University of Alberta

A STUDY OF NETWORK TRANSPORT PROTOCOLS AND ITS APPLICATIONS IN INTERNET-BASED TELEOPERATION AND E-SERVICE ROBOTIC SYSTEMS

by

Xiaoping Liu ©

A Thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**

Department of Electrical and Computer Engineering

Edmonton, Alberta
Fall 2002

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

0-612-81225-1

Canadä

University of Alberta

Library Release Form

Name of Author: Xiaoping Liu
Title of Thesis: A Study of Network Transport Protocols and Its Applications in Internet-based Teleoperation and E-service Robotic Systems
Degree: Doctor of Philosophy
Year of Degree Granted: 2002

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Xiaoping Liu
449K, Michener Park
Edmonton, Alberta
Canada, T6H 4M5

Date: July 12, 2002

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **A Study of Network Transport Protocols and Its Applications in Internet-based Teleoperation and E-service Robotic Systems**, submitted by Xiaoping Liu in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Dr. Max Q-H Meng (Supervisor)

Dr. Tongwen Chen

Dr. Qing Zhao

Dr. Hong Zhang

Clarence

Dr. Clarence W. de Silva (External Examiner)

Date: July 10, 2002

Abstract

In this thesis, we present a novel network transport protocol that we call the trinomial protocol for Internet-based teleoperation and e-service robotic systems. The trinomial protocol provides minimized delays and delay jitter. In the steady state of the network, its transmission rate is smooth, however, when network bandwidth varies, it adapts to the change quickly. The trinomial protocol satisfies all the constraints on transport protocols: it is responsive, inter-protocol fairness convergent, intra-protocol fairness convergent and efficiency convergent.

The implementation issues of the trinomial protocol are examined. In particular, the estimation of roundtrip time (RTT), which is an integrated component of the implementation of the trinomial protocol, is explored. It is revealed that the adjacent and near-adjacent observations in the delay time series are linearly dependent rather than random. It is also indicated that there are only weak or no non-linear correlations among RTT observations. Based on these findings, we present a novel algorithm based on the maximum entropy principle (MEP) for RTT estimation. Compared to traditional autoregressive moving average (ARMA) method, the MEP method is adaptive and capable of tracking RTT dynamics very well.

In the last part of the thesis, based on the trinomial protocol, we develop and implement a new modular platform for mobile e-service robotic systems. In the experiments, the users successfully guided a mobile robot remotely through a clustered laboratory environment over the Internet by using a Java-enabled web browser on an ordinary PC.

Acknowledgements

First and foremost, I would like to give my sincere thanks to my supervisor, Dr. Max Q-H Meng, for his masterly guidance, constant encouragement, and invaluable advice throughout my graduate studies at the University of Alberta. This work would have been impossible without his full support and enormous patience over the last three years.

I am heavily indebted to Dr. Tongwen Chen, Dr. Hong Zhang and Dr. Qing Zhao for their invaluable guidance, helps and suggestions throughout the research. A special thank is expressed to Dr. Clarence W. de Silva for serving as the external examiner and offering expert suggestions.

I would like to thank all my colleagues in the Advanced Robotics and Teleoperation (ART) Lab. They are: Mike Y. Chen, Simon X. Yang, Jason J. Gu, Cheng Chen, James A. Smith, Dalong Wang, Yan Liu, Axel von Bertoldi, Xiufen Ye, Chao Hu, Steve Dillen, and Sean Verret. Thank you all for your helps, encouragements and suggestions in the last three years.

I would like to acknowledge the financial supports by NSERC through grants to Dr. Max Q-H Meng, by the Province of Alberta Graduate Fellowship, and by the Department of Electrical and Computer Engineering Scholarship, University of Alberta.

Last but not least, I would like to give my thanks to my family. Especially thanks to my wife, Yuerong. Without her love, encouragement and patience, this work would be far from complete. I owe my whole life to my parents, Lihua and Congxuan, who are the reason I am. Their greatest, selfless and endless love is the root of the work and the continuing source of my future career and life. I would like to thank my daughter, Claire, for bringing me lucks and uppermost happiness.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Current Internet applications are primarily dominated by the transmission, exchange, and publication of static information, such as texts, pictures and stored videos. The underlying technologies continue to advance quickly such that wired and wireless networks are becoming increasingly accessible, affordable, and powerful. Current trend is for people not to be tethered to a desktop any longer; instead they can use the Internet wherever and the industry is responding with small mobile Internet devices – palm-tops, personal digital assistants (PDA), and multi-function cellular phones [1]. As a result, network-based applications are rapidly expanding into new areas and network-enabled systems are attracting more and more attention from both academia and industry [115].

1.1 Internet-Based Teleoperation and E-Service Robotic Systems

Around 1954, the first master-slave teleoperators were developed by Ramond Goertz at the Argonne National Laboratory near Chicago, USA [12]. These were mechanical pantograph mechanisms by which radioactive materials in a “hot cell” could be manipulated by an operator outside the cell. Electrical servomechanisms soon replaced the direct mechanical tape and cable linkages.

Since then, many telerobotic devices have been developed in a wide range of application fields such as military, undersea oil mining, space exploration, toxic waste cleanup, medical arthroscopy, warehousing, construction, agriculture, mail delivery, fire fighting, lifesaving, policing, assistive devices for the disabled, tele-diagnosis, tele-surgery and entertainments etc. [13, 141-151].

In 1994, almost at the same time, the first two telerobots were connected to the Internet [2, 3]. One of them is an ASEA IRb-6 robot at the University of Western Australia. People with web access are able to control the robot to manipulate wooden blocks on a table over the Internet. The other one is a SCARA robot at the University of Southern California. Since then, more and more robotic prototypes have been developed and tried on the Internet [3-11,152,153]. Most of them fall into one of two categories: telemanipulators and teleoperated vehicles (teleoperated mobile robots). Some famous examples among the former include a block-moving robot [2]; a tele-gardening robot [3]; and a tele-painting robot, PUMApaint [153,154] etc. Of the latter are an office delivery mobile robot, Xavier [4]; a tour guide mobile robot, RHINO [5]; a tele-embodiment device [6], and a maze navigation mobile robot, KhepOnTheWeb [7] etc.

We call all above systems Internet-based teleoperation systems and e-service robotic systems, which can be broadly defined as: *an Internet-based teleoperation system and e-service robotic system is a device or robotic system that can be invoked and controlled remotely by the user (human operator) via the Internet to provide services, such as remote-sensing, tele-inspection, tele-mobility, tele-manipulation and tele-embodiment etc.*

The underlying motivations for these systems lie mainly upon the following two facts.

- The Internet put homes, offices, libraries, hospitals, stores and many other real physical places on-line. We have a strong impression that everything will be connected and everything will be on-line;
- Advanced robots can see, hear and even smell. Sophisticated robotic systems can move and manipulate real objects. In one word, modern robots will more and more function as human beings.

If we integrate the Internet and robots, we have e-service robotic systems. If the controlled devices are not robotic devices, we call them teleoperation systems. The vision is that these systems can extend human sensory and motor abilities beyond physical limitations and support human activities without human physical presence.

There are a lot of prospective applications for these systems in areas, such as industrial manufacturing and inspection, telesurgery, housework assistance, home security and elderly care, military reconnaissance, tele-training, and entertainment etc. Here we mention some potential examples:

- When you are out of town, you are still able to check your home to make sure everything is in order by navigating a personal roving robotic presence device around inside your home remotely via the Internet by using a portable PC or only a wireless hand-held communication device;
- To save money, you can switch off your home heating system when you leave home for work in the morning and turn it on in your office when you are ready to go back home;
- Surgical robots teleoperated over the Internet or satellite links are gaining more and more attention, especially in military battle field medical applications; and
- As population aging becomes an increasingly urgent societal problem in Canada and around the world, inexpensive assistive devices or systems are becoming demanding, which allow healthcare centers or the relatives to remotely monitor and help senior people who stay at home via the Internet.

1.2 Challenges

The prospects of Internet-based teleoperation and e-service robotic systems are promising. However, many difficulties are yet to be conquered before these systems become real-world applications. Current systems are largely experimental and not ready to provide real-life services. It is well recognized that the most challenging and distinct difficulties are associated with Internet transmission time delays, delay jitter, bandwidth variation and bandwidth limitation between the human operator and remote device.

When the first master-slave mechanical teleoperators were developed by Goertz in 1950s [12], there were no such problems since the human operator was only a few meters away from the manipulator and the links were direct mechanical tape and cable linkages.

Since the early 1960s, telerobots have been applied to space and undersea applications. The distance between operators and robots ranges from a few kilometres to hundreds of thousands of kilometres and the communication links are usually radio. Transmission delays began to be a problem. Experiments reveal that very tiny time delays cause instability in a bilateral force-reflection telerobotic system [13]. For these systems, bandwidth (although limited) is usually guaranteed and time delays are almost constant.

However, data communication through the Internet is a different story. The major difficulties facing Internet-based teleoperation and e-service robotic systems are summarized as follows, among many others:

- *Bandwidth*: an Internet application may face a congested network and thus not be able to exchange data consistently; the available bandwidth is time-varying and the throughput is not guaranteed (*Throughput* is defined as bits of data received by the receiver per second).
- *Delays*: due to innumerable factors, time delays are introduced in the communication path; these delays are time-varying and sometimes even unbounded.
- *Delay jitter*: network load variation and/or switch/link failures may cause large delay variation and packet scrambling.
- *Packet losses*: routers discard packets that overflow their buffer capacities.

These conditions of Internet communications make teleoperation and e-service robotic systems difficult since these applications are *delay-sensitive* and *real-time.* The transmission time delays, delay jitter and unguaranteed bandwidth would give rise to the *instability* problem and lead to great *performance degradation* if they were not compensated properly.

1.3 Current Approaches

To overcome these problems and achieve real-time performance, most present endeavours are devoted to developing new advanced remote *compensation control algorithms*, such as discrete control [13,163], shared control [14,16,164], traded control [15,165,166] and supervisory control [16,156,156,167,168], and innovative *teleoperation interfaces*, such as predictive display or pre-simulation [17-18, 128-131,154-158].

1.3.1 Control Schemes

1.3.1.1 Continuous Command Control

Continuous command control (or called direct control) is the most widely adopted scheme in traditional bilateral master-slave force-reflection teleoperation systems [13, 143, 159-160]. Generally, the control loop is closed continuously by the operator using force and visual feedback. The control scheme is shown in Figure 1.1, where HO represents the human operator (the same for block diagrams hereafter).

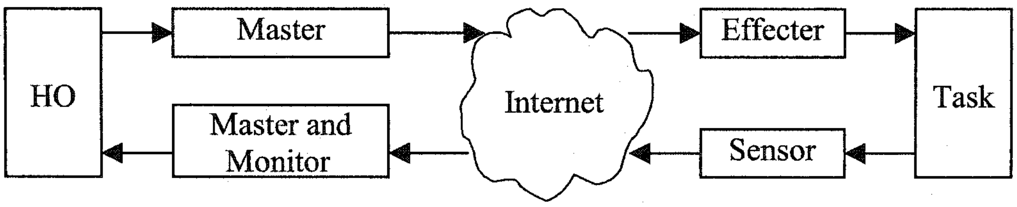


Figure 1.1: Continuous control scheme

This control scheme works fine when there are no time delays in communications. It is shown that time delays in the communication links cause contact instability and great degradation of system performance [11, 13, 161, 162].

1.3.1.2 Discrete Command Control

The earliest version of discrete command control is a "move-and-wait" strategy that was employed in space teleoperation in 1960s [13]. The operator makes a discrete control movement, then stops while waiting a roundtrip time for confirmation that the control action has been followed by the remote robotic device, then sends another discrete movement command, and so on. When repetitive tasks exist, high-level discrete command controls are usually adopted. For high-level discrete command control, the remote subsystem has some degree of autonomy and is able to carry out high-level discrete commands without operator's help [163]. Operators are able to issue individual or a sequence of control commands to the remote robotic device.

1.3.1.3 Shared Control

In shared control, a teleoperation mission is usually spatially decomposed into subtasks in advance. The human operator and the robot's on-board controller control different subtasks of the system simultaneously [14,16,164]. A typical example is the position and force control of a telemanipulator. The human operator is responsible for position control while the on-board controller for force interactions with the remote environment. The key issue of shared control is to decide which subtasks should be assigned to the operator and which to the robot's on-board controller. The diagram of the shared control scheme is shown in Figure 1.2, where RC stands for the robot's on-board controller and HID represents haptic interface devices (the same for the block diagrams hereafter).

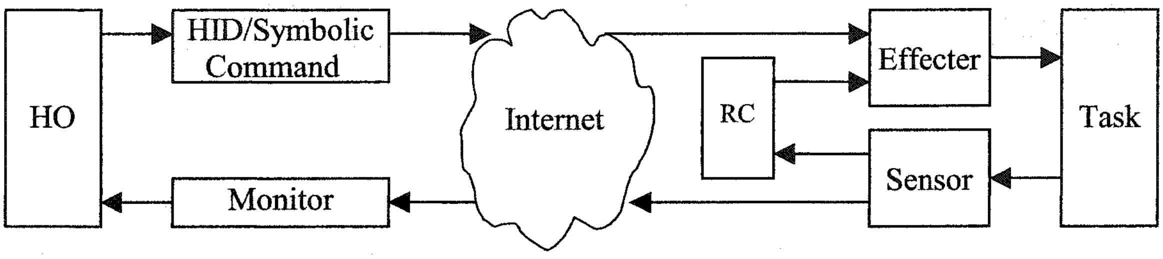


Figure 1.2: Shared control scheme

1.3.1.4 Traded Control

In traded control shown in Figure 1.3, the human operator and the robot's on-board controller take turns to control the system according to the confidence of successful control actions [15,165,166]. For example, during a task execution, the telemanipulator is following a desired trajectory, which is temporally taken care of by the robot's on-board controller under normal conditions. But when the operator notices an obstacle ahead and collision is likely to happen, the human operator takes over the control and helps the telemanipulator to get around the obstacle. Once the telemanipulator passes the obstacle, the operator returns the control to the robot's on-board controller.

