

A Web-based Telerobotic System for Research and Education at Essex

Lixiang Yu, Pui Wo Tsui, Quan Zhou, Huosheng Hu

Abstract--This paper describes steps toward building a web-based telerobotic system for both research and teaching in the Essex University. The system has standard network protocol and interactive human-machine interface. Using a Web browser, a remote operator can control and/or program a mobile robot to navigate in our laboratory while receiving visual feedback and simulated local perceptual map via the Internet. The employment of an intuitive user interface enables both researchers and students to control and program mobile robots and to do some experiments remotely.

Keywords: Telerobotics, WWW, Mobile robots

I. INTRODUCTION

Telerobotics has traditionally been implemented using a two-way communication link between the controller and the remote robotic system. This communication link is normally dedicated and private. When the controller and the robot location are determined, a private communication link should be built up between them before any operation. On the other side, the robots are just used by some specialized researchers during work time, in most time, they are locked in the labs without any attention. While many other researchers and students have no chances to access these expensive telerobots.

The rapid growth of the Internet in recent years have attracted the interests of many researchers to embed (intelligent) devices into it. Although Internet robotics or Web-based telerobotics is still a relatively new research area, it opened up a new range of real-world applications, such as on-line open laboratories and tele-training. Some demonstrations of telerobotic projects are now available on the Internet [5].

The goal of this project is to build a networked telerobotic system so that the Internet users, especially researchers and students, could control and/or program the mobile robot to explore the unknown environment remotely from their home and share this unique robotic system with us. At the same time, we also can collect test data for our own research works.

In this paper, the analysis of some related telerobot projects is given in section 2. Our project goal is briefly outlined in section 3. The main research issue, a framework for the network telerobotics system, is described in section 4. Some experimental results are presented in section 5, including remote operation and remote programming for autonomous navigation. Finally, section 6 presents a brief conclusion and future work.

II. RELATED WORKS

Since the first networked device "Cambridge coffeepot" appeared in the Internet, the rapid growth of the WWW had resulted in a growing number of telerobotics sites and web accessible devices in the Internet.

One of the earlier telerobot sites in the Internet is the Telerobot in Australia [3], which has been online since 1994. In their teleoperation system, the user can remotely operate a six degrees-of-freedom (DOF) manipulator in the University of Western Australia in Perth. Once in control, the user can pick and place wood blocks. In the previous version of their interface, the web server will feed back a still image and robot arm position after one action was executed. The later interface uses Java applet, providing a stream of captured images and a better layout.

Another teleoperation system on the web, in which most of our initial work based on, is MAX [1], developed by the Network-Centric Applied Research team (N-CART) at Ryerson Polytechnic University in Canada. They proposed some interesting view on building the web-based telerobot site, such as wireless operation, intuitive user interface and inexpensive construction. The user can control the mobile robot to move in a constraint area with smooth visual feedback.

Recently, the research began to focus on autonomous robots that can be used in more complicated environments, i.e. the real-world applications. The Khep-On-The-Web telerobotic project in Switzerland was developed with a long-term goal of enabling the sharing of expensive or unique equipment [4]. The user is challenged to control a mobile robot to navigate a maze. It provides a streaming of live video feedback for the user. Please refer to [5] for a more complete listing of the web sites with similar project demonstration.

Most of the successful teleoperation systems on the Internet share some common characteristics and goals, which enable them to withstand long-term usage, including intuitive user interface, easy control and maintenance. However, most of the networked telerobots implemented so far are fixed robot arms [3], operating within a very structured and limited environment. Only few of telerobots on the web are mobile robots, but they are mainly tele-operated systems [1][2], with very limited or even without good autonomy. To get real unbound interaction with a distant environment, a mobile platform with high degree of autonomy is needed for the real-world applications such as the laboratory work [6] and the museum tour guide [2]. In most case, the user involved into the interaction passively, and can only do simple remote control of the robot and very limited teleoperation.

Department of Computer Science, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK. Email: {lyut, pwtstui,qzhou, hhu}@essex.ac.uk

III. PROJECT GOAL

The aim of our research is to build a networked telerobotics system for both teaching and research. The system combines network technology with intelligent mobile robots. In this way, Internet users such as researchers and students could control and/or program our mobile robot to explore our Laboratory remotely.

