Multirobot Internet-Based Architecture for Telemanipulation: Experimental Validation*

R. Marín P. J. Sanz P. Nebot R. Esteller
Universitat Jaume I
Av. Sos Baynat, s/n. E-12006 Castelló - SPAIN
{rmarin, sanzp, al065289, esteller}@uji.es

Abstract - *The design of online robots is a difficult task that must take into account multiple aspects like for example: (1) The System Architecture, (2) The User Interface, (3) Communications, (4) Time delay, (5) Limited Bandwidth, etc. Besides this, taking into account that the web permits multiple people to easily access a single system, methods for managing the control of a unique robot must be designed (e.g. off-line virtual interfaces, connecting several robots to the system, etc.).*

In this paper we present the system architecture that have been used in our laboratory to let users program remotely two educational and two industrial robots through the same web-based system. Experimental results show different alternatives to organize the architecture and focus on the system performance aspects.

Keywords: Telemanipulation, TCP/IP distributed systems, Robotics TeleLabs, Remote Programming, and Education & Training.

1 Introduction

In octuber 2000 the UJI Online robot was connected to the web for research purposes. It consisted of an educational robot (Mentor), with three cameras that enabled a user to remotely control pick and place operations of objects located on a board. Experiments about object recognition, virtual reality, augmented reality, speech recognition, and telemanipulation were accomplished in order to enhance the way people interacted with the system.

Since then, we realized the experience would be very interesting in the education and training domain. The first experience in education and training with students was performed in November 2001, were they programmed several pick and place operations on the virtual environment. This was very interesting for them because they obtained a more practical view of the Robotics subject, which until that moment was practiced in a more theoretical manner. After that, they did some manipulations using other online robots (e.g. Telegarden, Australia's Telerobot, etc.), letting them to compare different ways to design user interfaces for telerobotic systems. In November 2002 the second experience in education and training came up. Again, we provided the possibility to the students to program more sophisticated pick and place operations, using this time both, the off-line and the on-line robot.

After that, we considered necessary to enhance the system in order to not only let students control the robot from a user interface, but also programming it by using any standard programming language. In April 2003 the UJI TeleLab project came up, which supposed the design of a Java library called "Experiments" that let students to program their own control algorithms from any computer connected to the internet and execute them over the real robot. Some pilot experiments were performed with researchers and students since then. The interest for the design of Internet-based Tele-Laboratories is increasing enormously, and this technique is still very new. A very good example of already existing experiments in this area can be found in [3]. At the same time, we were very concerned about extending the project to letting users to program not only one educational robot, but also several manipulators. In fact, at the moment of writing, the UJI TeleLab lets scientists and students program not only one educational robot from home, but also design algorithms for two industrial and two educational manipulators.

2 Experimental Setup

As appreciated in Figure 1, we have 4 robots:

(1) 2 educational Mentor robots (Figure 1 first row) where three cameras are presented: one taking images from the top of the scene, a second camera from the side, and a third camera from the front. The first camera is calibrated, and used as input to the automatic object recognition module and 3D-model construction. The other two cameras give different points of view to the user when a teleoperation mode is necessary in order to accomplish a difficult task (e.g. manipulating overlapped objects).

(2) 1 industrial Adept One robot (see Figure 1 second row) operating on top of a conveyor belt. A static camera on top of the scenario provides a calibrated view of the objects.

* 0-7803-7952-7/03/$17.00 © 2003 IEEE.

(3) 1 industrial and redundant PA10 manipulator (see Figure 1 third row) placed on top of a Nomadic mobile robot.