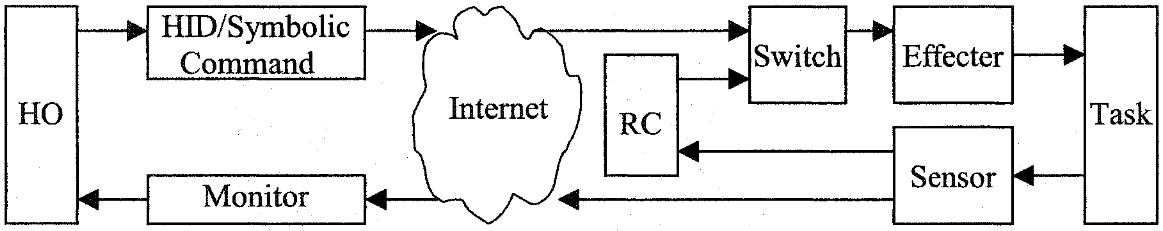


Figure 1.3: Traded control scheme

1.3.1.5 Supervisory Control

This is the most widely used control scheme in Internet-based teleoperation and e-service robotic systems [16,155-156,167,168]. In this control scheme, the remote device operates largely in an autonomous mode and only interacts with the operator when it encounters unexpected situations. High-level control commands are issued from the human operator and executed by the remote subsystem. The human operator and remote environment are decoupled. Thus, there are no direct interactions between the local control station and the remote environment. This scheme avoids the problems of the continuous command control that shifts moment-to-moment sensing and control responsibilities completely to the human operator. As a result, the load on the communication link is reduced. Human operators, acting in a supervisory manner, need only interfere with the remote operation infrequently with a finite set of high-level commands. The block diagram of the supervisory control is given by Figure 1.4.

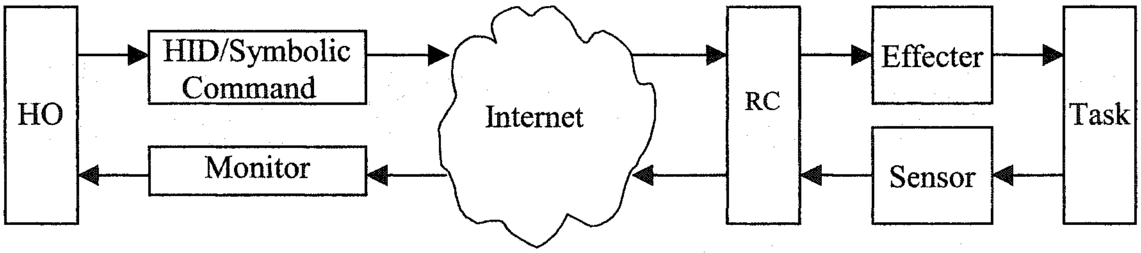


Figure1.4: Supervisory control scheme

1.3.1.6 Other Schemes

Other control schemes, such as non-time based control (event-based control) [169], cooperative control [170], and distributed control [171] etc., are also used in Internet-based teleoperation and e-service robotic systems: the non-time based control is actually a hybrid mode of the supervisory control and the non-time based control; the cooperative control is an extended mode of the traded and shared control; and the distributed control is only one variant of the supervisory control.

1.3.2 Interfaces

Regarding interfaces, the most common scheme to overcome the problem of time delays is predictive display. This technique is considered to provide visual aids to the operator to make control decisions under condition of time delays [128-131, 154-158]. Usually, a virtual model of the remote robotic system on a computer is extrapolated forward in time. There are two ways to do this. The first is based upon current states and time derivatives. The second involves current states and time derivatives, as well as expected near-future control signals, into the model. The interface helps the human operator by predicting "what will happen", given current conditions of the remote robotic system. Here are some examples:

Noyes [128] uses a graphical predictive display of the telerobot overlaid on the video image from the remote site. A line drawing graphical display is drawn at the same location as the real robot is predicted to be after a round trip time delay. With this scheme the operator is provided with immediate feedback of the control actions without having to wait. Noyes reports a more than 50% reduction in task completion time with this predictive display compared to non-predictive display cases.

Conway [129] introduces the concept of a *time clutch* and a *position clutch* to partially take the human operator out of the direct control loop. The time clutch allows de-synchronization of the robot control with real-time inputs from the operator. The effective position of the manipulator is made visible with a predictive display, which is sampled at regular intervals, buffered and continuously sent to the remote site, to be executed autonomously between the sampled instants, while the operator continues to receive feedback through video. The position clutch allows a de-synchronization of the predictive display/simulation from the real robot. The simulated robot is instructed to move to a new location first and only when this is deemed satisfactory is the command sent to update the position of the real robot with its simulated position. In this case, the intermediate trajectory is generated autonomously.

Bejczy [130] uses a graphical simulation of the remote operation to provide a predictive display of the operator's control actions. Harbinger *et al.* [131] also uses predictive graphical displays to estimate the position and orientation of a robot arm relative to moving objects in the task space. In the second case, the moving objects are tracked by a combination of processing of the video images and the data from the range sensors connected to the robot arm. The resulting estimated position is then used in the predictive displays.

1.3.3 Bandwidth Saving

Available bandwidth is a significant factor in limiting the transmission of visual data over Internet communication links between the human operator and the remote device. Even after using image compression the bandwidth demand of video data is still about 3 orders of magnitude higher than all other data channels combined [177]. Thus, bandwidth-saving schemes are mainly developed for visual image transmission. Bandwidth demand is usually reduced by decreasing frame rates, resolutions, numbers of bits/pixels, or by using image compression devices/algorithms [154].

1.4 Motivations for Developing New Transport Protocols

As mentioned in Section 1.3, to overcome the problems of time delays, delay jitter, bandwidth limitation and bandwidth variation, and achieve real-time performance, the majority of current research in this area is focusing on remote compensation control algorithms and interface designs, while the data communications through the Internet are usually treated as given conditions and rarely touched. *From the literature, currently, only the TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) transport protocol is employed for the data transmission between the human operator and the remote device/robot.*

In this thesis, we tackle the major problems in the data transmission between the human operator and the remote device/robot. Specifically, we deal with network transport protocols. Since teleoperation and e-service robotic systems are *semi-reliable*, *delay-sensitive* and *real-time* applications, it is conceivable that data communications in these systems are very different from those in transmitting static data, such as documents and stored images. We start with the question: *Do current transport protocols (TCP and UDP) work for teleoperation and e-service robotic systems*?

Before we answer this question, let us first review the Internet protocol architecture. We know that the Internet protocol architecture is based on a connectionless end-to-end datagram[1] service using the *IP* (Internet Protocol) protocol. Above the IP layer, two transport protocols, i.e., TCP and UDP are available to provide transport services to different applications as shown in Figure 1.5.

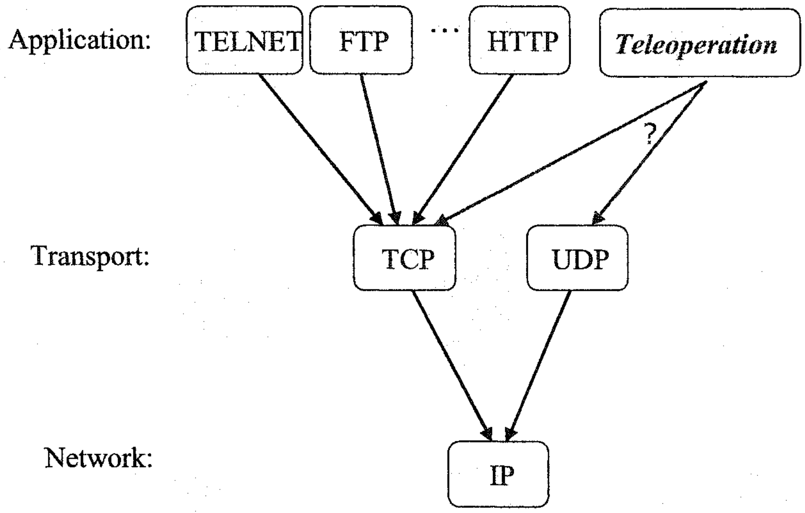


Figure1.5: Internet protocol architecture

TCP was originally designed for reliable static data communications on low-bandwidth, high-error-rate networks [19]. When a packet gets lost or corrupted, it is retransmitted. Therefore, TCP is suitable for applications, such as Telnet, FTP and HTTP etc., because in these cases guaranteed delivery is required and time delays are not of central importance. However, the retransmission mechanism to guarantee reliable delivery in TCP reduces network efficiency, increases average delays and causes great delay jitter. Since teleoperation and e-service robotic systems are delay sensitive, semi-reliable and real-time applications, when data packets are lost, it is more desirable to transmit the most current information rather than simply retransmitting the lost ones. In

[1] Datagram is a self-contained, independent entity of data carrying sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges between this source and destination computer and the transporting network.

addition, the TCP transmission rate varies dramatically with time, which is not suitable for teleoperation and e-service robotic systems either.

Compared to TCP, UDP supplies minimized transmission delays and delay jitter by omitting connection setup process and retransmission [20].

Figure 1.6 shows a comparison of delay and delay jitter between TCP and UDP where a TCP flow and a UDP flow use the same network route. From the simulation results, it is clear that TCP delay varies significantly while UDP delay changes very little. Also, the average delay of UDP is much smaller than that of TCP. Consequently, we can say that UDP outperforms TCP in terms of both delay and delay jitter.

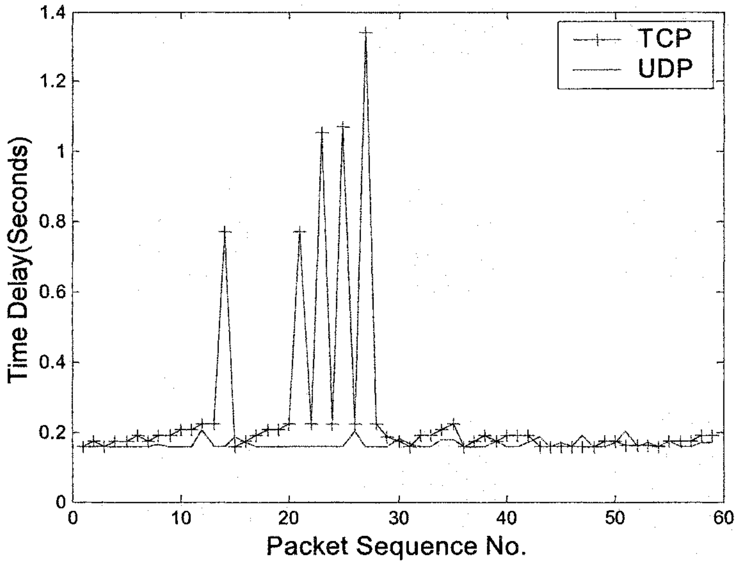


Figure1.6: Comparison of delay and delay jitter between TCP and UDP where a TCP flow and a UDP flow use the same network path

However, UDP is merely a *raw open-loop* protocol without any congestion[2] control mechanism. The UDP source just pushes datagrams into the net at a specified constant rate while the UDP sink accepts incoming datagrams off the net. Neither the source nor the sink has mechanisms to detect network states such as extra bandwidth availability and

[2] Congestion means a situation that the offered load of a data communication path exceeds the capacity of the network.

network congestion. As a result, the UDP source does not know whether its sending rate is proper and is not able to adapt its sending rate to network states.

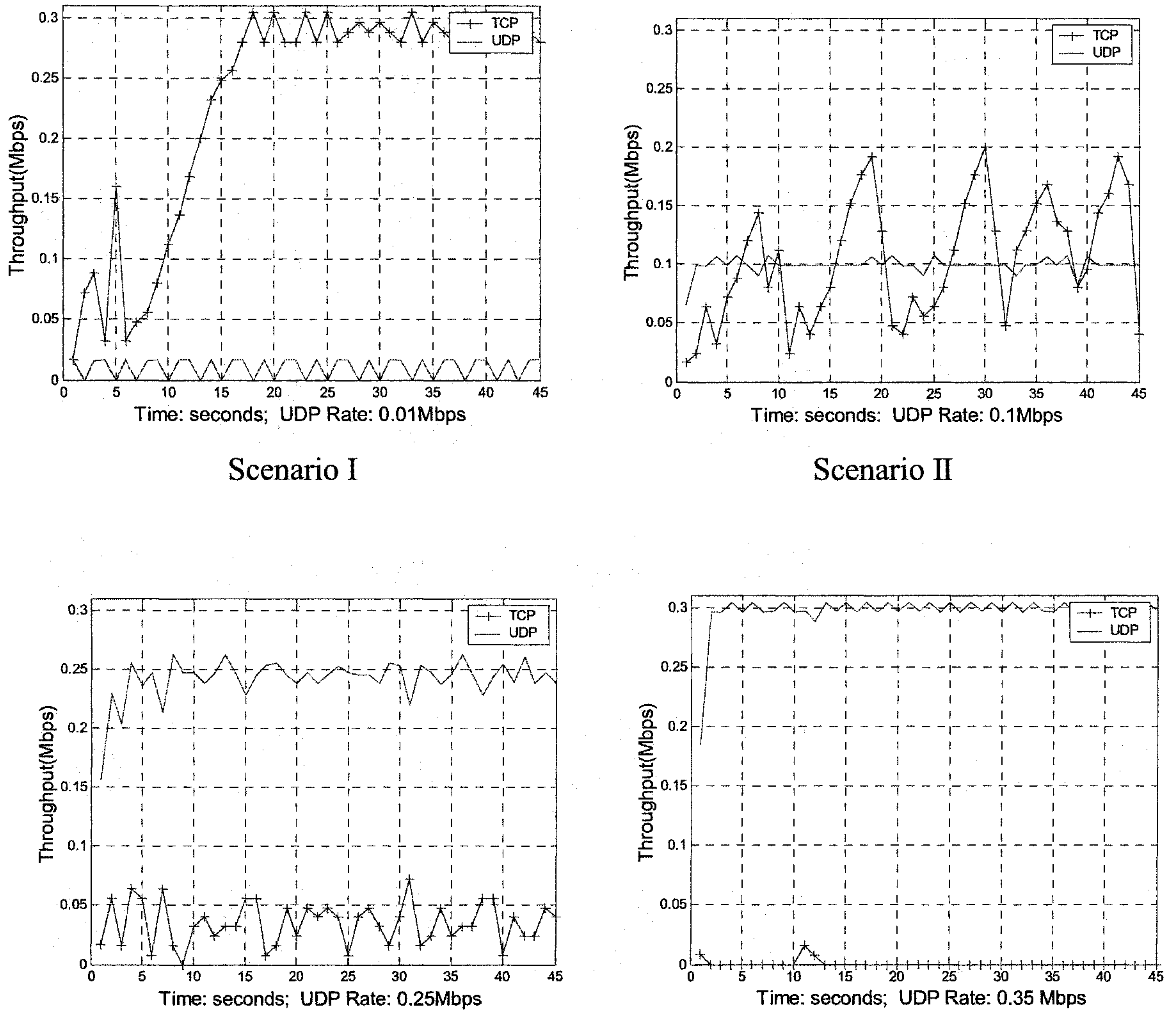
On the contrary, TCP is a *closed-loop* protocol. TCP is capable of probing extra bandwidth availability and detecting network congestion. It increases its transmission speed by increasing the congestion window size (the amount of data to be sent) when there is extra bandwidth available and reduces its sending rate by shrinking the congestion window size when congestion occurs. These features ensure an efficient network while protect the network from *congestion collapse*. As pointed out in [21], one of the keys to the success of today's Internet has been the sophisticated congestion control mechanisms provided by TCP.

