Transport Protocols for Remote Programming of Network Robots within the context of Telelaboratories for Education: A Comparative Analysis

Raul Wirz*, Raul Marín*, José M. Claver **, Josep Fernández ***, and Enric Cervera*

* Computer Engineering and Science, University Jaume I (UJI), 12071 Castellón, Spain.
** Computer Science Department, University of Valencia, 46071 Valencia, Spain
*** Engineering, Automatics, and Industrial Computers Department, University Politecnique of Catalonia, Barcelona, Spain.

Abstract – Within the context of Tele-Laboratories for Education the use of the Internet as communication media permits any researcher/student to perform remote experiments in a simple and reliable manner. Moreover, this situation introduces many interesting issues like network protocols for Internet robots, the effects of variable bandwidth and time-delays on telerobotics, etc. In this paper we present a comparative analysis of using several Internet transport protocols when performing a remote experiment within the UJI Industrial Telelaboratory. TCP, UDP, Trinomial and TEAR protocols are analyzed using the NS2 simulator. Conclusions show a set of characteristics the authors of this paper consider very important when designing an End-to-End Congestion Control transport protocol for Internet Telerobotics. These ideas are the basis for the definition of the SNRTP (Simple Network Robot Transport Protocol).

Keywords: *Networked Robots, Internet Congestion Control Protocol, Telerobotics, E-Learning, Industrial Robotics Telelaboratory.*

1 Introduction

One of the multiple applications of Networked Robotics is enabling Internet access to expensive devices (e.g. industrial robots, FPGA systems, conveyor belts, etc.) organized as telelaboratory for education. Thus, students and researchers can program their own robotic experiments via Internet and then obtain the results through, for example, a simple webpage [1-3].

One essential part of a Telelaboratory is the interconnection of sensors, cameras, and robots via a networked system [4-6]. In the scientific literature several works can be found that propose different ways and architectures to organize task-oriented applications of multiple network robots [7, 8]. Some of these architectures are focused on Internet software frameworks (e.g. Web Services at the application OSI layer) and have been extended from previous works in single-robot telerobotics.

Other works focus not only on the application protocols, but also at other levels of the OSI layers like transport and network, which enable real-time control and teleoperation of network robots over IP. In fact, as explained in [9, 10], solutions can be found to cope with the problems associated to the Internet in order to control networked robots: (1) time-varying transmission delay, and (2) not-guaranteed bandwidth.

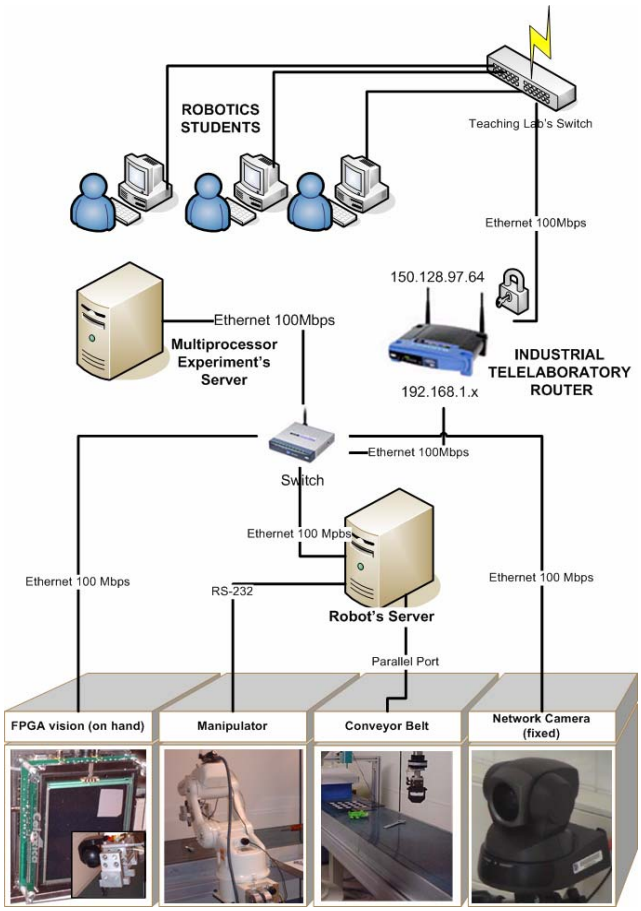

Figure 1. The UJI Industrial Telelab networking configuration

In the following paper we will present first the network architecture of the UJI Industrial Telelaboratory, including the application layer SNRP protocol for simple HTTP robots interconnection.