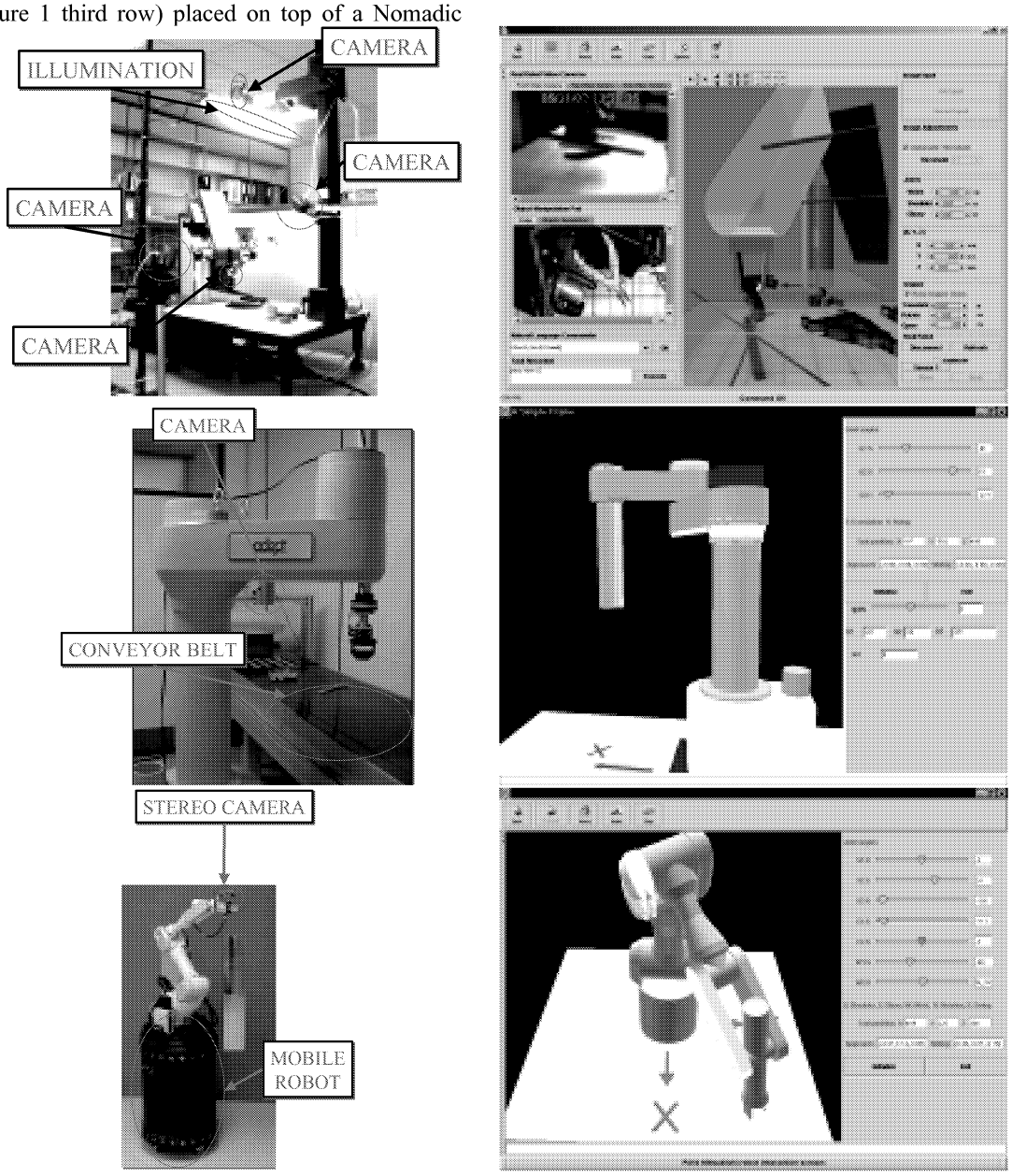

Figure 1. Experimental setup: Multirobot configuration

Once the user gets an online connection to a robot, the manipulator goes to the initial position, so the whole scene information is accessible from the top camera (calibrated position). At that moment, the computer vision module calculates a set of mathematical features that identify every object on the board [2]. Afterwards, the contour information is used in order to determine every stable grasp associated with each object [4], which is necessary for vision-based autonomous manipulation. This process is a must whether high level task specification from a user is required, and also in order to diminish network latency.

Meanwhile, we shall use the output from these visual algorithms in order to construct a complete 3D model of the robot scenario (i.e. the objects and the robot arm). Thus, users will be able to interact with the model in a predictive manner, and then, once the operator confirms the task, the real robot will execute the action [1].