While TCP is still employed by the majority (85%~95%) of current Internet applications, there is an increasing number of real-time applications, most of which are based on UDP [21]. The increasing UDP flows are jeopardizing the health and compromising the fairness of today's Internet. As a result, for the Internet as a whole, it would be very prone to be driven to congestion or even collapses by such UDP flows without appropriate congestion control mechanisms. For individual flows, it would be unfair for TCP flows if UDP sources persist in pushing data into the traffic at a high rate while TCP sources back off when congestion occurs; On the other hand, it would be inefficient for UDP to keep staying at a low sending rate when there is extra bandwidth available.

The simulation results (Figure 1.7) show how a TCP flow and a UDP flow share a bottleneck for which the bandwidth is 0.3M.

- In scenario I, where the UDP sending rate (0.01Mbps) is very low, TCP takes extra bandwidth very fast and consumes most bandwidth at the end. It is inefficient and unfavourable for UDP in this case.
- In scenario II, where the UDP sending rate (0.1Mbps) is proper, TCP and UDP share the bottleneck rather fairly and their average throughputs are close.

- In scenario III, where the UDP sending rate is too high (0.25Mbps), TCP starves by shrinking its congestion window size. In this case, UDP is not *friendly* to TCP.
- In scenario IV, where the UDP transmission rate (0.35Mbps) is extremely high, TCP drops its rate to nearly nothing. In this case, the network is collapsed for TCP traffic.



Scenario I

Scenario II

Scenario III

Scenario IV

Figure1.7: UDP affects TCP when a UDP flow and a TCP flow share a bottleneck

In addition, in environments with lower levels of statistical multiplexing or with per-flow scheduling, the delays and loss rates experienced by a flow are partially a function of the flow's own sending rate. A UDP flow with a high sending rate may cause great delays, delay jitter and high loss rates to its own packets.

From above discussions and simulations, we can see that neither TCP nor UDP works for teleoperation and e-service robotic systems. Consequently, we conclude that *new network transport protocols are needed if we allow today's Internet to support teleoperation and e-service robotic systems.*

1.5 Contributions

For data communications in Internet-based teleoperation and e-service robotic systems, as mentioned earlier, current approaches employ one of the two current Internet transport protocols, *i.e.*, TCP or UDP, directly. As shown in Section 1.4, neither TCP nor UDP is appropriate. It is already recognized that new transport protocols are missing [124,125], however, until the time the thesis is written, no new protocols that are optimally designed for teleoperation and e-service robotic systems have been reported in the literature.

In this thesis, we introduce a novel rate-based transport protocol that is specifically developed for Internet-based teleoperation and e-service robotic systems. Since there are three parameters (α, β, γ) that are adjustable based on specific applications in this protocol, we name it the *trinomial* (α, β, γ) *protocol*. Based on the protocol, a new modular platform for e-service mobile robotic systems is proposed and implemented. The main original contributions of this thesis can be summarized as follows:

1 A novel rate-based transport protocol called the trinomial (α, β, γ) protocol:

1.1 In terms of *individual* performances and capacities, the trinomial protocol inherits the advantages of both TCP and UDP, and thus provides much better transport service than TCP and UDP: since there are no retransmission mechanisms, its delay and delay jitter are minimized; in the steady state, its transmission rate is smooth; when available network bandwidth increases/decreases, it adapts its transmission rate to the changes quickly.

1.2 Regarding *social* behaviours, the trinomial protocol behaves properly: it is responsive since it slows down its transmission rate when the network is congested; a trinomial flow achieves a throughput that is reasonably close to that of a TCP flow traveling over the same network path; various trinomial flows reach the same transmission rate no matter how different their initial sending rates may be; it is network efficient since trinomial applications converge to the efficient state regardless of their initial transmission rates. In one word, the trinomial protocol is responsive, inter-protocol fairness (TCP-compatibility) convergent, intra-protocol fairness convergent and efficiency convergent.

Above characteristics of the trinomial protocol are shown theoretically, and evaluated and compared with both TCP and UDP in a wide range of network conditions through simulations. For all evaluations and comparisons, we have conducted simulations for both the RED and DropTail queues. To assess TCP-compatibility of the trinomial protocol, we examine its interaction with different TCP implementations (Tahoe, Reno, NewReno, Sack, Vegas and Fack). The simulation results validate the theoretical findings, proofs, and conclusions.

2 Roundtrip delay (RTT) characterization and estimation:

2.1 RTT characterization: A huge RTT time series collected from the Internet are studied statistically by using both linear and nonlinear methods. The experimental statistical results reveal that the RTT distribution is best fitted by a constant plus a “skewed right” gamma distribution. RTT is not random, but linearly correlated between adjacent and near-adjacent observations. It is also indicated that there are only weak or no nonlinear correlations among RTT observations. Based on these findings, it is concluded that RTT time series can be characterized by linear models in estimating the value of next RTT from past and current observations.

2.2 RTT and RTO (roundtrip timeout) estimation: Due to the multi-structure characteristic of RTT dynamics, the traditional parameter-fixed ARMA (autoregressive moving average) model is not capable of tracking RTT dynamics constantly. In this thesis, we introduce a novel parameter-varying adaptive algorithm for RTT estimation based on the information theory and maximum entropy principle. Since the coefficients of the proposed model are updated every ACK (acknowledgement) received, the model is adaptive and is able to catch up with RTT dynamics quickly. Since the solution to the coefficient is only a polynomial, the computing overhead of updating the coefficients is much small compared to other adopted adaptive algorithms.

3 A new modular platform for e-service mobile robotic systems:

3.1 As the first step towards a real-world network-enabled robotic application system, a new modular platform for e-service mobile robotic systems based on the trinomial protocol is proposed, developed and implemented. The system hardware architecture mainly consists of a commercial Pioneer 2 PeopleBot mobile robot including a Sony PTZ video camera, two sonar arrays, lower bumper switches, and speed and distance encoders. The mobile robot is connected to the Internet by using a pair of wireless adaptors. The system software employs a client-server architecture for robot control and feedback information display. The great benefits of this client-server architecture are that the client application software is insulated from the lowest level details of the mobile robot. As a result, it is very easy to implement and test new advanced teleoperation control algorithms, interface designs and applications on this platform without large programming work.

3.2 This platform is tested on the real Internet. In the experiments, by using any Java-enabled web browser, such as the Microsoft Internet Explorer or

Netscape Communicator, on a regular PC, the users successfully navigated the mobile robot through a laboratory environment. The preliminary experimental results look promising.

1.6 Thesis Overview

The thesis is organized as follows:

Chapter 2 discusses congestion control that is the core of data transport protocols. First, the reasons for congestion control in transport protocols are elaborated: to maintain Internet health, to be fair among different flows, to achieve good transmission performance, and to utilize network resources efficiently. Secondly the two different approaches to congestion control, i.e., the network based mechanism and the end-to-end mechanism (transport protocol approach), are reviewed. Finally, it is argued that the end-to-end approach plays a key role in congestion control for today's Internet.

In Chapter 3, we discuss relevant issues on the development of new transport protocols for Internet-based teleoperation and e-service robotic systems. We first review related work on the two approaches to real-time data transmission: network-level approach and application-level approach. We argue that, for today's Internet, the feasible way is the application-level approach: to develop new transport protocols that are suitable for real-time applications based on current Internet model. Then, we present the key performance metrics for the new transport protocol that is suitable for Internet-based teleoperation and e-service robotic systems: i) it should minimize delay and delay jitter; ii) in the steady state, its transmission rate should be smooth; iii) when available network bandwidth varies, it should adapt to the variation quickly; iv) it should be responsive; v) it should be efficiency convergent; vi) it should be inter-protocol fairness convergent; and vii) it should be intra-protocol fairness convergent.

In Chapter 4, based on the constraints and performance requirements on new transport protocols discussed in Chapter 3, we introduce a novel end-to-end rate-based transport protocol named the trinomial protocol for teleoperation and e-service robotic systems. First we tentatively consider a spectrum of linear transport protocols. The dynamics of the protocols is examined and the system state transitions are visualized as a trajectory through a 2-dimenional vector space. From the visualizations, we derive the constraints on the protocols to be convergent to fairness and efficiency. Then we formally present the trinomial protocol. Its transmission smoothness, efficiency, responsiveness, inter-protocol fairness and intra-protocol fairness are shown theoretically or demonstrated visually.

In Chapter 5, we discuss Internet roundtrip delay characterization and the estimation of RTT and RTO that is an integral component of the implementation of the trinomial protocol. The distribution of RTT data is found best fitted by a constant plus a gamma distribution. The "skewed right" characteristic of RTT distribution reveals that RTT time series is not a Gaussian or uniform white noise. The adjacent and near-adjacent observations are linearly dependent rather than random from the autocorrelation plot, power spectrum plot and mutual information plot. Then we introduce a novel parameter-varying adaptive algorithm for RTT and RTO estimation based on the information theory and maximum entropy principle (MEP). Since the coefficients of the proposed model are updated every ACK (acknowledgement) received, the MEP algorithm is adaptive and is able to track RTT dynamics quickly.

Chapter 6 discusses the implementation issues and performance evaluation of the trinomial protocol through extensive simulation studies. First, it is elaborated how to implement the trinomial scheme in the Internet. Secondly, the trinomial protocol is simulated extensively and compared with both TCP and UDP. To examine TCP-compatibility in a wide range of network conditions, experiments are carried out in various conditions in terms of different combinations of TCP implementations (Tahoe, Reno, NewReno, Sack, Vegas and Fack), queue managements (RED and DropTail), and traffic sizes. For all performance evaluations, simulations are conducted for both RED and DropTail queues. The simulation results validate the theoretical findings, proofs, and conclusions, which are shown in Chapter 4.

As the first step towards a real-world e-service robotic system, in Chapter 7, a new modular platform for e-service mobile robotic systems is introduced. The system enables Internet users to control a mobile robot remotely via the Internet by using any Java-enabled web browsers. The system hardware architecture mainly consists of a commercial Pioneer 2 PeopleBot mobile robot including a Sony PTZ video camera, two sonar arrays, lower bumper switches, and speed and distance encoders. The mobile robot is connected to the Internet by using a pair of wireless adaptors. The system software employs a client-server architecture for robot control and feedback information display. In the system, there are two servers, Video Server and Control Server, and two corresponding clients, Video Applet and Control Applet. The web server is a Linux Apache web server.

Chapter 8 summarizes the work presented in this thesis and discusses possible future research projects as extensions to presented work.

Chapter 2

Transmission Congestion Control

Before we dig into how to develop new transport protocols for Internet-based teleoperation and e-service robotic systems, let us first discuss network congestion control that is the core of network transport protocols. To a large degree, a transport protocol is a transmission congestion control algorithm.

The Internet protocol architecture is based on a connectionless end-to-end datagram service using the IP protocol. The advantages of its connectionless design, flexibility and robustness, have been amply demonstrated. However, these advantages are not without cost: careful design is required to provide good service under heavy load [21]. Proper congestion control (or avoidance) mechanisms should be provided to maintain the stability, fairness and efficiency of the network.

2.1 Needs for Congestion Control

Congestion control refers to a mechanism that enables the source of a flow to match its transmission rate to current available network bandwidth [22]. For the Internet -- a packet-switching[3] best-effort[4] network, when a source starts transmitting data, the available bandwidth between the source and the sink is unknown a priori. If the transmission rate is too high, it results in excessive transmission delays, large variations (jitters) of these delays, and network congestion or even collapse and subsequent data losses, whereas too low transmission rate leaves the network underutilized. Consequently, controlling or avoiding congestion is a critical component in network

[3] Packet-switching: a communications paradigm in which packets (messages or fragments of messages) are individually routed between nodes, with no previously established communication path.

[4] Best-effort: a classification of low priority network traffic for which there is no Service of Quality (such as minimum bandwidth and bounded latency) guarantee.

design and implementation. According to RFC 2914[21], end-to-end congestion control is necessary for the following reasons.

2.1.1 Internet Stability

Congestion control is critical for maintaining the stability of the Internet, i.e., preventing the network from congestion collapse. *Congestion collapse* means the situations that an increase in the network load results in a decrease in the useful work provided by the network.

Historically, the first wide-area networks (WANs) were *circuit-switching*[5] telephone networks. Since these networks carry traffic of a single type, and the traffic behaviour is well known, it is possible to avoid congestion collapse simply by reserving enough resources at the start of each call. By limiting the total number of users, each admitted call can be guaranteed to have enough resources to achieve its performance target, and so there is no congestion. However, resources can be severely underutilized, since the resources blocked by a call, even if idle, are not available to other calls [23].