After that, we will focus on the transport protocols that enable the end-to-end congestion control in a TCP-Friendly manner [11] for teleoperation and tele-programming of robot arms. Simulations using TCP, UDP, trinomial [10], and TEAR [12] (TCP Emulation at Recievers) protocols are presented within the UJI

1095-2055/07/$25.00 ©2007 IEEE

Industrial Telelaboratory in order to obtain some conclusions. Then, from this results, a set of ideas are presented in order to define the working in progress SNRTP (Simple Network Robot Transport Protocol).

2 The UJI Industrial Telelaboratory Network Architecture

In Figure 1 we can see the Network connectivity of the UJI Industrial Telelaboratory. In fact, in this system we consider that every device (i.e. industrial robot, conveyor belt, FPGA, etc.) is connected to the same Ethernet network, and they act as single Network Robots that communicate with each other through the SNRP web-based protocol. This architecture offers many advantages like scalability and maintainability, and it introduces interesting issues like device synchronization, bandwidth and time-delays, and end-to-end congestion control.

In order to make the SNRP simple to use and implement, it uses the HTTP protocol as basis, which give him more interoperability and flexibility. However, for this kind of situation the HTTP does not provide the following features: (1) Event Notification, and (2) Support for structured information. These two characteristics are very important to design the SNRP framework in the industrial robotics area. To accomplish this, we have incorporated into the SNRP protocol the REST model [13], which permits the implementation of state-oriented applications and a simple scenario to design event notification and structured information features.

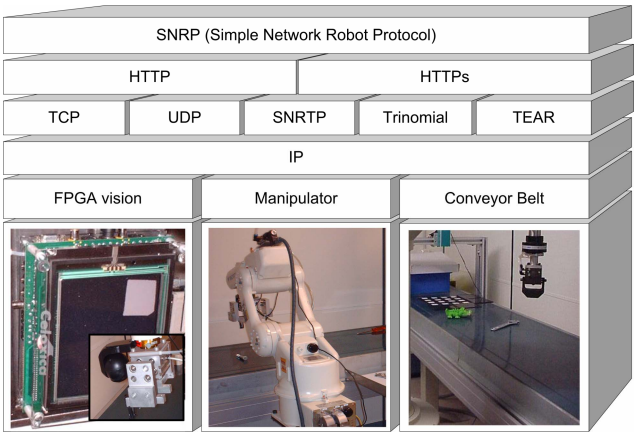

Figure 2. SNRP network architecture

Simplicity is maybe the most important challenge of a network robotics architecture, due to the fact that it must be possible for a very broad range of devices to be part of it. In fact, as explained in [14], thanks to this simplicity we were able to implement a prototype of SNRP Network Camera using a FPGA.

First of all, as we want enable the devices to be accessed through the internet, they should be able to manage the IP protocol. On top of it, the SNRP framework enables the device to accept TCP, and UDP connections. As explained in the next section, UDP and TCP are not the best solutions to perform remote control through the Internet, so the SNRP framework provides the possibility to transport the internet datagrams through other transport protocols like "trinomial", TEAR, or SNRTP.

3 Transport Protocols for Remote Control of Network Robots

The basic transport protocol available in the Internet for implementing remote control applications are the following:

(1) **UDP** (User Datagram Protocol) [15] that is based in the idea of sending a datagram from a device to another as fast as possible (i.e. best effort). This protocol does not guarantee that the information will reach the destination, and besides this, it does not manage any network congestion situation.

(2) **TCP** (Transmission Control Protocol) [16]. This guarantees the application level that the information will reach de destination performing the necessary retransmissions. Moreover, TCP takes care of the network congestion and adjust the transmission accordingly.

UDP is a protocol that does not maintain a connection with the Server side, it does not make retransmission of lost packets, it does not control the network congestion, and neither manages any confirmation of the packets that have reached the destination. The advantage of UDP, for remote control of devices via Internet is that having good network conditions the communication is accomplished without significant delays and without important fluctuations (i.e. delay jitter). Moreover, UDP does not assure that the packets have reached the destination in the proper order as they were sent, if fact, UDP does not inform if packets have even been received or not. Besides this, UDP does not perform any congestion control mechanism, which means the sending rate is not adapted according to the real bandwidth available. This situation implies that we need another protocol for controlling remotely devices via Internet.