At the first stage, the main issue is to design and build a basic teleoperation system framework, i.e. a test-bed for testing theories and ideas on the tele-operated control of mobile robots. The framework should provide the following features:

- ❑ Uniform interface for easy integration of different robots into the system's framework.
- ❑ Intuitive user interface and adequate feedback.
- ❑ Low cost and easy for maintenance.
- ❑ Easily extendable for more complex functionality.
- ❑ Allow the user to submit their simple program to the server and test it in the real robot remotely.

The first stage of our research effort has been made to realize some of these goals.

IV. THE FRAMEWORK

With the rapid growth of the Internet there are many communications technologies available to execute requests in a networked environment. Currently the most widely used web browser is the Hyper Text Transfer Protocol (HTTP). It can be executed with the Communication Gateway Interface (CGI) for remote control, which is one of methods used in many web-based telerobot systems [3][6]. Through the HTML form, a request can be passed from client to server, a process will be launched to perform some predetermined actions in the server, and a dynamically generated HTML page will return the results to the client. But CGI has a number of limitations such as its slow response speed. Moreover, a complete HTML page must be generated with each request while this resulting page is still static. So it is not suitable for real-time remote control.

In contrast, Java provides the capability to implement network connections and thus avoid the limitations of CGI. Java applet can operate within the browser and hence is accessible by most computers on the Internet. Rather than being static, Java applet also enables an interface to dynamically change its content due to the fact that Java applet is executable within a web page. However, the Java execution must make connections to each server in the system. So the client must know the location of all servers and other clients it want to connect, which can quickly become unmanageable as the number of servers and clients increases dramatically. There are also security restrictions associated with Java such as Java applets that can connect to the host they were served from.

A more flexible and extendable approach is to use the central server architecture shown as figure 1. All the clients and servers are connected to one central web server, they only need to know the location of the web server and

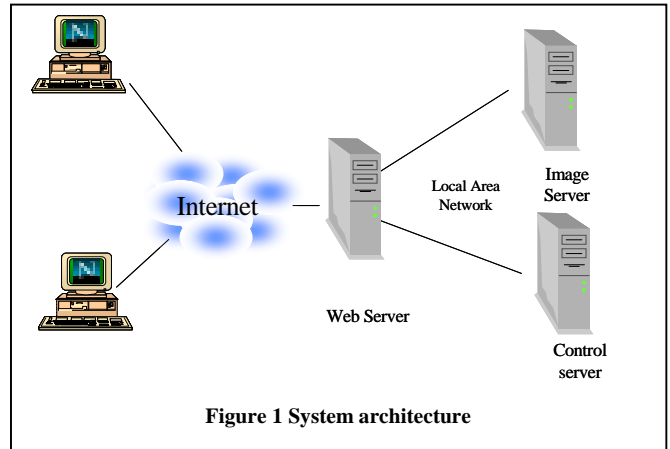


Figure 1 System architecture

communicate with each other through the web server. With this architecture, we can either put all image service, robot control service and web service in one computer or put them in several computers and connect them with TCP sockets. This architecture is also very easy to add more computers for robot control and image processing or for multi-robot control.

A. Hardware

The configuration of our system hardware is shown in figure 2. The host computer communicates with the mobile robot via a radio modem connected to serial port. The video signal is captured by the frame grabber based on bt848 chipset. The host computer is connects to the network with standard Ethernet card.

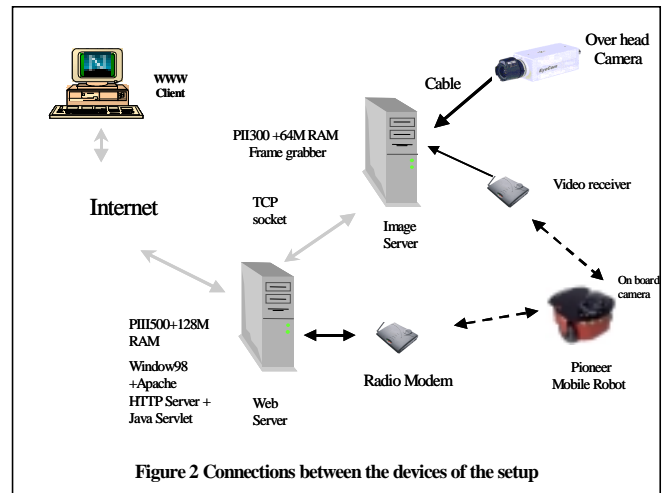


Figure 2 Connections between the devices of the setup

The Pioneer mobile robot produced by ActivMedia is powered by two reversible DC motors coupled to two wheels with a diameter of five inches (12.5 cm). They are equipped with eight ultrasonic sensors in which one is on each side and other sensors are forward facing. The data produced by these sensors is to be used to build a simulated local perceptual space of the mobile robot, which is presented at the client site. Several cameras are used in the system to provide the visual information or the robot from different site, An on-board camera that connected to the server though the wireless video link is placed on the front-top of the mobile robot in order to give user a clear view of

the environment in front of the robot. The others are overhead cameras that feedback a global view of the test site to the remote user from different angle..