Early research in computer data networking led to the development of reservationless store-and-forward *packet-switching* data networks [24]. However, these networks are prone to congestion since neither the number of users nor their workload is regulated. The efficiency gained by statistical multiplexing of network resources would be traded off with the possibility of congestion collapse if congestion could not be controlled effectively. The original specification of TCP [19] includes window-based flow control as a means for the receiver to govern the amount of data sent by the sender. This flow control mechanism is used to prevent the receiver from overflow of its data buffer space available for that connection. Nevertheless the flow control mechanism does not help prevent the networks from congestion collapse. The congestion collapse problems were early recognized in 1972 [25]. Matters developed into a crisis around the middle of

[5] Circuit-switching: a communications paradigm in which a dedicated communication path is established between the sender and receiver along which all packets travel. The telephone system is an example of a circuit switched network. Also called connection-oriented.

1980s. In October 1986, the Internet in UC Berkeley experienced a serious congestion collapse. The data throughput dropped from 32 Kbps to 40 bps in a link where two end nodes were separated by only 400 yards [26]. The congestion collapse was found largely due to TCP connections unnecessarily retransmitting packets that were either in transit or had already been received by the receiver. This Internet collapse crisis prompted two pioneering research efforts. Jain *et al.* developed the DECbit congestion control scheme [27,28]. Simultaneously, Jacobson [26] improved the TCP implementation by including a slow-start mechanism and a better timer [19, 29]. The success of Jacobson's congestion control mechanism has prevented the Internet from the congestion collapse resulting from unnecessarily retransmitting packets and brought the peaceful time of today's Internet.

However, that is not the end of the story. In these years, there has been a renewed concern on another congestion collapse. The potential congestion collapse of today's Internet is mainly from undelivered packets when bandwidth is wasted by delivering packets through the network that are dropped before reaching their ultimate destination [21]. The reasons lie on the following facts:

- While TCP is still employed by the majority (85%~95%) of current Internet applications, with the diversification of Internet applications, there are a rapidly increasing number of real-time applications such as video and audio. Most of them are based on UDP [21] in which there is no congestion control mechanism. Given the great heterogeneity, complete decentralization of network management and unbelievable size, the Internet would be easily driven to congestion collapse if no proper congestion control mechanisms were adopted.
- The intermixing of old and new technologies causes increasing mismatch in link speeds. This heterogeneity has resulted in a mismatch of arrival and service rates in the intermediate nodes in the network causing increased queuing and congestion. For example, Giga fibber network links coexist with slow switches. Optical fibber trunks offer bandwidth that is a factor of 10,000 larger than that of earlier circuits. A number of researchers have recognized the critical role of

congestion control in today's high-speed (Wide Area Networks) WANs, since the bandwidth-delay product of a single connection in such networks can be as large as 50 Mbits (1000 Mbps × 50 ms roundtrip delay across the USA) [23]. With such large products, a single source could introduce a transient load large enough to swamp the buffers at switches, leading to packet losses and excessive end-to-end delays for all other hosts on the network.

2.1.2 Fairness

The second reason for congestion control is fairness. Since the Internet is a shared environment and does not currently micromanage utilization of its resources, end nodes are expected to be cooperative by reacting to congestion and adapting their transmission rates properly and promptly to ensure fairness [30].

Fairness is perhaps the most delicate part of Internet congestion/flow control [41]. Many criteria to measure fairness exist, however there is no criterion agreed on by the whole networking community. The contexts of fairness depend on how to define congestion that is a notion related to both user's satisfaction and network load. A detailed and complete discussion of this issue is too involving and beyond this thesis. Readers who are interested in this issue can refer to [41], [57] and [63-65] for details. In this thesis, we adopt the widely accepted context of fairness that: *to be fair, all flows share the same bottleneck should have approximately the same amount of throughput.*

The issue of fairness among competing flows has become increasingly important for several reasons. First, using window scaling [61], individual TCP's can use high bandwidth even over high-propagation-delay paths. Second, with the growth of the web, Internet users increasingly want high-bandwidth and low-delay communications, rather than the leisure transfer of a large file in the background. The growth of best-effort traffic that does not use TCP underscores this concern on fairness between competing best-effort traffic in times of congestion. For example, the popularity of the Internet has caused a proliferation in the number of TCP implementations. Some of these may fail to implement the TCP congestion control mechanisms correctly because of poor implementation [21,62]. Others may deliberately be implemented with congestion control algorithms that are more aggressive in their use of bandwidth than other TCP implementations: this would allow a vendor to claim to have a "faster TCP". The logical consequence of such implementations would be a spiral of increasingly aggressive TCP implementations, or increasingly aggressive transport protocols, leading back to the point where there are effectively no congestion control mechanisms and the Internet is chronically driven to collapse.

2.1.3 Performance

The third reason for a flow to adopt congestion control is to optimize its own performance regarding throughput, delay and loss. In some circumstances, for example, in environments of high statistical multiplexing, the delays and loss rate experienced by a flow are largely independent of its own sending rate. However, in environments with lower levels of statistical multiplexing or with per-flow scheduling, the delays and loss rate experienced by a flow are partially a function of the flow's own sending rate. Thus, a flow can use congestion control to limit the delays and losses experienced by its own packets.

However, it should be noted that in an environment like current best-effort Internet, concerns regarding congestion collapse and fairness with competing flows limit the range of congestion control behaviours available to a flow.

2.1.4 Efficiency

The fourth reason for congestion control is efficiency. Congestion control mechanisms should result in higher overall utilization of the network. A congestion control mechanism should have the capacities of determining the available bandwidth based on the knowledge of network states and the application should then use the bandwidth efficiently to maximize the quantity of the delivered service to the user.

2.2 Approaches to Congestion Control

There are two principal approaches to the congestion control in today's Internet: network-based (router-based) approaches and end-to-end approaches.

2.2.1 Router-Based Mechanisms

Two classes of router algorithms are related to congestion control: *scheduling* and *queue management* algorithms. To a rough approximation, scheduling algorithms determine which packet to send next and are used primarily to manage the allocation of bandwidth among flows while queue management algorithms manage the length of packet queues by dropping packets when necessary or appropriate. While these two router mechanisms are closely related, they address rather different performance issues [42].

Scheduling is the most direct control over allocated bandwidth to individual flows since it controls the order in which individual packets are served. Two of the most popular scheduling algorithms are: *First-in-first-out* (FIFO) [50] and *Weighted Fair Queuing* (WFQ) [51,55]. FIFO is widely used due to its simplicity. With the FIFO algorithm, all packets experience the same queuing delay in average. However, it does not provide isolation for individual flows, thus a misbehaved flow can affect delivered service to other flows. It could also result in unfairness among well-behaved flows [44, 45] and packet clumping [46]. The WFQ algorithm or any of its functional equivalents tries to allocate the available bandwidth evenly based on the specified weight among all active flows [45]. WFQ provides a high level of isolation against misbehaved flows and reduces the effect of packet clumping. Since the implementation of WFQ requires per-flow states at each switch along the path, there are some scalability concerns on deploying it over wide area networks. Thus there have been rough functional equivalents of it that essentially trade the number of required states with obtained fairness [47, 48]. Scheduling alone cannot prevent substantial packet losses. Because of sudden change in network load and the bursty nature of traffic, the switch is forced to drop any packet that cannot be sent. Adequate buffer space enables the switch to absorb short-term extra-load without dropping a large number of packets. Thus the switch requires a queue management scheme to control the usage of buffer space as we describe next.

The *queue management* algorithm recommended by IETF (Internet Engineering Task Force) for congestion control is the so-called *Random Early Detection* (RED) [43] queue, which is an active queue management algorithm compared to the traditional technique, *Drop-Tail* queue. The traditional Drop-Tail queue for managing router queue lengths is to set a maximum length (in terms of packets) for each queue, accept packets for the queue until the maximum length is reached, then reject (drop) subsequent incoming packets until the queue length decreases because packets from the queue have been transmitted. This technique is known as *tail-drop,* since the packet that arrives most recently (i.e., the one on the tail of the queue) is dropped when the queue is full. This method has two important drawbacks. The first one is so-called *Lockout.* In some situations drop-tail allows a single connection or a few flows to monopolize queue space, preventing other connections from getting room in the queue. This *Lockout* phenomenon is often the result of synchronization or other timing effects. The second one is *Full-Queue.* The drop-tail discipline allows queues to maintain a full (or, almost full) status for long periods of time, since tail-drop signals congestion (via a packet drop) only when the queue has become full. It is important to reduce the steady-state queue size to minimize delays, and this is perhaps the most important goal of queue management. These problems of the DropTail queue scheme inspire another queue management scheme, RED queue, which is an active queue management algorithm. In contrast to the DropTail queue algorithm, which drops packets only when the queue buffer is full, the RED algorithm drops arriving packets probabilistically. The probability of drop increases as the estimated average queue size grows. RED responds to a time-averaged queue length, not an instantaneous one. Thus, if the queue has been mostly empty in the "recent past", RED will not tend to drop. On the other hand, if the queue has recently been relatively full, indicating persistent congestion, newly arriving packets are more likely to be dropped. RED effectively controls the average queue size while still accommodating bursts of packets without excessive losses. RED's usage of randomness breaks up synchronized processes that lead to the *Lockout* phenomena.

2.2.2 End-to-End Mechanisms

In the end-to-end approach, end nodes are expected to adapt to network states properly by adjusting their transmission rates. End nodes have knowledge of network states from feedback signals such as packet drop or delay magnitude. Usually, end nodes do not know the deployed scheduling algorithms (e.g. WFQ or FIFO) in network switches because of the heterogeneity of the Internet. End nodes cannot make any specific assumptions about the underlying queue management algorithm either.

Currently, there are two ways to adjusting transmission rate in congestion control: the *window-based* mechanism [19, 28, 52, 57] and the *rate-based* mechanism [49,53,56]. In the window-based scheme, a source directly controls the number of packets in transit by adjusting a window that is the upper bound for the number of packets in flight. Thus, gap between consecutive packets may vary. In the rate-based scheme, the source controls its transmission rate by adjusting the gap between every two consecutive packets, called *inter-packet-gap* (IPG). The window-based scheme is more popular because it is easier to implement. The rate-based scheme requires a relatively accurate timer, which is usually based on the estimation of next roundtrip time/delay (RTT) [49]. We note that most real-time applications are rate-based, which is usually determined by the nature of the applications.

2.3 Importance of the End-to-End Approach

Since the core of the Internet (i.e. scheduling, queue management and feedback) is relatively fixed, the end-node adjustment is the most accessible component of the congestion control loop. The proposed WFQ scheduling and RED queue management algorithms have not been widely implemented in today's Internet. Most of current routers in the Internet still implement the FIFO scheduling and the DropTail queuing [22]. Thus, the end-to-end approach plays a key role in congestion control for today's Internet.

Actually, the danger of today's Internet congestion collapse is due primarily to the increasing deployment of open-loop applications not using end-to-end congestion control[6]. Even more destructive would be best-effort applications that increase their sending rates in response to an increased packet drop rate. Thus, effective end-to-end congestion control by the end nodes is crucial to maintain the health of the Internet [30].

Furthermore, even if router-based congestion mechanisms are widely adopted in the future, end-to-end congestion control is still necessary. We should note that while RED can manage queue lengths and reduce end-to-end latency in the absence of end-to-end congestion control, RED will be able to reduce packet dropping *only* in an environment that continues to be dominated by end-to-end congestion control [42]. As recommended by the IETF RFC 2914 [21], the choice between the router-based mechanism and end-to-end mechanism does not have to be an either/or decision. Congestion collapse can be prevented by the use of effective end-to-end congestion control mechanisms by some of the traffic, and the use of router-based mechanisms from the network for the rest of the traffic. End-to-end mechanisms are needed in end nodes to complement router-based congestion control algorithms.

[6] Currently, a lot of papers propose that future routers detect and penalize flows that are not employing acceptable end-to-end congestion control [30]

Chapter 3

Development of New Transport Protocols

3.1 Approaches to Real-Time Transmission

Internet real-time applications require guaranteed transmission qualities, such as bounded delays and minimum bandwidth, to ensure real-time performance. Current best-effort, packet-switching Internet does not provide quality of service (QoS[7]) guarantee [85]. However, lack of support for QoS has not prevented the rapid growth of real-time applications on the Internet. In this section, we first give a brief review of current approaches to the data transmission for real-time applications and then discuss the relevant issues of the development of new transport protocols for Internet-based teleoperation and e-service robotic systems.

3.1.1 Network-Level Approach: Resource Reservation

For the Internet to provide appropriate QoS for real-time applications, the intuitive idea is to go beyond current best-effort service model and allow the network to reserve resources for real-time flows. This approach is usually referred to as *integrated service* because it incorporates QoS requirements and provides transmission guarantees for real-time applications. This idea is similar to the open-loop congestion control paradigm. The network implements admission control mechanisms during the call setup phase and allocates part of its resources to each flow based on client's requirements. During a session[8], the network should ensure that the allocated resources are available for the flow

[7] QoS: the performance properties of a network service, possibly including throughput, transit delay, priority. Some networks allow packets or streams to include QoS requirements.
[8] Session: a lasting connection between a user (or user agent) and a peer.

and the end nodes should ensure that their offered traffic is within the presented profile during the establishment phase [118-120].

The *Internet Stream Protocol, Version 2* (ST2) [116] is an early experimental connection-oriented resource-reservation protocol that operates at the same layer as the connectionless IP. It has been developed to support efficient delivery of real-time data streams to single or multiple destinations in applications that require guaranteed QoS. The main application areas of the ST2 protocol are the real-time transport of multimedia data, e.g., digital audio and video packet streams, and distributed simulation/gaming, across the Internet. It ensures that real-time packets are delivered within their deadlines, that is, at the time when they need to be presented. This facilitates a smooth delivery of data that is essential for time-critical multi-media applications, but can typically not be provided by the best-effort IP communication. Just like IP, ST2 actually consists of two protocols: ST for the data transport like IP and SCMP (the Stream Control Message Protocol) for all control functions like ICMP (Internet Control Message Protocol). ST2 can be used to reserve bandwidth for real-time streams across network routers. However, this reservation has to work together with appropriate network access and packet scheduling mechanisms in all nodes running the protocol.