3 Multirobot Hw Architecture

In previous projects, we have experimented different ways of accessing a single robot by several remote users [1]. A possibility would be letting just one operator to have control on the real robot, while the others are programming an off-line virtual environment. As an alternative to this problem (i.e. having a unique robot and several users), a possible solution would be allowing

access to more than one robot at the same time. In fact, in our laboratory we have two educational robots and two industrial ones that have been integrated into the system in order to allow scientists and students to program them remotely (see Figure 2).

The user selects in the user interface the robot he/she wants to interact with. The result is having several operators that are using a maximum of four robots at the same time, while the others are experimenting with simulated off-line virtual environments. In any case, while somebody is performing a manipulation in any of the situations abovementioned, he/she could ask other people connected into the system to help (via chat) in a particularly difficult task.

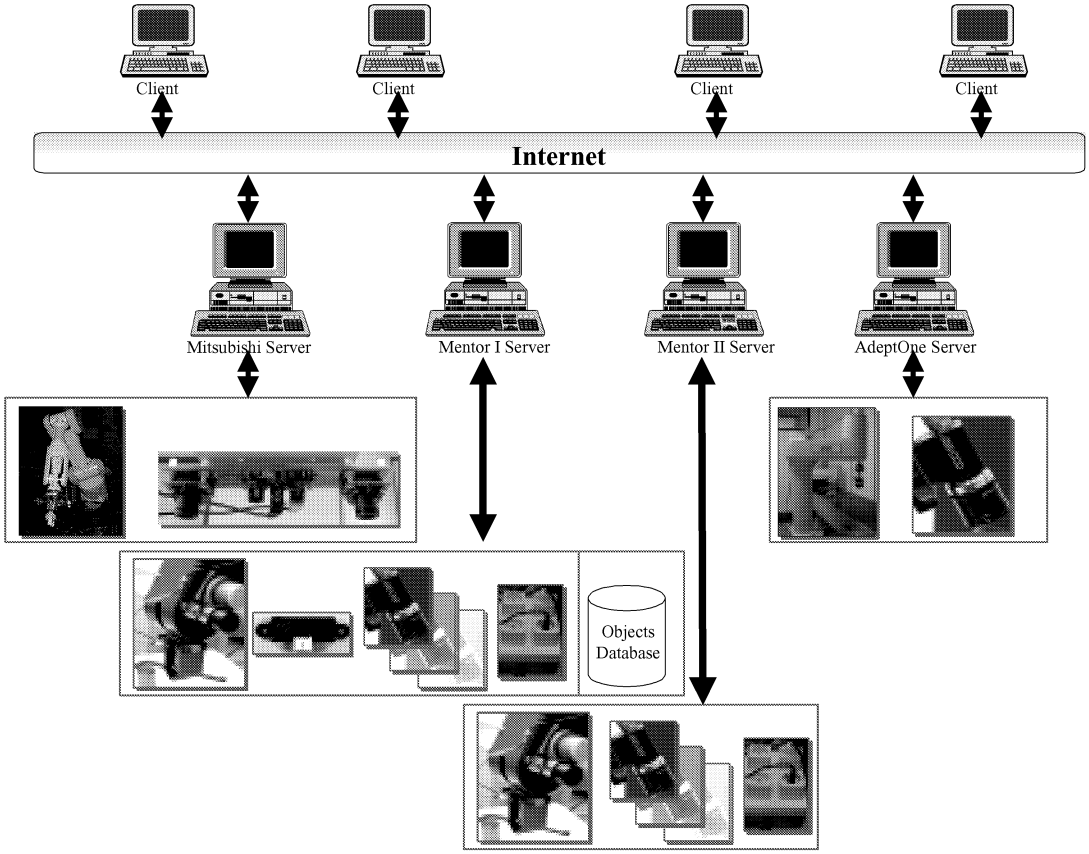

Figure 2. Multirobot HW architecture

4 Multirobot Sw Architecture

As we can see in Figure 3, the Tele-Lab accepts experiments (i.e. algorithms) as inputs using any programming language capable of managing TCP/IP sockets. We already provide a Java library for using the robot in a simply manner.