On the other hand, TCP is a very sophisticated protocol that establishes a virtual connection between the sender and the receiver. Moreover, as TCP manages the confirmation of packets received properly, we can assure that the communication will be reliable. However, when TCP was designed they had in mind the reliable communication for application like e-mails and files (ftp), and not controlling devices like robots. The congestion control mechanism and the connection establishment implies having big delay jitter (fluctuation), a situation that is not appropriate for applications such as internet teleoperation of a robot manipulator using a haptic device. In the following figure we can see the results obtained when controlling a robot using both, TCP and UDP.

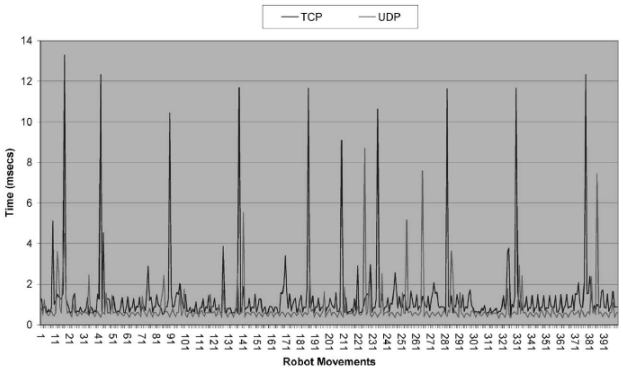

Figure 3. Delay response when controlling an industrial Motoman robot via Internet using UDP and TCP (i.e. On campus)

The majority of current telerobotic applications using the Internet (e.g. telelaboratories) use TCP or UDP. For this, the variable time-delay and bandwidth effects are resolved in the application level by using intelligent sensors, predictive displays, and high level commands . On the other hand, if we really need to perform a teleoperation, we need to find applications that are closer to real time [17]. In this situation we need more specific protocols [18].

As this is a very emergent research field, in the scientific literature we cannot find many articles describing specific protocols to teleoperate networked devices (i.e. like robots) via Internet. On the other hand, we can find many protocols to design networked applications that require the transmissions of Multimedia content via Internet: (1) TFRC (TCP-Friendly Rate Control Protocol) [19], RAP (Rate Based Adaptation Protocol) [20], LDA (Loss-Delay Adjustment Protocol) [21], SIMD (Square-Increase/Multiplicative-Decrease Protocol) [22], and RTP (Real Time Protocol) [23]. These protocols are not very convenient for telerobotics due to the fact that they use an intermediate buffer to compensate the delay jitter when receiving video and audio. In telerobotics using buffers implies obtaining an overall higher delay that affects enormously to the immediate control of robots.

Some of the few works that specifically design protocols for teleoperation are the following:

(1) Trinomial method [10]: It is a rated-based protocol, which means it manages the network congestion by adjusting the inter-packet gap (IPG) instead of the window size schema that uses TCP. Thus, the protocol controls the number of datagrams per second depending on the available bandwidth. The trinomial method uses UDP as basis. It means that the trinomial is able to adapt to the network congestion and available bandwidth without affecting very much the way the user teleoperates the robot. As observed in [10], the trinomial protocol provides a sending curve that is quite smooth and better uses the available bandwidth, obtaining then a very good efficiency compared to the UDP and TCP protocols. In the following section we will study some parts of the trinomial that we consider can be improved in order to be applied in the telelaboratories field.

(2) Real-Time Network Protocol (RTNP) [24] is a very simple protocol that uses an identification in the UPD/TCP headers to inform the real-time operating system that the received packet has the category of "real time", in order to give it the maximum priority when passing the packet to the application level. The RTNP shows that the overall time-delay between the client and the server depends not only on the network but also on the software provided by the operating system.

(3) Interactive Real-Time Protocol (IRTP) [25] is a protocol that takes the advantages of both, TCP and UDP, to improve the response in teleoperation systems. It is a connection-oriented protocol that implements congestion control and error control. To enhance the efficiency, the IRTP protocol simplifies the packet header as much as possible, getting then a major relationship between the data that is sent by the application level and the control information.