The *ReSerVation Protocol* (RSVP) [119] is another well-known reservation protocol for providing *integrated service* over the Internet. Like ST2, RSVP allows bandwidth to be reserved for specific applications. RSVP sends its packets using existing IP, i.e., RVSP is not a new version of IP and so it works better with existing networks than ST2. However, to be of any use, new routers (or router software) must still be installed. Unlike ST2, RSVP is initiated from the receiving end, i.e., a user asks for bandwidth. RSVP works by telling the routers to inspect traffic *en* route and gives priority to those packets that have requested bandwidth. Not surprisingly, this is a computationally intensive task – RSVP's main drawback.

The main challenge in supporting the integrated service model is the need for maintaining per-flow states at each intermediate router in the network. Many researchers involved in the area have realized some of the resulting difficulties associated with providing integrated service and per-flow reservation of resources. Some of these difficulties are as follows [22]:

- *Scalability*: It results in high memory requirement and processing overhead for each intermediate router to maintain per-flow states. Thus, there are some scalability concerns on how to deploy such a mechanism in a router with a large number of simultaneous flows.
- *Flexibility*: The integrated service framework only provides a small number of pre-specified service classes. This set of classes does not allow more qualitative definition of service models that are more appropriate and useful for end nodes.
- *Need for Implementing RSVP Signalling*: Most of the hosts in today's Internet do not support RSVP signalling.

These problems motivate a new service model, called *differentiated service* [54], which is currently being developed at IETF (Internet Engineering Task Force). The differentiated service proposes different classes of services within the Internet in a scalable and flexible fashion. To accommodate scalability, per-flow states are kept at edge routers where the number of flows is relatively small and the router is able to manage the complexity and resource requirement. Edge routers are mainly in charge of 1) assigning each packet to a particular class of services based on the requested service by the host, and 2) conditioning the offered traffic by that host. Core routers simply serve each packet based on the specified class of services in the packet header. Currently, there are two main frameworks under active discussion within the *diffserv* group at IETF. They are the *expedited forwarding PHB* (Per-Hop-Behaviour) [121] and *assured forwarding PHB* [122].

3.1.2 Application Level Approach: New Transport Protocols

3.1.2.1 Arguments for New Transport Protocols

The integrated service and differentiated service models, which reserve resources from the network to guarantee QoS, are still under research and evolving. They need supports from the network and thus depend on a modification or updating of current Internet model. No people can predict when this will occur. Even in a network that supports reservation, different users that fall into the same class of service or share a reservation still interact as in the best-effort networks [22]. Furthermore, it is also believed that even if these services really become widely available, there will remain a significant group of users who are interested in using real-time applications at low cost.

Therefore, for today's Internet to support real-time applications, the most feasible way is the application-level approach: to develop new transport protocols that are fit for the transmission requirements of real-time applications based on current best-effort network model.

Another important argument for new transport protocols lies on the system design principle called the end-to-end argument.

Choosing the proper boundaries among functions is perhaps the primary activity of the system design. Proper function placement is the key to system performance, feasibility, and reliability. In the teleoperation or e-service robotic system, a distributed system that includes communications, it is apparent that there is a list of functions each of which might be implemented in any of several ways: by the communication subsystem; by the end nodes; as a joint venture; or perhaps redundantly, each doing its own version. In reasoning about the choice, the requirements of the application provide the basis for a class of arguments, which go as follows:

The application-specific functions in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. [117].

For instance, to secure transmission of data, the argument here is threefold. First, if the lower-level network subsystems perform encryption and decryption, it must be trusted to manage securely the required encryption keys. Second, the data will be clear to every intermediate node and thus vulnerable as it passes into the target node and is fanned out to the target application. Third, the *authenticity* of the message must still be checked by the application. On the other hand, if the endpoint (the application) performs end-to-end encryption, it obtains its required authentication check, it can handle key management to its satisfaction, and the data is never exposed outside the application. Thus, to satisfy the security requirements of the application, there is no need for the low-level communication subsystems to provide automatic encryption.

3.1.2.2 Related Work

Gilge *et al.* [174] propose an early end-to-end scheme, called network-integrated video encoding, for video transmission over best-effort networks. They use explicit feedback from the receiver to regulate the transmission rate of the source. In this approach, congestion control is implemented by the encoder based on the feedback from the receiver.

Kanakia *et al.* [87] build on Gilge's model and introduce an architecture in which the feedback is generated by a congested switch along the path. The bottleneck switch communicates its queuing delays back to the source. A controller at the source exploits this information to control the output rate of a MPEG [88] encoder before packet losses occur at the bottleneck due to queue overflow.

Turletti *et al.* [89, 90] propose a multiplicative increase, multiplicative decrease algorithm to periodically adjust the output rate of a H.261 [91] codec based on the reported loss rate from the receiver.

Jeffy *et al.* [92] design an unreliable connection oriented transport protocol on top of UDP/IP called the Multimedia Transport Protocol (MTP). MTP monitors the local packet transmission buffer to detect congestion. Once a packet is discarded due to buffer overflow, the protocol signals the application to reduce its data rate. This scheme is only suited for local area networks where congestion could result in increased media access latencies at the local adaptor.

Chen *et al.* [93] propose a datagram protocol, called VDP, to integrate video/audio streams with the web. The adaptation mechanism of VDP degrades or upgrades the quality of the stream based on client's feedback that reports frame drop rate due to CPU bottleneck or loss rate due to network congestion. However, they do not describe a specific strategy for rate adaptation.

Cen *et al.* [94] present the SCP (standard communication protocol) protocol for media streaming. SCP deploys a modified version of TCP's congestion control mechanism that performs Vegas-like rate adjustment in the steady state.

All above approaches either ignore congestion control or do not examine various aspects of the proposed congestion control mechanisms, such as inter-protocol fairness, intra-protocol fairness or network stability, etc.

Several different end-to-end approaches explicitly addressing congestion control have been proposed in the recent literature (e.g. [30], [40], [41], [49], and [68-76]).

Jacobs and Eleftheriadis [68] present an algorithm that emulates TCP's congestion control mechanism. Like TCP, it is based on window size adjustment but without retransmitting lost packets. The sender maintains the congestion window just as if it would be for a TCP connection under the same conditions of losses and delays. The sender estimates the proper sending rate based on the window size. As noted in [73], its drawback is inflexibility. Although the idea is straightforward, since this algorithm strictly adheres to TCP window dynamics, it would be difficult to modify it to take into account timeliness requirements of real-time applications. Moreover, this scheme still inherits TCP's bursty behaviour [49].

Rejaie *et al.* ([22], [49] and [81]) describe a rate adaptation protocol (RAP), which employs the additive-increase, multiplicative-decrease (AIMD) algorithm for sending rate control. RAP treats packet losses as the only signal of network congestion, and uses timeouts and gaps in the sequence space to detect losses. Because RAP adheres to the

AIMD algorithm, its sending rate changes too drastically. In addition, this scheme is simulated and targeted on a future scenario in which SACK TCP [82] and the RED [43] queue management schemes will be widely deployed. However, today's Internet, TCP-Reno [83] is still the dominant TCP implementation and very few RED queue management schemes have been employed [80].

Sisalem and Schulzrinne [71] develop a scheme called the loss-delay based adjustment algorithm (LDA) for adapting sending rate of the sender. It relies on the end-to-end *Real Time Transport Protocol* (RTP) to feedback information on packet losses and round trip times (RTT's). For a unicast application, during the periods without losses the sender increases its sending rate by an additive increase rate, which is estimated using the loss, delay and bottleneck bandwidth values included in the feedback reports in RTCP packets. The limitation of this scheme is that it is based on the RTP protocol. In addition, in the scheme, there are several tunable parameters whose values are not specified by the authors and have to be set by the user [73].

Padhye *et al.* [74] model TCP throughput as a function of packet loss rate, packet size, roundtrip time (RTT) and timeout (RTO). Based on the model, Floyd *et al.* ([53], [73-76] and [80]) propose an equation-based sending rate control (TFRC) protocol that gives the maximum sending rate for rate-based applications. The great advantage of the TFRC protocol is that it has a much lower variation of throughput over time compared to TCP, making the sending rate much smoother and thus ameliorate delay jitters. However, as pointed out in the IETF working documents, the TFRC protocol is still 'work in progress'. The protocol also has some limitations. First, it is based on the average loss rate and delay observed during the lifetime of a connection while adaptation decisions should be made on the most current delay and loss rate. In the mean time, the TFRC protocol does not specify how the sending rate evolves when there are no loss events.

3.2 Needs for New Transport Protocols for Teleoperation and E-Service Robotic Systems

All above protocols reviewed in Section 3.1.2.2 are Internet multimedia oriented. While both the multimedia system, and the teleoperation and e-service robotic system are real-time applications, their implementation mechanisms and requirements on data transmission are different.

First, the data transmission of Internet multimedia systems is usually semi-realtime in implementation. Specifically, a buffer is used at the receiving end. The data presentation at the receiving side is largely a playback process. The effects of delays and delay jitter can be meliorated through this buffer mechanism. The incoming data is to refill the buffer and it would work well should the data tank were not empty. The receiver is always behind the sender with a period of time delay. However, for teleoperation and e-service robotic systems, this buffer mechanism does not work since feedback presentation and motion control should be as current as possible. Data cannot be held a while and then transmitted.

Second, their requirements on transmission rate are different. For multimedia systems, the data should arrive at the receiving side at a regular basis. Consequently, the transmission rate should not vary quickly or significantly over the duration of the application. In other words, the transient response of transmission rate to network bandwidth variations could not be quick and significant. For teleoperation and e-service robotic systems, the transmission rate should be smooth in the steady state, however, when available network bandwidth increases/decreases, the transmission rate should adapt to the change quickly. For example, when the network is heavily loaded or some intermediate paths are broken, only low transmission rate is allowed. Thus, a supervisory control scheme is appropriate and there is no need for highly frequent transmission. On the other hand, when there is large extra bandwidth available, the transport protocol should take extra bandwidth quickly and thus high sending rate can be employed. As a result, a fine direct control mode can be used.

Until today, there have been *no* transport protocols, which satisfy above requirements of teleoperation and e-service robotic systems.

3.3 Transmission Control Protocol (TCP)

For a new transport protocol, the essence is that the protocol incorporates a proper end-adjustment strategy for the sake of congestion control. The Internet Engineering Task Force (IETF) initialized the RFC2914 [21] requiring that any new transport protocols must integrate congestion control mechanisms. The key goal of congestion control is to be *TCP-compatible* (We will give a formal definition in next section). In addition, the dynamics of today's Internet is mainly shaped by the TCP protocol and its applications. Consequently, it is natural to study TCP before presenting a new transport protocol to the Internet.

As mentioned in previous chapters, TCP is a transport layer protocol that provides two main services: reliable delivery over the best-effort IP layer and congestion control. TCP is perhaps the most successful protocol in congestion control so far and is certainly the most popular transport protocol. Its various aspects have been studied extensively during the last decade. The end-to-end congestion control mechanisms of TCP have been a critical factor in the robustness of the Internet. As pointed out in [21], one of the keys to the success of today's Internet has been the elegant congestion control mechanism, i.e., the famous *additive increase, multiplicative decrease* (AIMD) [26,31] algorithm, provided by TCP.

3.3.1 AIMD Algorithm

TCP does not get much help from the network layer (IP layer). Most Internet routers do not send back explicit congestion notifications to the sender, but instead simply drop packets when there are too many packets to fit in the router buffer. The TCP sender finds out packet losses based on the *acknowledgement* (ACK) received from the receiver and a *timer*. For the Internet, packets get lost for two main reasons: they are damaged in transit, or the network is congested somewhere on the path when there is insufficient buffer capacity. On current wired network paths, losses due to damage are rare (<< 1%) so it is rather safe to say that packet loss is due to congestion in the network (wireless network is an exception) [26]. In TCP, packet losses are considered the only indication of congestion and trigger TCP's congestion control mechanisms. After detecting a packet loss-- the sign of congestion, a *slow-start* algorithm is employed to gradually increase the amount of data in-transit. Slow-start ends when the sender's congestion window is greater than the slow-start threshold. Then the AIMD algorithm takes effect. The AIMD (α,β) algorithm is expressed as

On no congestion:

$$W_{t+1}=W_i+\alpha \ , \tag{3.1}$$

On congestion:

$$W_{t+}=\beta\, W_i, \quad 0<\beta<1, \tag{3.2}$$

where *On no congestion* means there is no timeout (RTO) (All packets sent are received by the receiver) and the congestion window size (W_i) is increased by α per roundtrip time (RTT); and *On congestion* refers to that a RTO event occurs (packets get lost) and the congestion window size (W_i) is reduced to βW_i immediately. In current TCP implementation, α=1 and β=0.5.

The performance of TCP has been extensively and intensively studied [26, 31-35]. The AIMD algorithm is shown to converge to an efficient and fair state in a distributed manner [27].

3.3.2 TCP Implementation

TCP has evolved during the last decade and various modifications have been proposed to improve its congestion control mechanism. However, the core of its congestion control mechanism, i.e. the AIMD algorithm, remains intact and most of the refinements are to improve the efficiency of the error control mechanism.

The version of TCP with an *AIMD* algorithm and a *slow-start* mechanism is known as TCP *Tahoe*, which is Jacobson's version [26]. In this version, TCP tightly couples congestion and error control mechanisms, i.e., packet losses signal congestion, and trigger a back-off in transmission rate and a retransmission of the lost packet. The main challenge is to detect losses as soon as possible. TCP Tahoe deploys a timeout mechanism for loss detection. This may result in a long delay and decrease in throughput.

TCP *Reno* [83] incorporates a fast-retransmission mechanism to use duplicate ACKs as a signal for congestion and improve the efficiency of error control.

Selective Acknowledgment (*SACK*) is a recent revision of TCP that further improves the performance of TCP's error control mechanism in that it prevents unnecessary retransmission and prevents TCP from losing its ACK clocking.

The most recent enhancement in TCP's performance is *FACK* TCP [37]. The FACK algorithm is designed to use with the proposed TCP SACK option. It also decouples congestion control from other algorithms, such as data recovery, to attain more precise control over the data flow in the network.

TCP *Vegas* [38] is another revision of TCP. The main contribution of Vegas is the improvement of the congestion avoidance mechanism in TCP. TCP Vegas compares observed throughput with expected throughput and adjusts its transmission rate accordingly before congestion occurs. As a result, it reduces the number of losses due to congestion as well [38,39]. The main concern is whether TCP Vegas can coexist with TCP Reno and Tahoe in harmony. The congestion avoidance mechanism in Vegas slows down its transmission rate before congestion occurs while Reno and Tahoe increase the rate until they experience loss. Thus, Vegas is likely to achieve less share of resources when it coexists with Reno or Tahoe.