The outputs of the experiments are returned to the user by means of the Tele-Laboratory "Client Side" user interface, which permits the operator to see the results of their programmed actions directly on a multimedia Java3D user interface.

The client side applet launches the user interface that allows the user to interact with the remote robots. The connection of this applet with the server side (Servers Manager module) is performed through a unique connection via RMI (Java Remote Method Invocation API). It means every connection for every one of the robots is passing through this server.

Of course, by having many robots connected to the system the Servers Manager would be a bottleneck. However, this configuration is very important for the synchronization of the robots' operations as well as for the specification of reliable multirobot tasks. The Servers Manager connects to the different robot servers by using the CORBA standard. It facilitates enormously the configuration and installation of the different server sides (robots' servers).

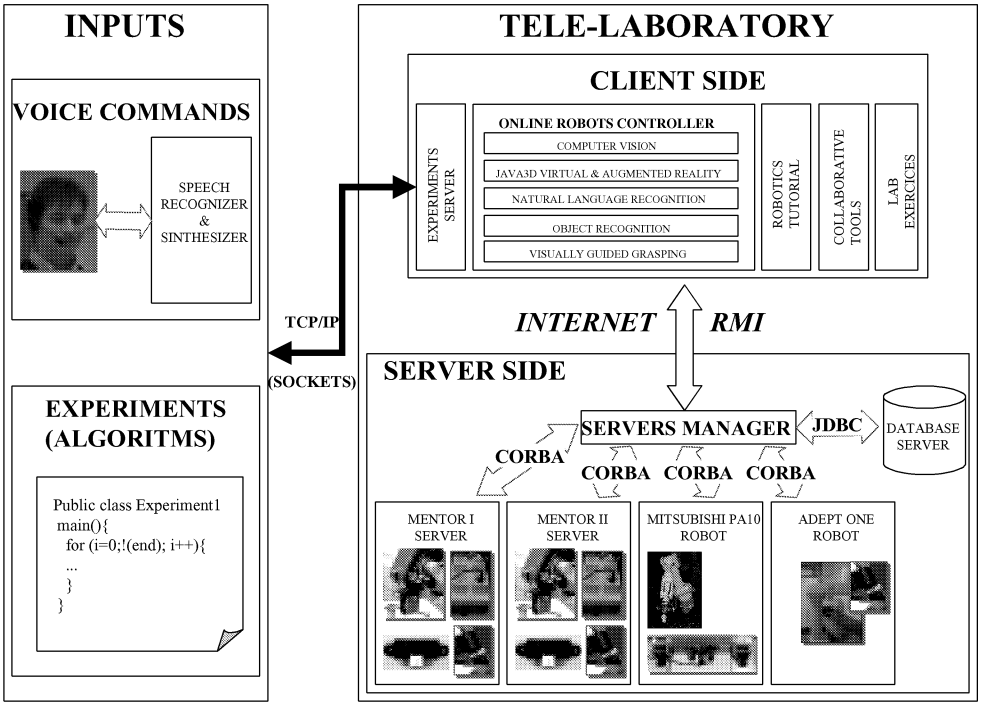

Figure 3. Software architecture for the Internet-based Tele-Lab

5 Results

In this section different configurations of the architecture are presented. For every configuration we have performed 400 remote movements on the robot. Some experiments have been done by using TCP, UCP and RMI protocols. Performance details are given.

5.1 Unique Client/Server IP (UCS_IP)

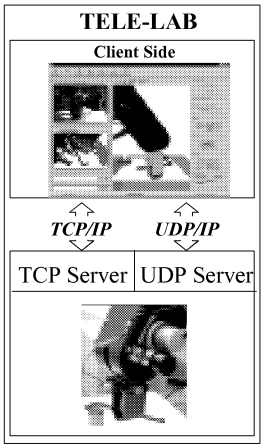

Figure 4. Unique Client/Server IP (UCS_IP) Software Architecture

By looking at Figure 6 we can appreciate that the latency **on campus** for both, TCP and UDP protocols is convenient enough to perform teleoperation experiments. Although there are some robot movements that have invested almost 14 msecs on communicating with the server side, the average is less than one msecs. Moreover, by looking at Figure 7 can be seen that by teleoperating the robot from home (same city) using the TCP/IP approach is rapid enough for our purposes (less than 90 msecs of average).