Moreover, in the telelaboratories context there are situations where the student/researcher is performing an experiment from home using an ordinary ADSL connection. This kind of asymmetric communication gives normally a poor upload link and a good download bandwidth. The TEAR protocol (TCP Emulation at Receivers) [12] is specifically designed to the transmission of multimedia streams on asymmetric connections. In the next section we will provide some simulations to compare the performance of the trinomial, TCP, and TEAR protocols within the telelaboratories context.

4 RTT behaviour

In this section we are going to observe the RTT behaviour of trinomial, TCP and TEAR protocols on a simple scenario where 3 computers (e.g. students) access information from a remote host (e.g. robot) via a router.

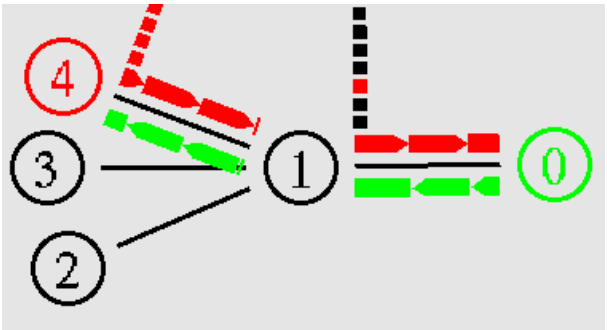

Figure 4. Nodes configuration of the RTT experiment

As seen in Figure 4, we are having the node 0 that represents the industrial robot of the telelaboratory. Node 1, represents the router that gives access to every device in the telelaboratory. Nodes 2 and 3 represent 2 students that are connected to the telalaboratory, and they are

monitoring the experiment performed by node 4. The Node 4 represents a student that is performing a teleoperation (or visual servoing) experiment on the industrial robot (i.e. node 0). For the simulation the traffic from nodes 2 and 3 is TCP based, and the traffic from Node 4 (i.e. the experiment) will vary from trinomial, TCP and TEAR.

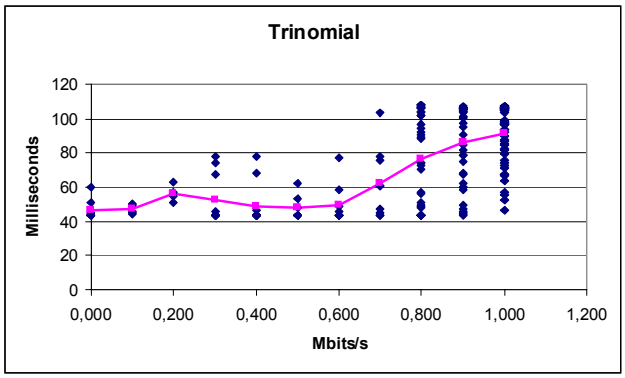

Figure 5. Results of the RTT behaviour NS-2 simulation when Node 4 uses the Trinomial protocol

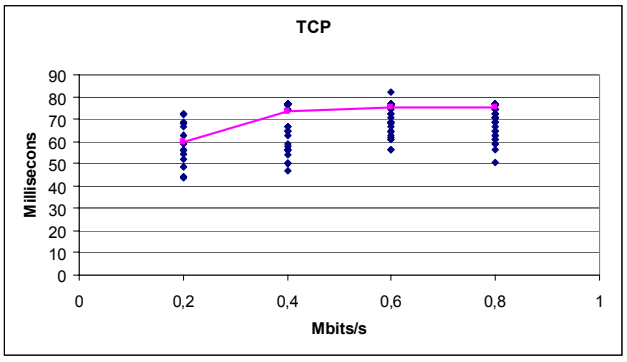

Figure 6. Results of the RTT behaviour NS-2 simulation when Node 4 uses the TCP protocol

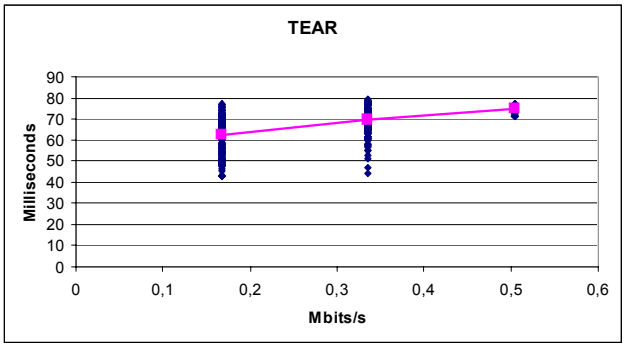

Figure 7. Results of the RTT behaviour NS-2 simulation when Node 4 uses the TEAR protocol