B. Software

The web server used is Apache HTTP web server working on the Windows 98 platform. Whole system consists of several independent modules for custom service, and each of them includes a server-side program and client-side applets. These modules are named as the robot control module, the visual feedback module and the virtual representation module. The Java Servlet in the web server (Apache) handles the communication between the clients and the server except for image transmission where TCP/IP socket is used. Their relationship is shown in figure 3.

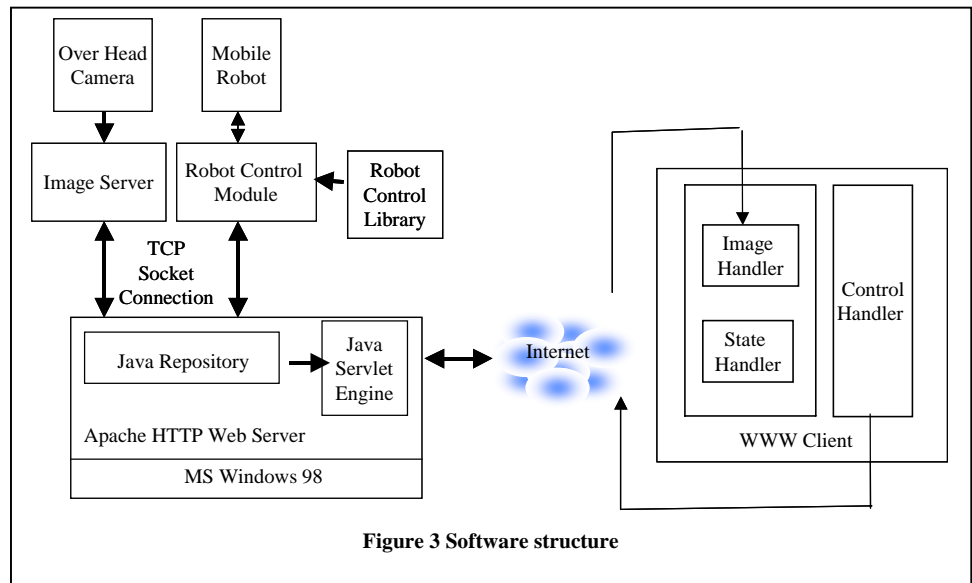


Figure 3 Software structure

Robot States				Sonar 1 info				Sonar 2 info			
x	y	th	speed	x	y	Range	valid

Figure 4 Message structure for data feedback

1) Robot Control Module

The robot control module is responsible for controlling the mobile robot. Since the control program of the Pioneer robots was implemented with C++, it is necessary to build an interface to Java program. Therefore, JNI (Java native Interface) is used to interface a DLL (Dynamic Link Library) file implemented with C++. No intelligence was integrated into the module at this stage, and only some basic motion commands such as forward, recede, change speed and direction and set speed are integrated.

The Java program will run continuously once the system starts. It receives commands sent from the client through a socket, and controls the movement of the mobile robot through the radio modem connected to the serial port. Only one user can control the robot each time, the other users have to wait in queue until the current operator left. The steps for handling the client's request are as follow:

At the same time, this Java program will feed back the robot information and the sonar readings to the clients every 100ms. In order to reduce the transmission time, all information were combined to form a string shown in figure 4, and sent to all the clients that connect to the server. This string will be interpreted at the client side to display an environment map and the necessary robot status.

```

Wait connection
While (request != bye)
{
    Receive request
    Check request
    Call related function in DLL to
    execute the command
}
Close connection