5.2 TeleLab Experiment RMI (TE_RMI)

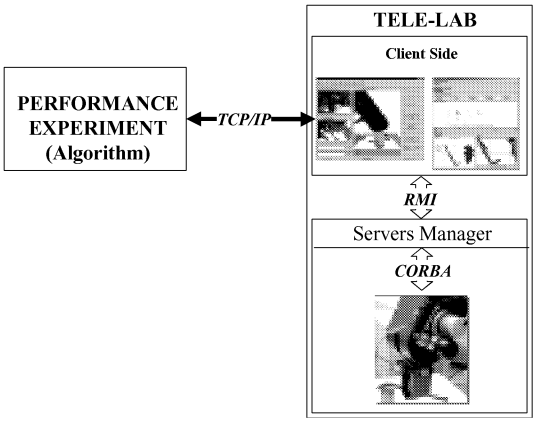

Figure 5. TeleLab Experiment RMI (TE_RMI) Software Architecture

The second configuration consists of using the TeleLab feature in order for the students and scientists to program

their own algorithms remotely. This means the experiment will connect first via TCP/IP on the local machine with the TeleLab user interface (Client Side), and then, once the operation is confirmed, the action is executed on the real robot via a RMI connection with the Servers Manager. Again, the Servers Manager connects via CORBA with the remote robot.

It must be taken into account that once an operation is sent to the Client Side, this operation is executed virtually over the predictive interface, and then, once confirmed the action by sending the command "confirm", this operation goes through the RMI and CORBA interfaces to move the real robot. It means this "predictive" configuration is very useful for students that are working off-line or just want to make sure on the virtual environment about the next robot movement. And of course, it supposes the time latency is increased a lot. In fact, the average for the experiments executed on campus is 2227 msecs, and for those working from home is 2439 msecs.

