on the Web". Carnegie Mellon University. USA.

Stevens, W. R. 1993. "Advanced Programming in the UNIX Environment". Addison Wesley Longman. USA.

Sun Microsystems. 1999. "The Java Tutorial". Available in Internet by WWW. URL: http://java.sun.com/docs/books/tutorial/index.html.

Sutton, M. 1999. "Matrox Meteor FrameGrabber Driver for Linux v1.5.4". Available in Internet by FTP. URL: ftp://ftp.rwii.com/pub/linux/system/Meteor/meteorman.html

The Independent JPEG Group. 1998. "JPEG Software". Available in Internet by FTP. URL: ftp://ftp.uu.net/graphics/jpeg/

Tourino, S. Álvares, A. 1999. "Navigation Algorithm for Autonomous Mobile Robots", CREEM 1999. Brasilia.

Tourino, S. Álvares, A. 2000."Desenvolvimento de um Robô Móvel Autônomo Teleoperado Via Internet" , CONEM 2000, Natal.