```

2) Visual Feedback

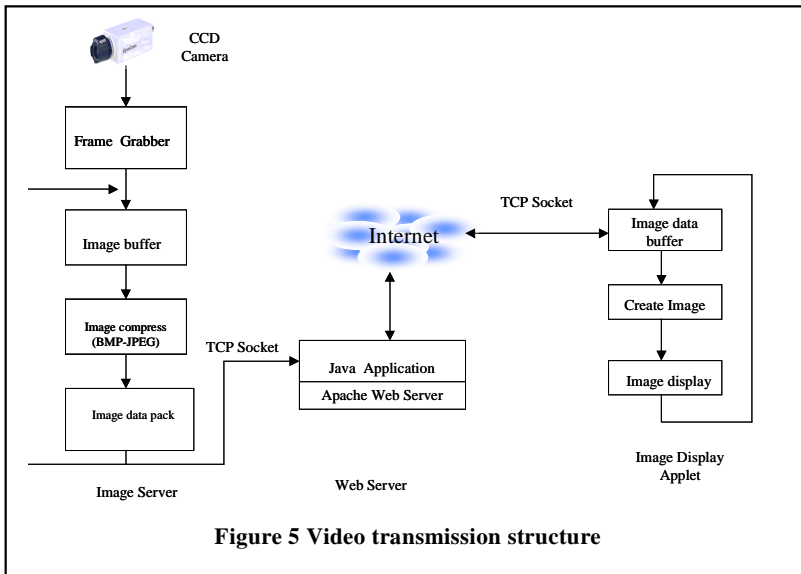
The continuous and steady image stream feedback from the robot site is necessary when the Internet users control the mobile robot at the client site. Moreover, the image quality should be good enough in order to provide as much information as possible about the remote site for teleoperation. The steps are as follow:

```

Create a new byte array
Run forever
{
    Receive data from the socket
    Wait until one image frame was receive
    Stored in the byte array
    Create a new image from the byte array
    Wait until the image was created
    Draw the image.
    Clear byte array buffer
}

```

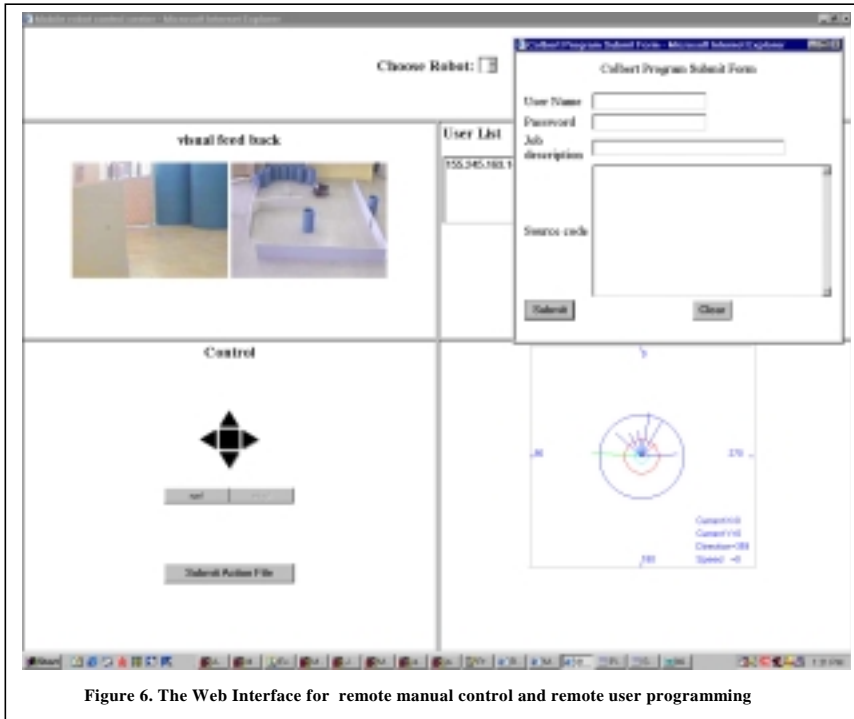
Most of the programs use server push technology, where the video was made up from a stream of still images, and sent by a Java program(server) to a Java applet(client) via a socket, and interpreted by the applet in either GIF or JPEG format, as shown in figure 5. In this system, the images are captured from the frame grabber based on bt848 chipset and compressed to JPEG format by software implemented in C++. Then, these image packages are sent from the image server to the web server through a socket. The Java program would stream these JPEG images to all the clients that are connected to this web server at a fixed interval. On the client side, the Java applet will recreate the image when it receives an entire frame and displays it.