As we can observe from figures 5, 6, and 7, the trinomial almost consumes the available bandwidth through the router, obtaining an average RTT of 74,25 milliseconds. Moreover, there are packets that reaches the 110 milliseconds of RTT. The trinomial protocol sets the router buffers to the maximum load, which implies increasing the RTT average between the student and the robot. On the other hand, the trinomial protocol is the one that sends more packets per second, increasing the information that comes from the student to the robot and vice versa.

The TCP protocol consumes the 80% of the available bandwidth, at an average RTT of 74,63 milliseconds. We can observe that TCP is more TCP-friendly than trinomial. On the other hand, as TCP performs retransmissions the number of received packets at Node 0 is not so significant than using the trinomial protocol.

For the TEAR protocol, it sets the router buffers at the 50% of the available bandwidth, at an average RTT of 67 milliseconds. In some situations the RTT of the trinomial protocol goes twice the TEAR one.

	Generated packets	**Dropped packets at router**	**Lost packets**
Trinomial	11140	906	230
TCP	9638	3	63
TEAR	8584	4	56

In summary, the TEAR has an RTT more stable and shorter, using less bandwidth and sending less packets between the student and the robot. The trinomial uses more bandwidth (in our simulation it reaches the 100% available bandwidth). It has the biggest RTT and loses more packets than any other. The TCP loses less packets than any other, but it has the highest RTT and uses 80% of the available bandwidth.

5 Visual Servoing at home NS-2 simulation within the Industrial Telelaboratory

In this section we are going to study the behaviour of the TCP and TEAR protocols for the transmission of the monitoring camera on the telelaboratory.

As we can see in Figure 8, there is a student that performs a visual servoing experiment from home over the industrial telelaboratory (i.e. Node 10). At the same time, several students "on campus" are accessing to the information from the telelaboratory cameras for monitoring purposes.

In the simulation, the student's experiment sends a UDP packet to the FPGA, which returns the grasping line of the object at the robot scenario. It applies a control law following the on-hand visual servoing control until the grasping line is centered at the middle of the gripper. As shown in figures 9 and 10, the TEAR protocol is smoother than the TCP, which is very much appropriate for the monitoring camera link.

However, as we can see in the figures the TEAR and the TCP protocols does not use the 100% of the available bandwidth, fact that is accomplished by the trinomial protocol. However, the trinomial protocol is specifically

designed for the robot interaction and not for multimedia transmission. As well, as we have seen in the previous section, it sets the communication link to the maximum bandwidth (which is very convenient), but to the maximum time-delay too. In fact, for performing visual servoing experiments the time-delay must be minimized.