3.3.3 TCP Transmission Rate Modelling

To develop a new rate-based *TCP-compatible* protocol, we should first have a mathematical model of TCP transmission rate.

The so-called *slow-start* process of TCP is actually not slow since TCP's sending rate is doubled every roundtrip time (RTT) in the slow-start phase and thus the increase in the congestion window size is exponential [26]. This means the congestion window opens quickly enough to have a negligible effect on performance, even on links with a large bandwidth-delay product [26]. Consequently, we do not consider the effects of the *slow-start* process when we model TCP transmission rate.

To simplify the problem, we consider the deterministic AIMD model described in [84]. The deterministic AIMD (α, β) model assumes that a packet is dropped each time when the congestion window reaches Wpackets, as shown in Figure 3.1. We define a *congestion epoch* as a period beginning with a congestion window of $(1-\beta)W$ packets. The congestion window (W_t) is increased additively by α packets per round-trip time (R) up to a congestion window of W, when a packet is dropped. The congestion window (W_t) is then decreased multiplicatively back to $(1-\beta)W$ at the end of the congestion epoch [58].

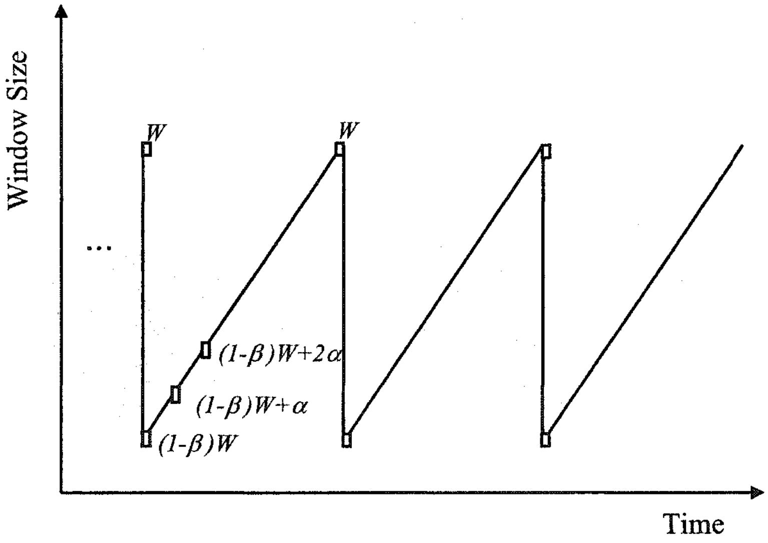


Figure 3.1: AIMD (α, β)'s congestion window in the steady state

From Figure 3.1, each congestion epoch consists of

$$\frac{\beta}{\alpha}W+1, \tag{3.3}$$

roundtrip times (R's). Let S denote the sending rate in packets per R, and λ denote the sending rate in packets per second. The average sending rate over one congestion epoch is

$$S=\frac{2\text{-}\beta}{2}W, \tag{3.4}$$

packets per roundtrip time, or

$$\lambda=\frac{2\text{-}\beta}{2R}W, \tag{3.5}$$

packets per second where R represents the value of roundtrip time. This gives a total of

$$(\frac{\beta}{\alpha}W+1)S \;\text{¡Ö}\; \frac{\beta(2\text{-}\beta)}{2\alpha}W^2, \tag{3.6}$$

packets in one congestion epoch, with one of these packets dropped at the end of the congestion epoch. The packet drop rate p is therefore

$$p=\frac{2\alpha}{\beta(2-\beta)W^2+\alpha(2-\beta)W}, \tag{3.7}$$

or, using an approximation,

$$p\;\text{¡Ö}\frac{2\alpha}{\beta(2-\beta)W^2}, \tag{3.8}$$

Thus, we can get the following:

$$W\;\text{¡Ö}\sqrt{\frac{2\alpha}{\beta(2-\beta)p}}, \tag{3.9}$$

We define the *response function* of a congestion control algorithm as the sending rate λ. Substituting (3.9) into (3.5), we can get

$$\lambda=\frac{\sqrt{\alpha(2-\beta)}}{R\sqrt{2p\beta}}, \tag{3.10}$$

From (3.10), we can see that the response function of the ARMA algorithm is a function of the decrease/increase parameters, α and β, the roundtrip time, R, and the packet loss rate, p. Applying AIMD (1, 0.5), which is the actual TCP congestion control mechanism, to the response function shown in (3.10), we get TCP's response function, λ_{TCP}, as

$$\lambda_{TCP}=\frac{\sqrt{3}}{R\sqrt{2p}}, \tag{3.11}$$

The equation (3.11) is further elaborated using the stochastic TCP model by Mathis *et al.* [37] and Padhye *et al.* [74]. Their stochastic TCP model assumes that packets are dropped with a random probability, and takes into account the role of TCP's retransmission timeouts. Although the stochastic TCP model gives a more accurate description of TCP, it is too complicated. Generally, the deterministic AIMD model works well [58]. Thus, we adopt the deterministic AIMD model for TCP modeling in this thesis.

3.4 Design Issues and Goals

To develop a new transport protocol for teleoperation and e-service robotic systems, the key metrics that should be taken into account are as follows.

3.4.1 Transport Quality

With the rapid diversification of services provided by the Internet, it is impossible to have a protocol general enough to support any needed types of services. Different applications have different requirements, such as reliability, delays, and bandwidth, on data transmission. It is difficult to build support for all of them into one protocol.

For traditional web applications, for instance, guaranteed and correct delivery is required and mandatory. Consequently, *reliability* is the first concern while delivery time delays and delay jitter are less important. The delivery order of fragmented packets is not so relevant either given that it could be reorganized at the receiving side. These requirement specifications caused TCP in which retransmission and error recovery mechanisms are integrated to ensure guaranteed delivery [19, 59]. As shown in the simulation results in Chapter 1, the costs of reliable delivery are great time delays and delay jitter.

On the contrary, Internet-based teleoperation and e-service robotic systems are *semi-reliable*, *delay-sensitive,* and *real-time* applications. *Semi-reliable* means that a packet is not necessary to be retransmitted should it got lost or corrupted. Compared to traditional web applications, the primary requirement is not a *guaranteed-delivery* service, but a service that minimizes *delays* and *delay jitter* in the delivery of packets. The real-time scene information and control commands should be delivered to the corresponding receivers as quickly as possible. The validity of timeliness is prevailing. Thus, if a data packet gets lost somewhere along the path, it is more suitable to transfer a packet of the most current information rather than simply re-transmit the old one. As for the transmission rate, in the steady state, the sending rate should be smooth, however, when available network bandwidth increases/decreases, the transmission rate should adapt to the changes quickly since *throughput* is especially precious for teleoperation and e-service robotic systems. Consequently, the transport protocol should be able to detect extra network bandwidth. Apparently, neither TCP nor UDP fits in these transport requirements.

3.4.2 Internet Health

To prevent the Internet from congestion collapse, the new protocol should respond to the network dynamics properly. In other words, it should have a mechanism to detect whether the network is congested and be *responsive* to congestion signals. *A protocol is responsive means it slows down its transmission rate when congestion occurs* [21].

It should be noted that many of the proposed schemes for congestion control cannot be deployed on today's Internet since the required scheduling or feedback components are not supported in the Internet. As mentioned in previous chapters, most current routers implement the FIFO scheduling and DropTail queuing. The proposed Explicit Congestion Notification (ECN) mechanism [60] is just in its experimental stage. As a result, end nodes can only rely on implicit signals, such as packet losses, delay variations, and/or throughput variations, to determine whether the network is congested.

The potential for the congestion collapse of the Internet is due to the flows that do not use responsive end-to-end congestion control. *Any form of congestion control that successfully avoids a high sending rate in the presence of high packet drop rate should be sufficient to avoid congestion collapse from undelivered packets* [21].

3.4.3 Inter-Protocol Fairness

Equitable sharing of bandwidth among different-protocol flows (inter-protocol fairness) depends on the fact that all flows are running compatible congestion control algorithms. Today's Internet traffic is dominated by TCP-based applications, such as HTTP, FTP, Telnet, and SNMP etc. The robustness of TCP is shown theoretically and demonstrated by the success of the Internet. TCP is already there and it is infeasible to expect a whole-scale change of TCP. Consequently, it is crucial that new transport protocols perform *TCP-compatible* congestion control to be fair to TCP.

TCP is a moving target and its behaviours may change substantially with network parameters (e.g. RTT, loss rate, available bandwidth). For example, under heavy load TCP's congestion control mechanism diverges from AIMD and starts a timer-driven mode when only one packet per RTT is sent. TCP might also become bursty in the absence of sufficient statistical multiplexing over long-delay paths. Thus, designing a TCP-compatible congestion control algorithm over any time scale and a wide range of network conditions is challenging.

The widely accepted definition of TCP-compatibility is: *A TCP-compatible flow is a flow that behaves under congestion like a flow produced by a conformant TCP. A TCP-compatible flow is responsive to congestion notification, and in the steady state uses no* *more bandwidth than a conformant TCP running under comparable conditions (drop rate, RTT, MTU, etc.)* [21, 42].

It is convenient to divide flows into three classes: 1) TCP-compatible flows, 2) flows that are responsive but not TCP-compatible, and 3) unresponsive flows, i.e., flows that do not slow down when congestion occurs. The last two classes contain aggressive flows and pose significant threats to Internet performance.

3.4.4 Intra-Protocol Fairness

In addition to inter-protocol fairness, the new protocol should be intra-protocol fair.

The first widely used metric is the *max-min fairness* [66]. It is defined as the ratio of the minimum average flow throughput to the maximum average flow throughput. The max-min fairness measure lies in the interval [0, 1]. A fairness value of zero indicates that *at least* one flow receives no bandwidth at all. A value of one is achieved in case of an equal distribution of bandwidth.

The second widely adopted metric is called the *equality fairness* that is developed by Jain *et al.* [63]. This fairness index is applied to the normalized throughput of the flows of a protocol as

$$f(x) = (\sum_{i=1}^{n} x_i)^2 / (n \sum_{i=1}^{n} x_i^2) , \tag{3.12}$$

where a system allocates resources to n contending users and x_i is the resource allocated to the i^{th} user. $f(x)$ is the fairness index which belongs to [$1/n$, 1]. If all users get the same amount of resource allocation, *i.e.* x_i*'s* are all equal, then the fairness index $f(x)$ is 1, which indicates 100% fair. As the disparity increases, fairness decreases. A scheme favouring only one selected user will lead to a fairness index equal to $1/n$. Compared to the max-min fairness index, the advantage of this metric is that it is dimensionless and independent of scale.

3.4.5 Efficiency

The most direct way to characterize the efficiency of a transport protocol is the ratio of protocol *throughput* to link bandwidth. Since the throughput that a flow can achieve is largely determined by the design constraints regarding network congestion collapse and fairness with competing flows, the efficiency can be improved only by protocol implementation in a narrow range.

In implementation, there are two aspects to the problem of achieving good protocol efficiency. The first one is to send data at minimum transmission cost. This has to do with how the protocol packet itself is organized internally. For example, the implementation should minimize the header of the packet. The second is how the protocol implementation is integrated into the host *operating system* [67].

A real important issue concerning efficiency is that the protocol should be *efficiency convergent*. In order to guarantee convergence to efficiency, the protocol should react correctly to the feedback signal by moving in the right direction. That is, when the system requires the protocol to decrease transfer rate, the protocol should ensure that the total load will not increase; and when the system asks the protocol to increase, the total load will at least not decrease [27].

3.5 Summary

In this chapter, we discuss the relevant issues preparing for the development of new transport protocols for Internet-based teleoperation and e-service robotic systems in next chapter.

We first review the related work on the two approaches to real-time data transmission in the Internet. The Internet is a best-effort packet-switching network without QoS guarantee. However, real-time applications such as teleoperation and e-service robotic systems require guaranteed bandwidth and bounded delays. The first approach to overcoming this dilemma is to go beyond current best-effort service model and allow the

Internet to provide integrated or differentiated service, such as ST2, RSVP, AF PHB, and EF PHB etc. However, these approaches need more or less supports from the network and are still under research. It is also believed that even if these approaches really become widely available, there will remain a significant group of users who are interested in using real-time applications at low cost. Therefore, for today's Internet, the feasible way is to develop new end-to-end transport protocols that are suitable for real-time applications based on current Internet model. Currently, there is quite a lot of research in this area. The most famous and successful protocols are such as LDA, RAP and TFRC etc. However all these protocols are Internet multimedia system oriented and not quite fit in the requirements of teleoperation and e-service robotic systems. So far, there have been no such transport protocols reported in the literature. Consequently, we conclude that new transport protocols are needed that are specifically designed for teleoperation and e-service robotic systems.

Then, we explore TCP transmission rate modelling by employing the deterministic model and derive TCP response function, which is found to be a function of roundtrip time and packet drop rate. The TCP response function will be used for the protocol development and design in next chapter.

Finally, we present the key metrics for the new transport protocol for Internet-based teleoperation and e-service robotic systems. To minimize delays and delay jitter, the new protocol should have no retransmission mechanisms. As for transmission rate, in the steady state, its sending rate should not change so much. It should be able to detect network states such as extra bandwidth availability and congestion. When there is a change of available network bandwidth, it should be able to adapt to this change and get to the new steady state quickly. To maintain Internet health, the new protocol should be responsive. It should also be inter-protocol fair (TCP-compatible), intra-protocol fair and network efficient.

Chapter 4

The Trinomial Transport Protocol

As mentioned in previous chapters, in today's Internet, routers play a relatively passive role in congestion control and traffic adaptation. Most of current router implementations drop packets when the router buffer is overflowed. End nodes (transport protocol) detect network states, i.e., whether the network is underloaded (no congestion) or overloaded (congestion), based on packet losses and/or delay variations. The transport protocol can only rely on the *binary* information feedback (no congestion or congestion) to adjust transmission rate properly [36]. In this chapter, based on current best-effort, packet-switching Internet model, we introduce a novel end-to-end rate-based protocol for teleoperation and e-service robotic systems.