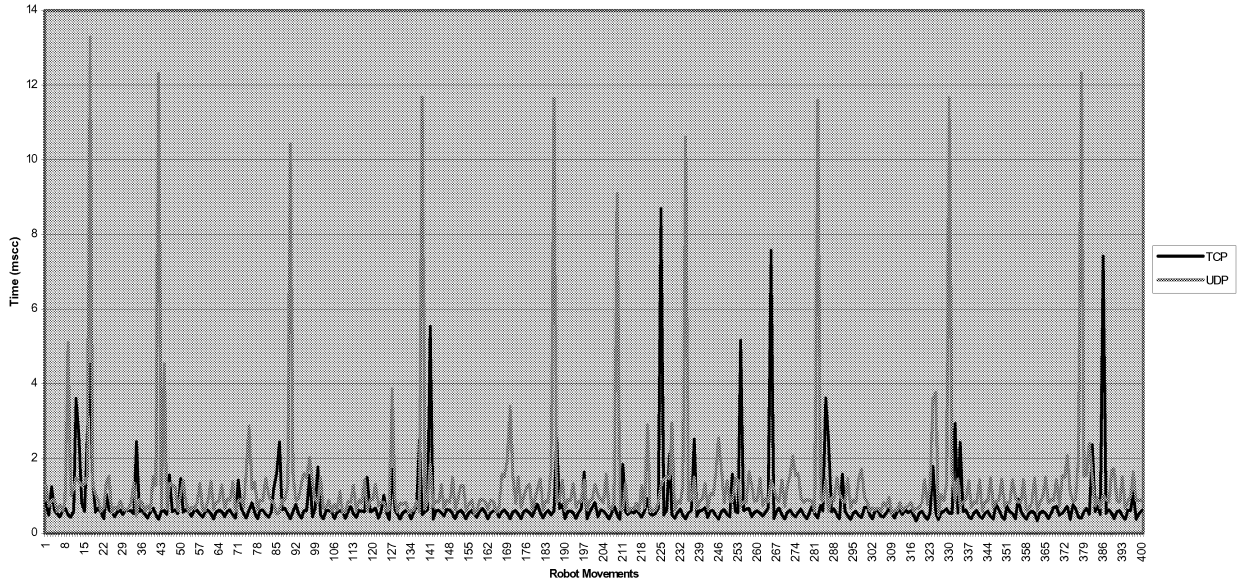

Figure 6. Graphical representation of the latency for the configuration Unique Client/Server IP (UCS_IP) **on Campus**, using both, the TCP and the UDP protocols

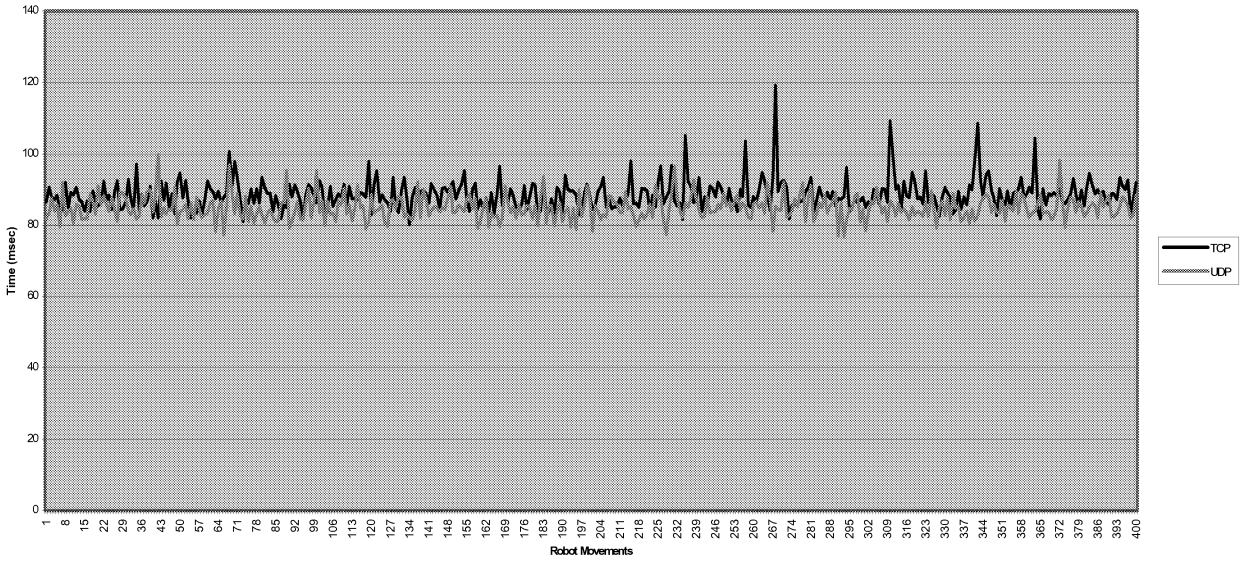

Figure 7. Graphical representation of the latency for the configuration Unique Client/Server IP (UCS_IP) **in the same City**, using both, the TCP and the UDP protocols