C. Web Interface

The user interface is designed with the intention of making it easy for researchers and students to interact with the mobile robot. We design a simple interface that provides as much information as possible for teleoperation. This user interface consists of several Java Applets as shown in Figure 6. It can work on any web browser that supports Java1.2 or above. There is an on-line instruction is supplied with this control console.

The control panel is made up of four direction buttons and one stop button at this stage. The user can select a robot from the robot group (we just connect one robot into the Internet at present), and directly control the mobile robot by click the button on the control panel. If they want to test their own program,



they can click the “Submit Action File” button, one “Colbert program submit form” will pop up, the user can fill in the form and submit it. Then click the “RUN” button, the program will be loaded in to system and start to work. The user can stop the program at any time during the operation by clicking the “Remove” button. The navigation buttons are still functional during the process. The visual feedback is shown in continuous JPEG image with 200x150 pixels at 24-bit colour depth. VLPS applet displays some basic information about the mobile robot and the remote environment by analyzing the data gathered by the mobile robot. The user can find the obstacles near the robot and the trajectory, the current position and the speed of the mobile robot.

With this intuitive user interface, one user can control and program the movement of the mobile robot from the web browser according to the information get from this visual feed back and VLPS. The other users just have the visual

3) Virtual local perceptual space (VLPS)

The virtual representation module is the Applet that work on the web page to handle the information string sent by the robot control module. The virtual world model draws an icon representing the robot position and the direction, as well as the trajectory at the specific coordinates based on the internal odometer. each sonar’s reading will be represented by a beam according to its range and direction. if one sonar reading is detected to close to the robot. It will change to different color, and be updated every 100ms. With this virtual environment map, the user can find suspected obstacles nearby, and help to make correct decision when visual feedback suffering serious time-delay or obstacles beyond the camera’s scope.

feedback and a virtual map at the same time, and have to wait in queue until the previous user left and got the control right automatically according to the “first arrive, first served” rule. The user can find out how many people are currently viewing/waiting to control the system from the user list.

V. EXPERIMENTS AND PRELIMINARY RESULTS

During the project development, different configurations were tested in different environments. The aim is to develop a more reliable system framework that can be used in the real world. One Pentium III 500+128M RAM PC resides at a static IP address works as the web server and the robot control server. Another AMD-K6-300+64M RAM PC

nearly runs under Windows98 and works as a video server for image processing.

Remote Control

Some preliminary results have been obtained from the tests. There is no local intelligence on the robot such as obstacle avoidance at this stage. The user can control the mobile robot freely. The advantage of having a direct control is that the user can see the result of his/her own action without any external contribution. The drawback is that control of the robot is more difficult in a complex environment without help or under the serious time delay. Therefore, more autonomy should be put on the robot for better control and autonomous navigation at next stage.

The overhead camera can give a good global view of the test site, which is very useful when the mobile robot has limited sensing capability. The users can see the movement of the mobile robot. But it also restricts the view of the robot in a small area. The on-board camera that connects with the server through the video link enables the robot to move in a quite large area and only restricted by the transmitter’s range. The user can see what the mobile robot see. But the disadvantages are that the user just can see the front site of the mobile robot and can not see the other sides of the robot and itself. This is often very disorienting, especially at the time-delay and the low updating rate. A possible solution is to use both the on-board camera and the overhead camera, or a controllable on-board camera. Figure 7 shows an example test run of robot performing task by remote manual control. Where the user was challenged to control the mobile robot move from the entrance at the lower left hand side of the environment to the upper left of the environment.

In the test, we achieve 9 to 12 frames per second in the local area network (10-BaseT Ethernet) with time delay less than 200ms and 7-8 fps when the test was carried out from other universities in the U.K (T1 network). Each frame has 200x150 pixels (medium image quality). The comparison of the image transmission speed in different projects shown in table 1. The speed ensures a quite reliable view in most experiments. Although the bigger image size and the high updated rate can give more information, the network

Project name	Test environment	Tech.	Image size	fps
MAX[1]	Internet	Server push	320x240	~7
EPFL[7]	LAN	Server push	200x150	10-15
HIVIEW [8]	JAENET	Video conference	176x144	~8
IntMedium [9]	28.8k modem	Video conference	176x144	6-7
Essex	Internet	Server push	200x150	7-8

Table 1 Comparison of image transmission speed

bandwidth and the performance of the computer restricted the video transmission and recreate speed.