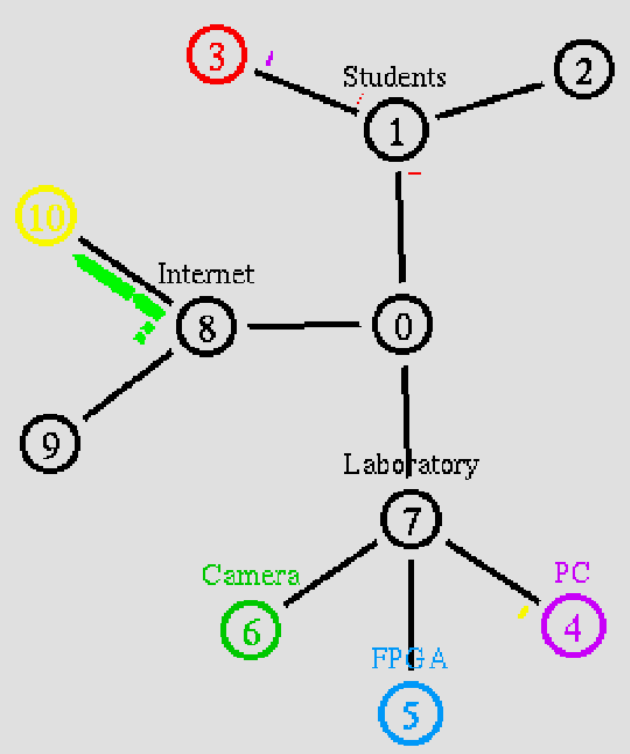

Figure 8. Nodes configuration of the visual servoing experiment NS-2 simulations

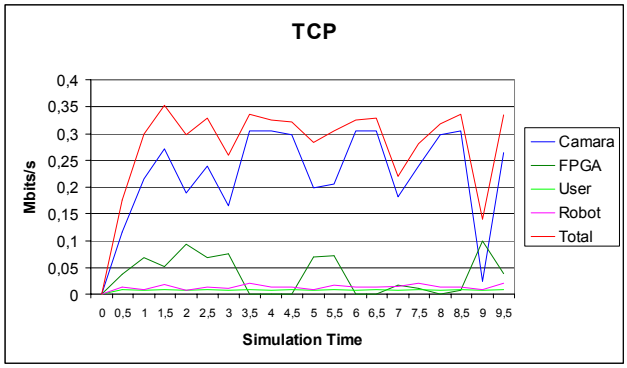

Figure 9. Telelaboratory experiment using TCP for the monitoring Camera.

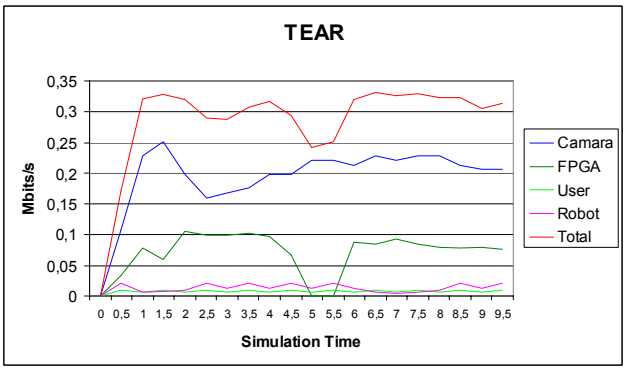

Figure 10. Telelaboratory experiment using TEAR for the monitoring Camera.

6 Conclusions

Within the telelaboratories context for education the UDP and TCP protocols can be improved in order to acquire better performance and smoothness. The trinomial protocol is a nice solution which uses as much bandwidth as possible, providing smoothness for a bilateral teleoperation via Internet. However, it introduces extra time-delay due to the fact that it sets the router buffers to the maximum load. As well, as seen in the RTT section, depending on the parameters configuration it can be not so TCP-Friendly like other protocols. The RTT behaviour is very important for some experiments like visual servoing and teleoperation. Please note these conclusions about the trinomial are extracted from the simulations done by the authors of this article, as they are not available via other alternatives.

The TEAR protocol is more conservative than the trinomial and the TCP (i.e. it uses less bandwidth and RTT), in a very smooth way. However, for the telelaboratory context this is not sufficient, due to the fact that we need to set priorities for every data flow in advance. For example, for the visual servoing experiment, the FPGA and robot flows must have the minimum RTT and priority, and the Camera flow does not need to have such configuration.

For that, the requirements we have for the SNRTP (Simple Network Robot Transport Protocol) are the following:

1. Smooth Congestion Avoidance: SNRPT will study the smooth equilibrium between bandwidth and time delay for master/slave teleoperations. This equilibrium depends on the robot configuration and the specific application.
2. Differentiated Services: Including priorities in the SNRTP flows will allow the bandwidth allocation of cameras, robot control, and sensor information in a differentiated manner.

Acknowledgements

This work has been partially funded by the Spanish Ministry (MEC) under Grants DPI2005-08203-C02-01, DPI2004-01920, TSI2004-05165-C02-01, TIN2006-15516-C04-02, by the European Commission FEDER funds "Consolider Ingenio-2010" CSD2006-00046', by the Fundació Caixa Castelló under Grants P1-1B2003-15, and P1-1A2003-10, and by the EU-VI Framework Programme under grant IST-045269 - "GUARDIANS" of the EC Cognitive Systems initiative

7 References

[1] R. Marin, P. J. Sanz, P. Nebot, and R. Wirz, "A multimodal interface to control a robot arm via the web: A case study on remote programming," *Ieee Transactions on Industrial Electronics,* vol. 52, pp. 1506-1520, Dec 2005.

