END-TO-END CONGESTION CONTROL PROTOCOLS FOR INTERNET TELEROBOTICS

Raul Wirz*, Raul Marín, José M. Claver+, Josep Fernández, Manuel Ferre, Rafael Aracil
*Jaume I University
Avda. Vicente Sos Baynat, s/n
Castellón, Spain
{wirz,rmarin}@icc.uji.es, jclaver@uv.es, josep.fernandez@upc.edu, m.ferre@upm.es, aracil@etsii.upm.es

ABSTRACT
Internet Telerobotics presents many interesting aspects within the context of master/slave teleoperation like variable bandwidth and time-delays. Some of these aspects have been treated in the literature from the control point of view. However, only a few works are related to the way internet protocols can help to minimize the effect of delay and bandwidth fluctuation on teleoperation. In this paper we present the capabilities of TCP, UDP, TCP Las Vegas, TEAR, and Trinomial protocols when performing an Internet teleoperation. The comparative analysis is presented through simulations within the NS2 platform. Conclusions present a set of requirements that are necessary in order to define the SNRTP protocol (Simple Network Robot Transport Protocol), which is specifically being designed for Internet teleoperation. The SNRTP protocol tries to minimize the end-to-end time-delay by using as much bandwidth and possible (e.g asymmetric networks), and maintaining the TCP-Friendly criteria. Moreover, the SNRTP is being designed to be fully integrated with the HTTP-based robotics application protocol SNRP (Simple Network Robot Protocol), which have been described in detail in previous works.

KEY WORDS
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.
In the following paper we will present first the network architecture of the UJI Industrial Telelaboratory. 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 Industrial Telelaboratory in order to obtain some conclusions. Then, from this results, a set of ideas are presented in order to define the requirements for SNRTP (Simple Network Robot Transport Protocol), which are presented in the last section of the paper.

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 SNRTP 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 [13] [14].

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 should reach the destination performing the necessary retransmissions. Moreover, TCP takes care of the network congestion and adjusts the transmission accordingly.

UDP is a protocol that does not maintain a connection with the Server side, and 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 remotely controlling 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 using both, TCP and UDP.

![Figure 3. Delay response when controlling a robot via Internet using UDP and TCP (i.e. On campus)'](image-url)
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 rate-based protocol, which means it manages the network congestion by adjusting the inter-packet gap (IPG) instead of the window size schema that TCP uses. 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 telerobotics field.

(2) **Real-Time Network Protocol (RTNP)** [24] is a very simple protocol that uses an identification in the UDP/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 and the control information.

Moreover, in the telerobotics 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 telerobotics context.

### 4. RTT behaviour

In this section we are going to observe the RTT (Round-Trip Time) behaviour of trinomial, TCP, TCP Las Vegas, and TEAR protocols for the industrial telerobotic. As seen in Figure 4, we are having the Node 4 that represents the industrial robot of the telerobotic. Node 7, represents the router that gives access to every device in the telerobotic. Nodes 2 and 3 represent 2 students that are connected to the telerobotic, and they are monitoring the experiment performed by node 10. The Node 10 represents a student that is performing a teleoperation (or visual servoing) experiment on the industrial robot (i.e. node 4). For the simulation the traffic from nodes 2 and 3 is TCP based, and the traffic from
Node 10 (i.e. the experiment) will vary from trinomial, TCP, TCP Las Vegas, and TEAR.

Figure 5. Results of the RTT behaviour NS-2 simulation when Node 10 uses the Trinomial protocol (x-axis represents the used bandwidth, y-axis the RTT, and the purple line the average of every sample)

Figure 6. Results of the RTT behaviour NS-2 simulation when Node 10 uses the TCP protocol

Figure 7. Results of the RTT behaviour NS-2 simulation when Node 10 uses the TCP Las Vegas protocol

As we can observe from figures 5, 6, 7, and 8, the trinomial almost consumes the available bandwidth at the router, obtaining an average RTT of 74,25 milliseconds. Moreover, there are packets that reaches 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.

Figure 8. Results of the RTT behaviour NS-2 simulation when Node 10 uses the TEAR protocol

As we can observe from figures 5, 6, 7, and 8, the trinomial almost consumes the available bandwidth at the router, obtaining 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 51,46 milliseconds. In some situations the RTT of the TCP protocol reaches twice the TEAR one.

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.

The experiment consists of performing a visually guided grasping of an object using on-hand visual servoing control. For this, the telelaboratory already provides the student with a networked computer vision system implemented through a FPGA, which returns geometrical data of the objects in the scene (i.e., centroid, grasping points, grasping orientation, etc.). In the simulation, the student’s experiment sends a UDP packet to the FPGA, which returns the grasping points of the object at the robot scenario. The student 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.

As we can see in the figures, the trinomial protocol uses 100% of the available bandwidth. 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.

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 this 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 (ie. 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 a priori priorities for every data flow. For example, for the visual servoing experiment, the FPGA and robot flows must have the minimum RTT and the maximum priority, and the Camera flow does not need to have such a configuration.

For that, the requirements we want 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.

7. Future Applications

In the near future, the protocols presented will be evaluated in a force reflecting teleoperation scenario, which we call the ‘thumb wrestling’ demonstrator. This demonstrator is focused on the transmission of haptic interactions between users. A user attempts to capture his opponent’s thumb while avoiding to be wrestled out. During the match, the wrist and the rest of fingers are kept still. Each user has a haptic device (called MasterFinger) and a computer display:

- The haptic device registers the user movements and transmits interaction forces.
- The computer runs a graphical simulation for showing the users’ movements.
All haptic information is managed by a computer server that receives, processes and sends the information back via Ethernet. The whole system is composed by five networked devices:

- PC-Control, it is the application server that managed all information. It is running under RT-Linux Pro.
- 2 graphical stations. They provide the user with a graphical simulation showing the movement of both hands.
- 2 haptic controllers. They send the user movements and reflect forces according to the thumb collisions.

Figure 11. The ‘thumb wrestling’ demonstrator scenario.

Acknowledgements

This work has been partially funded by the Spanish Ministry (MEC) under Grants TIC2003-08496, DPI2004-01920, TSI2004-05165-C02-01, Consolider Ingenio 2010 CSD2006 00046, TIN2006-15516-C04-02, by the Fundació Caixa Castelló under Grants P1-1B2002-07, P1-1B2003-15, and P1-1A2003-10, and by the Generalitat Valenciana under grant GV04A-698.

References