Since all JPEG images were compressed in software, the video server can only generate about 13 JPEG frames per second although the frame grabber can capture 50 frames

every second before compression. The variable network traffic also limits the packet size that can be sent in the Internet. Since transmission bandwidth is a product of the image resolution and the frame rate. There should be a mechanism to control the image resolution and the updating

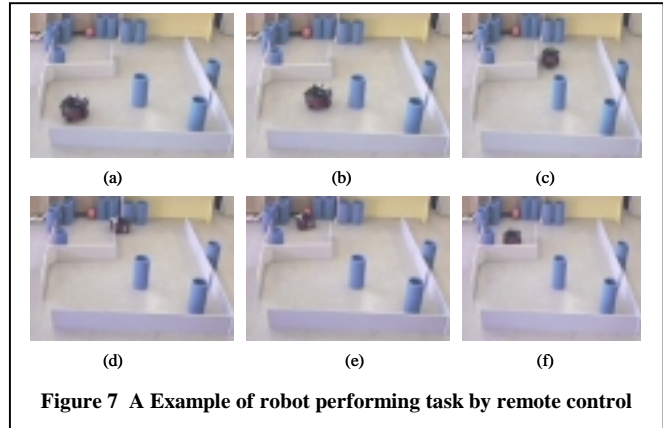


Figure 7 A Example of robot performing task by remote control

rate. For example, we can increase the image quality while reducing the updating rate if we need a clear view; or, we can increase the updating rate while reduce the image quality when avoiding collision.

The VLPS is built based on the sonar readings and the internal odometer of the robot. An environment map provides much help when the mobile robot had to avoid collision. But, due to the perceptual limitations, the sensor noise and the odometry errors accumulated over time, it is impossible to build an accurate world model. This environment map just can be used as short-term environment map at this stage. A reference must be introduced into the system if we want to build a correct environment map.

There is no any algorithm to control the access time for each user. So all the users had to wait in queue until the first user left. A possible solution is to allocate each user the same control time slices or the robot only execute the command selected by voting from all Web users.

Remote Programming

In the 2nd stage, we have modified and extended the system to include the mechanism for transferring robot control scripts to the telerobot for execution. This would enable the students to program the telerobot to perform activities defined in the transferred scripts for testing their programs to solve a given task.

A web form is presented to the registered user to input the robot control script as shown in figure 6. The syntax of the script follows the Colbert language available in the Saphira robot library [11]. The completed script will then be submitted via the Internet to the web server in our laboratory for processing. The Colbert evaluator is called to evaluate the script and then the script is executed by the telerobot. The student is able to observe the telerobot execution of the robot control script through the web interface and stop it if necessary. The steps below give an overview of the whole process:

- Step 1: Fill in web form
- Step 2: Submit script for source code
- Step 3: Verify script content
- Step 4: Load into Colbert Evaluator
- Step 5: Execute user script
- Step 6: Delete script that has been tested
- Step 7: Back to Step 1(Ready for next script)

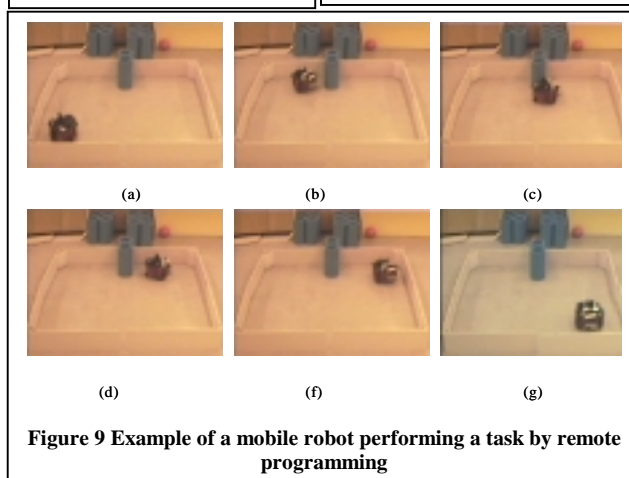
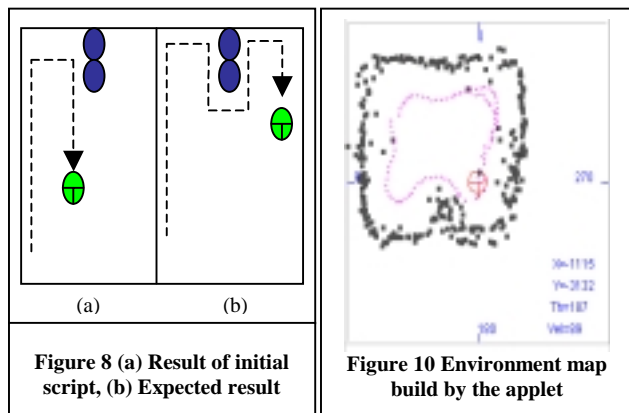