4.1 A Class of Linear Transport Protocols

First, let us tentatively consider a spectrum of linear transport protocols as

$$\textit{Increase}:\ x(t+1) = b_I(t)x(t) + a_I(t)\,, \tag{4.1}$$

$$\textit{Decrease}:\ x(t+1) = b_D(t)x(t) + a_D(t)\,, \tag{4.2}$$

where *Increase* refers to the transmission rate increasing policy when there is no congestion; *Decrease* refers to the decreasing policy when congestion occurs; $x(t)$ is the transmission rate at time t; and $a_I(t)$, $b_I(t)$, $a_D(t)$, and $b_D(t)$ are coefficients that are independent of $x(t)$. Next, for simplicity, we drop the independent variable t for all the coefficients

To examine the dynamics of a protocol and have a better idea how a protocol works, it is helpful to review the system state transitions as a trajectory through a vector space. Next, we illustrate the dynamics of the class of protocols shown in (4.1) and (4.2) visually using the techniques introduced by Chiu and Jain [27]. For simplicity, we consider a two-user case that can be viewed in a 2-dimensional space. From the visualization, we derive the constraints on the protocols to be convergent to fairness and efficiency.

4.1.1 Vector Representation of Protocol Dynamics

As shown in Figure 4.1, a 2-user resource allocation $\{x_1(t), x_2(t)\}$ can be represented as a point (x_1, x_2) in a 2-dimensional space. The horizontal axis represents allocations to user 1, and the vertical axis represents allocations to user 2. All allocations for which $x_1+x_2 = X_{goal}$ are efficient allocations, where X_{goal} is the maximum capability of the Internet. Thus, the straight line that corresponds to $x_1+x_2 = X_{goal}$ is marked "*efficiency line.*" All allocations for which $x_1=x_2$ are 100% fair. This corresponds to the straight line marked "*fairness line.*" The two lines vertically intersect at the point $(X_{goal}/2, X_{goal}/2)$ that is the *optimal point.* In terms of fairness (intra-protocol fairness) and efficiency convergence, the goal of the transport protocol is to bring the system to the optimal point regardless of the starting position.

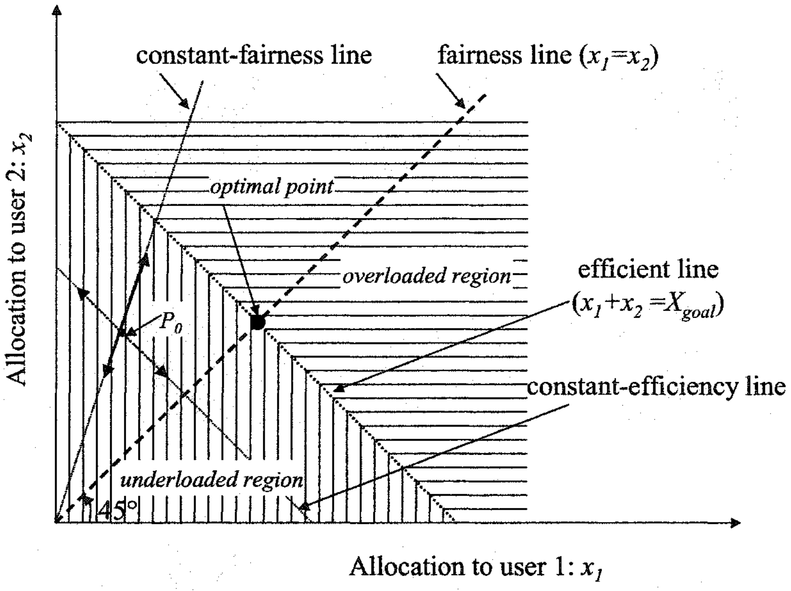


Figure 4.1: Vector representation of fairness and efficiency for a two-user case

All points below the *efficiency line* represent an "*underloaded*" system and ideally the protocol would ask users to increase their load. Similarly, all points above the *efficiency line* represent an "*overloaded*" system and the protocol would ask users to decrease their load.

The fairness (intra-protocol fairness) at any point (x_1, x_2) is given by $f(x_1,x_2)=\frac{(x_1+x_2)^2}{2(x_1^2+x_2^2)}$. Note that multiplying both allocations by a factor does not change fairness. Thus, all points on the line joining a specific point to the origin have the same fairness as the point. Similarly, all points on the line parallel to the *efficiency line* have the same efficiency. For example, for a random starting point P_0 in the *underloaded region* and left above the *fairness line*, as shown in Figure 4.1, the allocation prefers x_2 to x_1 and the system is inefficient. If the allocation moves towards or away from the origin along the *constant-fairness line*, only allocation efficiency changes and it does not affect fairness. Similarly, if the allocation moves along the *constant-efficiency line*, only fairness changes and it has no impacts on efficiency.

4.1.2 Convergence Constraints

The constraints on the spectrum of protocols given by (4.1) and (4.2) are imposed by the efficiency and fairness convergence conditions. We use Figure 4.2 and 4.3 to illustrate these constraints for a 2-user case.

First, we consider the situation that the starting point (P_H) is in the overloaded region. As shown in Figure 4.2(a), the starting point is above the *efficiency line*. Congestion occurs and the protocol asks the users to decrease their transmission rates. Thus, for convergence to efficiency, it is necessary to ensure that next state $x(t+1)$ moves into the shaded area as shown in Figure 4.2(a).

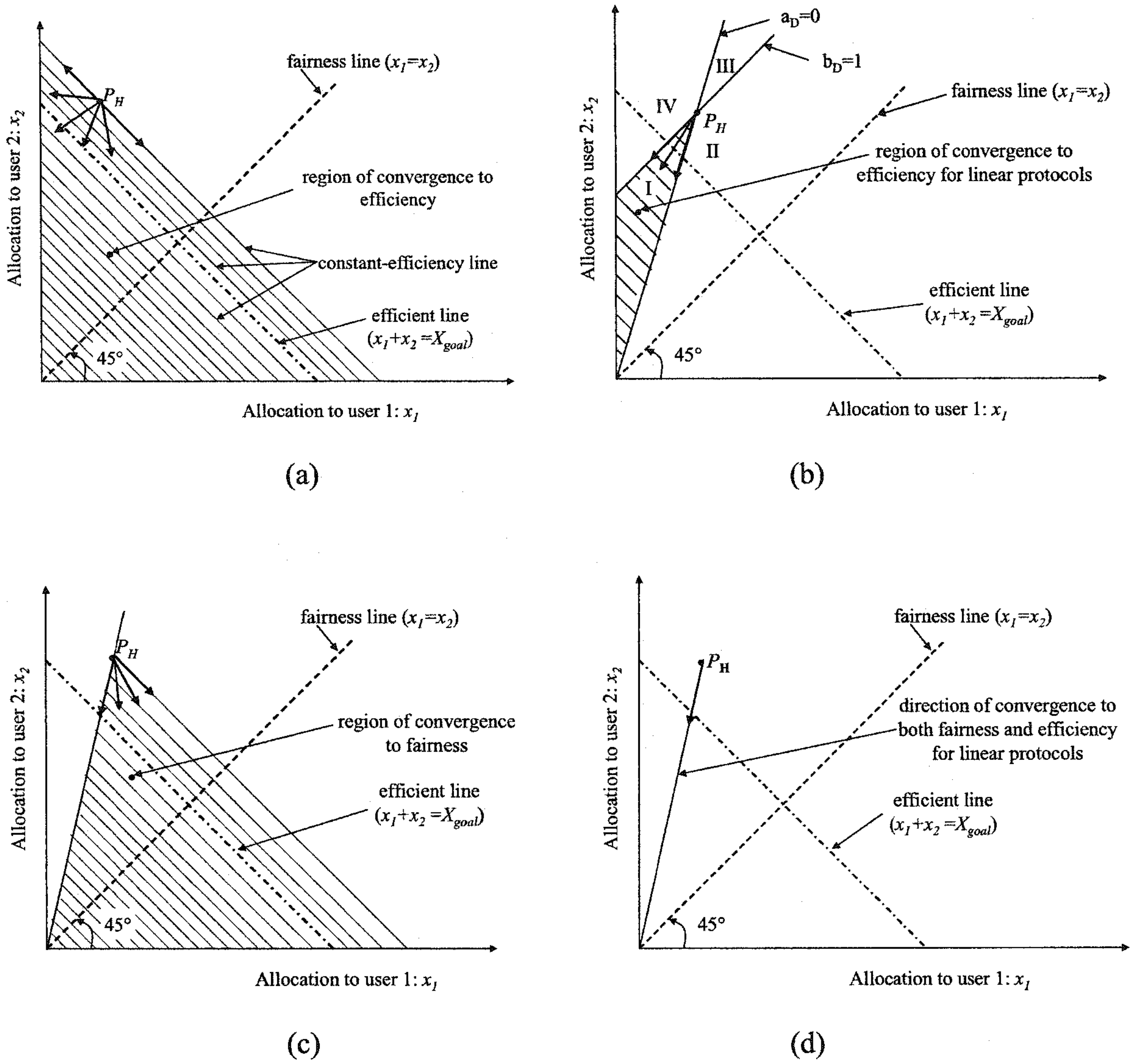

fairness line (x_1=x_2)
P_H
region of convergence to efficiency
constant-efficiency line
efficient line (x_1+x_2 =X_{goal})
45°
Allocation to user 2: x_2
Allocation to user 1: x_1
(a)
a_D=0
b_D=1
III
IV
II
I
P_H
fairness line (x_1=x_2)
region of convergence to efficiency for linear protocols
efficient line (x_1+x_2 =X_{goal})
45°
Allocation to user 2: x_2
Allocation to user 1: x_1
(b)
fairness line (x_1=x_2)
P_H
region of convergence to fairness
efficient line (x_1+x_2 =X_{goal})
45°
Allocation to user 2: x_2
Allocation to user 1: x_1
(c)
fairness line (x_1=x_2)
P_H
direction of convergence to both fairness and efficiency for linear protocols
efficient line (x_1+x_2 =X_{goal})
45°
Allocation to user 2: x_2
Allocation to user 1: x_1
(d)

Figure 4.2: Constraints on the decrease policy for the protocol to converge to both fairness and efficiency

In the two-dimensional vector space, the new state vector $x(t+1)$ is a sum of two vectors corresponding to a_D and $b_D x(t)$. a_D is represented by a 45° line through $x(t)$. This is shown in Figure 4.2(b) by the line marked b_D=1. All future states corresponding to b_D=1 lie on this line. Points to the left of this line can be reached if and only if we choose b_D>1. Similarly, points to the right of this line can be reached if and only if

b_D<1. The second vector corresponding to $b_D x(t)$ is represented by the line marked a_D=0 as shown in Figure 4.2(b). If we choose a_D=0, the state $x(t+1)$ will lie on this line. Points to the left of this line can be reached by choosing a_D<0. Similarly, points to the right of this line can be reached by a_D>0. Depending upon the values of a_D and b_D, the set of reachable states of $x(t+1)$ will lie in one of the four regions (I, II, III, and IV) formed by the two lines: a_D=0 and b_D=1.

From (4.2) we know $x(t+1)-x(t)=(b_D-1)x(t)+a_D$. Because the feedback is binary, the users can only know whether the network is *overloaded* or *underloaded.* The users do not know how much the X_{goal} could be and how far P_H is from the *efficiency line.* Thus, to ensure that the next movement is a decrease of transmission rate, i.e., $x(t+1)-x(t)\leq 0$, we should make sure $b_D \leq 1$ and $a_D \leq 0$, which corresponds to region I, because $x(t)$ could be any value. Consequently, if $x(t+1)$ is located in region I, the shaded region in Figure 4.2 (b), it is sufficient to ensure a decrease in transmission rate. If we choose parameter values of b_D and a_D corresponding to other regions, it is not guaranteed that transmission rate is decreased.

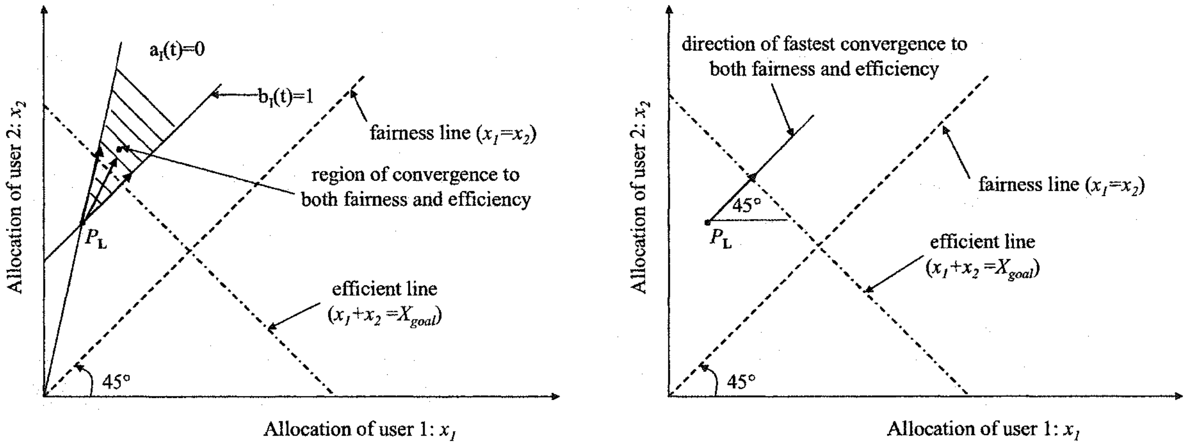
Because all the points on the line passing through P_H and the origin have the same fairness as $P_{H,}$ and all the points in the region between this line and the *fairness line* have higher fairness than P_H, the next movement should at least go into the shaded area as shown in Figure 4.2(c) for fairness convergence.

Combining all the restrictions, the region for the protocols to converge to both fairness and efficiency is given by the intersection of the shaded regions shown in Figure 4.2 (a), (b) and (c). The intersected region is actually the line segment from P_H to the origin as shown in Figure 4.2(d). As a result, the policies *on congestion* that would satisfy all these constraints are those that move the operating point along the line towards the origin, i.e., a_D=0 and 0<b_D<1.