[2] R. Wirz, R. Marin, and P. J. Sanz, "Remote Programming over Multiple Heterogeneous Robots: A Case Study on Distributed Multirobot Architecture." vol. 33 Industrial Robot, 2006.

[3] R. Marin, P. J. Sanz, and A. P. Del Pobil, "The UJI Online Robot: An education and training experience," *Autonomous Robots,* vol. 15, pp. 283-297, Nov 2003.

[4] R. Zurawski, "Industrial Information Technology Is Coming of Age," *IEEE Transactions on Industrial Informatics,* vol. 3, 2007.

[5] A. P. Kalogeras, J. V. Gialelis, C. E. Alexakos, M. J. Georgoudakis, and S. A. Koubias, "Vertical Integration of Enterprise Industrial Systems Utilizing Web Services," *IEEE Transactions on Industrial Electronics,* vol. 2, p. 9, 2006.

[6] G. T. McKee, D. I. Baker, and P. S. Schenker, "Network robotics: Dynamic reconfigurable architectures," in *SPIE Intelligent Robots and Computer Vision XXII*, 2004.

[7] G. T. McKee, D. I. Baker, and P. S. Schenker, "Robot Spaces, Module Networks and Distributed Robot Architectures," in *IROS 2004 Workshop on Networked Robotics*, Sendai, Japan, 2004.

[8] B. K. Kim, "Web Services Based Robot Control Platform for Ubiquitous Functions," in *IEEE International Conference On Robotics and Automation (ICRA)*, 2005.

[9] P. X. Liu, M. Q. H. Meng, and S. X. Yang, "Data Communications for Internet Robots," *Autonomous Robots,* vol. 15, 2003.

[10] P. X. Liu, Meng, M. Q. H., P. R. Liu, and S. X. Yang, "An End-to-End Transmission Architecture for the Remote Control of Robots Over IP Networks," *IEEE Transactions on Mechatronics,* vol. 10, 2005.

[11] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking,* vol. 7, p. 14, 1999.

[12] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP Emulation At Receivers: flow control for multimedia streaming," Department of Computer Science, NCSU 2000.

[13] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern Web Architecture," in *ICSE 2000*, 2000, pp. 407-415.

[14] R. Marin, G. Leon, R. Wirz, J. Sales, J. M. Claver, and P. J. Sanz, "Remote Control within the UJI Robotics Manufacturing Cell using FPGA-based vision," in *ECC'2007 European Control Conference*, 2007.

[15] J. Postel, "RFC 768: User Datagram Protocol," 1980.

[16] J. Postel, "RFC 793: Transmisión Control Protocol, DARPA Internet Program Protocol Specification," 1981.

[17] J. Park and O. Khatib, "Robust Haptic Teleoperation of a Mobile Manipulation Platform," in *Experimental Robotics IX, Star, Springer Tracts in Advanced Robotics*, 2005.

[18] S. E. Butner and M. Ghodoussi, "A real-time system for tele-surgery," in *21st International Conference on Distributed Computing Systems*, 2001.

[19] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCPfriendly rate control protocol," in *NOOSDAV'1999*, 1999, pp. 137-151.

[20] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *IEEE Infocom*, 1999, p. 1337–1345.

[21] D. Sisalem and H. Schulzrinne, "The loss-delay adjustment algorithm: A TCP-friendly adaptation scheme," in *Int. Workshop Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, 1998, p. 215–226.

[22] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-friendly SIMD congestion control and its convergence behaviour," in *9th IEEE Int. Conf. Network Protocols*, Riverside, CA, 2001, p. 156–164.

[23] H. Schulzrinne, S. Casner, R. Fredrick, and V. Jacobson, "RFC 1889: RTP: A transport protocol for real-time applications," 1996.

[24] Y. Uchimura and T. Yakoh, "Bilateral robot system on the real-time network structure," *IEEE Transactions on Industrial Electronics,* vol. 51, 2004.

[25] L. Ping, L. Wenjuan, and S. Zengqi, "Transport layer protocol reconfiguration for network-based robot control system," in *IEEE Networking, Sensing and Control 2005*, 2005.