Figure 9 shows an example test run of the robot performing wall following task by remote programming, and figure 10 shows the resulting feedback during the experiment. In the first experiment, there are no obstacle in the environment and the robot control script executed successfully without any difficulties. In the subsequent experiment, obstacles were put into the environment. Ideally, the robot should navigate around the obstacle as shown in figure 8(b). The robot running the initial script failed, as shown in figure 8(a). During the process of improving the script, sometimes the robot ran into the obstacle and fail to recover (i.e. free itself). We are able to use the remote manual control to free the robot from the situation. This mixture of manual and behavioral control shows that the telerobotic system, is more flexible than using only remote manual control or robot program. It provides a more interactive way for the user to control and test the telerobot remotely.

VI. CONCLUSION

A modular framework for networked telerobotics system and its web interface has been presented. The system enables Internet users, researchers and students, to control

and program our mobile robot from the web browser remotely. This framework has been tested in our Brooker laboratory, and the preliminary result looks promising. The system is extensible to include more mobile robots and connect more video cameras. The visual feedback module written in Java allows for fast image updating, and presents a quite reliable view for the Web user. The control module allows the user to directly control or transfer robot control script to the mobile robot. The VLPS provides a short-term world model of the environment.

There are still some problems to be solved before the remote exploration of an unknown and complex environment through the Internet becomes reality. The next step of the research will be testing the system in a class room environment where students will be ask to use the system to solve some predefined tasks. A survey will be conducted to get the feedback from the students, which will help improve and guide our future research directions. On the development side, we will focus on how to provide a telerobot with higher degree of local intelligence (e.g. learning, memory) to handle uncertainty, and integrating multiple mobile robots into the system to achieve redundancy and robustness.

References

- [1] A. Ferworn and K. Plataniotis, "Effective Teleoperation over the World Wide Web", IASTED Int. Conference on Robotics and Applications, California, October 1999. <http://max.scs.ryerson.ca/>
- [2] D. Schulz, W. Burgard, D. Fox, S. Thrun and A.B.Cremers, "Web interface for Mobile robots in Public Places", IEEE robotics & Automation Magazine, March 2000 <http://www.informatik.uni-bonn.de/~rhino/>
- [3] K. Taylor and B. Dalton, "Internet Robots: A New Robotics Niche", IEEE Robotics and Automation Magazine, March 2000, <http://telerobot.mech.uwa.edu.au/>
- [4] P. Saucy and F. Mondada "KhepOnTheWeb: Open Access to a Mobile Robot on the Internet", IEEE Robotics and Automation Magazine, Vol. 7, No. 1, ISSN 1070-9932, March 2000. <http://diwww.epfl.ch/lami/team/michel/RobOnWeb>
- [5] NASA Space Telerobotics, "Real Robots on Web". http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html
- [6] R. Simmons, "Xavier: An Autonomous Mobile Robot on the Web", IROS'98 Workshop on Robots on the Web, Victoria, B.C. Canada, 12-17 October 1998. <http://jubilee.learning.cs.cmu.edu:8080/>
- [7] R. Siegwart, C. Wannaz, P. Garcia, R. Blank "Guiding Mobile Robots through the Web" IROS'98 Workshop on Robots on the Web, Victoria, Canada, October 1998.
- [8] S. Lavington. "HIGHVIEW: High-quality Resilient Video Over Existing Networks", Department of Computer Science, University of Essex, 1998.
- [9] S. You, T. Wang, Q. Zhang "Internet-based Multimedia Interaction in Teleoperated Robotic System" Robotics Institute, Beijing Univ. of Aeronautics & Astronautics, 2000
- [10] T. Sheridan, "Telerobotics, Automation, and Human Supervisory Control", MIT Press, Cambridge, Massachusetts, 1992.
- [11] K. Konolige, "Colbert: A Language for Reactive Control in Saphira", <http://www.ai.sri.com/~konolige/saphira>