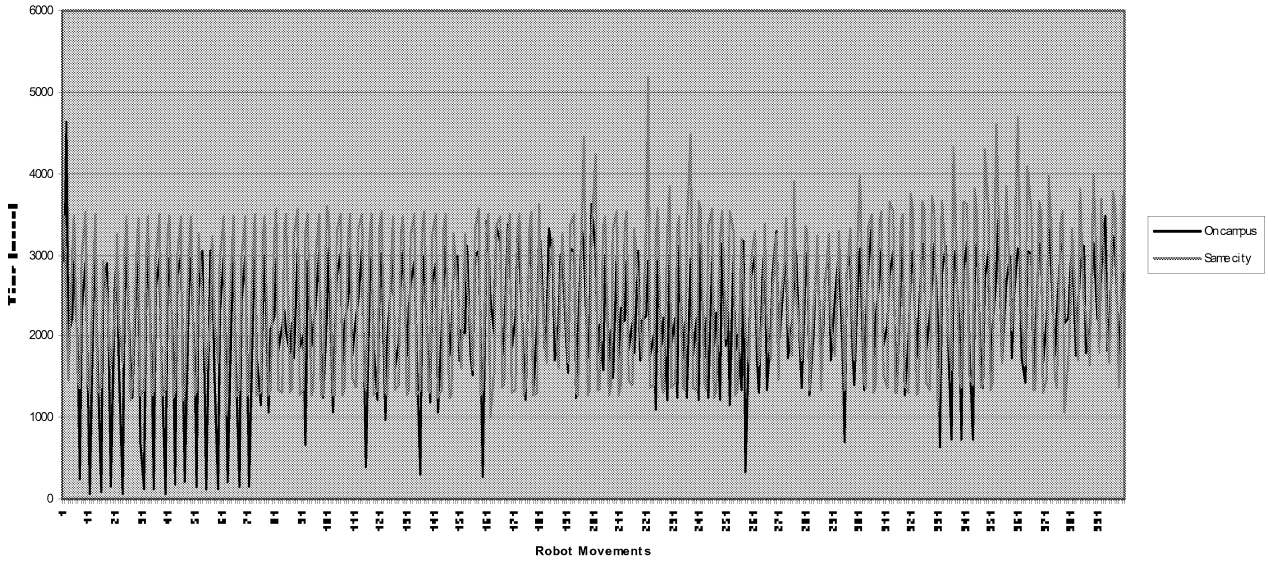

Figure 8. Graphical representation of the latency for the configuration TeleLab Experiment RMI (TL_RMI), using both on Campus and in the Same City configurations

6 Conclusions

The present paper has presented a pilot experiment with several telerobotic configurations that enable students and scientist take control of a robot remotely. The first configuration (UCS_IP) is the fastest one, and is the most convenient for traditional telerobotic applications. In fact the latencies shown in Table I justify this.

Time Latency in msecs	UCS_IP_TCP On Campus	UCS_IP_UDP On Campus	UCS_IP_TCP Same City	UCS_IP_UDP Same City	TE_RMI On Campus	TE_RMI Same City
Average	0,56	0,97	88,160	86,68	2227	2439

Table I. Summary of the Time Latencies for every configuration experimented

On the other hand, we have presented the Tele-Lab Multirobot Architecture (RE_RMI) that permits any student or scientist to not only control the robot from a user interface, but also allows him to control the manipulator from a Java program (remote programming). This second configuration is much more expensive because it requires a predictive display feature (requires confirmation of the command) and also is uses not only a TCP/IP protocol but three (TCP; RMI and CORBA). As a pilot experiment we consider the results obtained with the TeleLab configuration are good enough for several kinds of applications (e.g. Pick & Place experiments). By the other hand, performance should be improved a lot if more sophisticated experiments are going to be implemented in the future (i.e. Remote Visual Servoing).

Acknowledgments

This is to acknowledge sources of financial support for this research, that was provided in part by the Spanish Ministry of Science and Technology (CICYT) under project CICYT-DPI2001-3801, and by the "Conselleria de Cultura i Educació" (Generalitat Valenciana, Spain) under project GV01-244.

References

[1] R. Marín, P.J. Sanz., J.S. Sanchez, A Very High Level Interface to Teleoperate a Robot via Web including Augmented Reality. In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA). Washington, May, 2002.

[2] R. Marín, J.S. Sanchez, P.J. Sanz. "Object Recognition and Incremental Learning Algorithms for a Web-based Telerobotic System". In Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA). Washington, May 2002.

[3] G. T. McKee, The Development of Internet-Based Laboratory Environments For Teaching Robotics and Aritificial Inteligence. In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA). Washington, May, 2002.

[4] P.J. Sanz, A. P. del Pobil, J. M. Iñesta, G. Recatalá. "Vision-Guided Grasping of Unknown Objects for Service Robots". In Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 3018-3025, Leuven, Belgium. May 1998.