It should be noticed that these decrease policies do not change fairness. Consequently, it is necessary that the *increase* policies *on no congestion* should improve fairness.

For the *increase* policies *on no congestion* given by (4.1), similarly, let us look at a staring point (P_L) in the *underloaded* region. As shown in Figure 4.3(a), P_L is below the *efficiency line*. No congestion occurs and the protocol asks the users to increase their transmission rate. It is necessary to ensure $x(t+1)-x(t)=(b_I-1)x(t)+a_I \geq 0$ for convergence to efficiency. For the same reason as above, we can derive the result that $b_I \geq 1$ and $a_I \geq 0$. These constraints form the shaded region as shown in Figure 4.3(a).

In the feasible region shown in Figure 4.3(a), the line corresponding to $b_I = 1$ is the fastest direction to converge to both fairness and efficiency since it is vertical to the *efficiency line* and closest to the *fairness line*. Consequently, it is reasonable to believe that the line, $b_I = 1$, is the optimal direction to increase transmission rate as shown in Figure 4.3 (b)



$a_I(t)=0$
$b_I(t)=1$
fairness line (x_1=x_2)
region of convergence to both fairness and efficiency
P_L
efficient line (x_1+x_2=X_{goal})
45°
Allocation of user 2: x_2
Allocation of user 1: x_1
direction of fastest convergence to both fairness and efficiency
45°
P_L
fairness line (x_1=x_2)
efficient line (x_1+x_2=X_{goal})
45°
Allocation of user 2: x_2
Allocation of user 1: x_1

(a) (b)

Figure 4.3: Constraints on the increase policy for the protocol to converge to both fairness and efficiency

In summary, in order to fulfill the requirements of optimal convergence to both efficiency and fairness for the spectrum of protocols given by (4.1) and (4.2), the following constraints should be satisfied [27]:

for *increase* policy, $a_I \geq 0$ and $b_I = 1$, i.e.,

$$x(t+1) = x(t) + a_I \text{, where } a_I \geq 0, \tag{4.3}$$

for *decrease* policy, a_D=0 and 0<b_D<1, i.e.,

$$x(t+1) = b_D x(t) \text{, where } 0<b_D<1, \tag{4.4}$$

4.2 The Trinomial Protocol

From (4.3) and (4.4), we can see that a_D and b_I are fixed, and only a_I and b_D are designable.

Note that a_I and b_D can be constants or functions of time, given $a_I \geq 0$ and 0<b_D<1 are satisfied. a_I and b_D actually determine convergence speed. Larger values of a_I and b_D cause a faster convergence but larger overshoot and oscillation while smaller values of a_I and b_D give rise to a slower convergence but less overshoot and oscillation. It is interesting to find out that the oscillation (or smoothness) of the transmission rate in the steady state is mainly determined by the magnitude of b_D while the transient response speed of the protocols is largely governed by a_I.

Based on these observations, and inspired by and extending Jin *et al.*'s work [95], in this section, we describe a novel end-to-end rate-based transport protocol, which we call the trinomial (α,β,γ) protocol, for teleoperation and e-service robotic systems. In this protocol, the sender adjusts its sending rate according to (4.5) and (4.6).

$$\textit{Increase}: \; S_t = S_0 + (\frac{t}{\alpha})^\gamma \quad \alpha \geq 1, \textit{ and } \gamma \geq 0 \tag{4.5}$$

$$\textit{Decrease}: \; S_{t+} = (1-\beta)S_t, \quad 0 < \beta < 1 \tag{4.6}$$

where *Increase* refers to the increase in sending rate as a result of the receipt of acknowledgements (there is no congestion); *Decrease* means the decrease in sending rate on the detection of packet losses (congestion occurs); t is the number of roundtrip times (RTT's); S_t is the sending rate (packets/RTT) at instant t; S_0 is the initial sending rate (packets/RTT) after the last decrease; and , α, γ,and β are constants.

It is clear that the decrease policy given by (4.6) meets with the constraint given by (4.4). By differentiating (4.5), we can get $S_t = S_{t-1} + \frac{r}{\alpha}(\frac{t}{\alpha})^{\gamma-1}$. Thus the increase policy (4.5) satisfies the constraint given by (4.3).

It should be mentioned that nonlinear algorithms different from (4.1) and (4.2) are also possible. However, nonlinear schemes tend to be sensitive to system parameters and difficult to implement [28,31]. That is why we still stick with linear protocols.

4.2.1 Transmission Smoothness and Transient Response Speed

The three parameters (α, β, γ) play different roles and should be determined by the requirements of specific applications and other constraints on transport protocols.

α: it appears in the denominator of an exponential term. The larger value of α, the smoother the transmission rate locally in the steady state (A → B in Figure 4.4). However, too large α leads to a slow response to network bandwidth increase;

γ: it determines how fast the protocol probes and takes extra bandwidth. It specifies the transient response speed of the protocol. Larger γ means a quicker transient response as well as larger overshoot and oscillation in transmission rate;

β: it determines the cutback step magnitude of transmission rate when congestion occurs (the height from C to D in Figure 4.4). Thus, it determines the global smoothness of the transmission rate in the steady state.

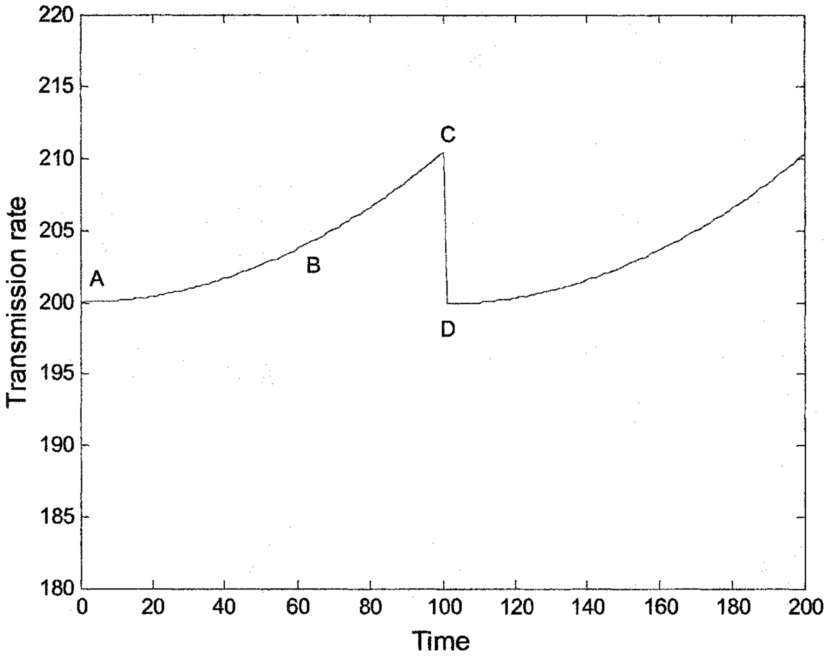


Figure 4.4: Illustration of the roles of α, β, and γ

The key idea underlying the trinomial protocol is that by a proper combination of α, β and γ, the transient response of the trinomial protocol to network bandwidth variation is quick while the transmission rate is globally smooth in the steady state. For instance, as demonstrated in the simulations shown in Chapter 6, by choosing $\beta = 0.05$ and $\gamma = 3$ (the value of α is determined by the constraint given by 4.18 that will be discussed next), the transmission rate of the trinomial protocol is much smoother than that of TCP in the steady state while the transient response to bandwidth increase/decrease is fast.

4.2.2 Responsiveness

As discussed in previous chapters, to prevent the Internet from congestion collapse, the new protocol should respond to the network dynamics properly. To be specific, it should be *responsive* to congestion signals. *A protocol is responsive means it slows down its transmission rate when the network is congested.*

The trinomial protocol is responsive because it slows down its sending rate according to (4.6) once packets are detected lost, which is the signal of congestion.

4.2.3 Inter-Protocol Fairness

As mentioned earlier, since Today's Internet traffic is dominated by TCP-based applications, we only show how the trinomial protocol coexists with TCP to evaluate the inter-protocol fairness. By the widely accepted context of TCP-compatibility (TCP-friendliness), a TCP-compatible flow is a flow that behaves under congestion like the one produced by a conformant TCP. A TCP-compatible flow is responsive to congestion notification, and in the steady state uses no more bandwidth than a conformant TCP running under comparable conditions (drop rate, RTT, MTU, etc.) [21,42]. Thus, for a protocol to be TCP-compatible, the transmission rate λ must be no more than λ_{TCP}, which is given by (3.11), i.e.,

$$\lambda \leq \lambda_{TCP} = \frac{\sqrt{3}}{R\sqrt{2p}}, \tag{4.7}$$

Based on this definition, next we derive the constraints on the parameters (α, β and γ) for the trinomial protocol to be TCP-compatible.

4.2.3.1 Deterministic Model

First, we consider the deterministic Internet model described in [58] where loss events occur periodically with a fixed probability p. The increase and decrease in sending rate are deterministic as shown in Figure 4.5. Each epoch has the same duration (N_d number of RTT's). t_b is the time right after the last packet drop and t_e is the time right before another packet drop. H is the total number of packets sent in each epoch (between two successive drops). S_m (packets/RTT) is the maximum sending rate (at t_e) and S_0 (packets/RTT) is the initial sending rate (at t_b) right after the last drop.

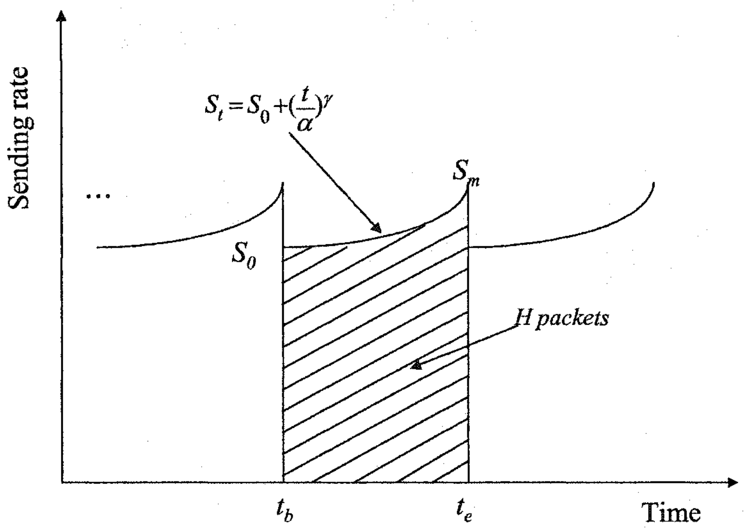


Figure 4.5: Deterministic Internet model

According to the decrease rule (4.6), we have:

$$S_0 = S_{t_b} = (1-\beta)S_m, \tag{4.8}$$

The duration (N_d) of each epoch in RTT is:

$$N_d = t_e - t_b = \alpha(\beta S_m)^{1/\gamma}, \tag{4.9}$$

and the number of packets sent (H) in each congestion epoch is given by:

$$H = \int_0^{N_d} S_t dt = \int_0^{N_d} ((1-\beta)S_m + (\frac{t}{\alpha})^r)dt$$

$$= (1-\beta)S_m N_d + \frac{N_d^{r+1}}{(\gamma+1)\alpha^\gamma}, \tag{4.10}$$

Substituting (4.9) into (4.10), we have:

$$H = \alpha(1-\beta)\beta^{(1/\gamma)} S_m{}^{((\gamma+1)/\gamma)} + \frac{\alpha}{\gamma+1}(\beta S_m)^{((\gamma+1)/\gamma)}, \tag{4.11}$$

The average sending rate λ (packets per second) is the number of packets sent in each epoch (H) divided by the duration of the epoch in seconds (T_dR), where R is the roundtrip time (RTT). The packet loss probability is $p=1/H$. Thus, we can get:

$$\lambda = \frac{H}{N_d R} = \frac{1}{pRN_d} , \tag{4.12}$$

By the definition of TCP-compatibility and (4.12), we have:

$$1.5N_d^2 \geq H , \tag{4.13}$$

Combining (4.8), (4.9), (4.11) and (4.13), we have:

$$\alpha \geq \frac{(r+1-r\beta)S_0^{((\gamma-1)/\gamma)}}{1.5(r+1)(\beta)^{(1/r)}(1-\beta)^{((r-1)/r)}} , \tag{4.14}$$

4.2.3.2 Stochastic Model

Because the stochastic Internet model gives a more accurate description of the real Internet, in this section, we consider the stochastic model where congestion (loss) events occur randomly with a fixed probability p. Let H_i donate the number of packets sent in the i^{th} epoch up to but not including the first packet lost. The probability that H_i takes the value of k (k is the number of packets that are sent successfully before the first loss event takes place in the i^{th} epoch) is given by [95]

$$\begin{aligned} P[H_i = k] &= (1-p)^k p \\ &\approx pe^{-pk}, \quad for\ p \ll 1,\ k = 0, 1, 2, \cdots , \end{aligned} \tag{4.15}$$

Let N_{di} be the duration of the i^{th} epoch in roundtrip time (RTT) and $N_{di} = H_i / \overline{S}_i$, where $\overline{S}_i$ is the average sending rate in the i^{th} epoch. Thus, the total increase in sending rate (ΔS_i) in the i^{th} epoch from the beginning to the time that the first loss event occurs is

$$\Delta S_i = (\frac{N_{di}}{\alpha})^{\gamma} = (\frac{H_i}{\alpha \overline{S}_i})^{\gamma} , \tag{4.16}$$

It is very involving to derive $\overline{S}_i$ because H_i and $\overline{S}_i$ are inter-dependent. We use the time-average sending rate $\overline{S}$ to approximate $\overline{S}_i$, thus $\Delta S_i = (\frac{H_i}{\alpha\overline{S}})^\gamma$. The expectation of ΔS_i is [95]:

$$\begin{aligned} E[\Delta S_i] &= \sum_{x=0}^{\infty} \Delta S_i P[H_i = k] \\ &\approx \sum_{k=0}^{\infty} (\frac{H_i}{\alpha\overline{S}})^\gamma P[H_i = k] \\ &\approx \sum_{k=0}^{\infty} (\frac{k}{\alpha\overline{S}})^\gamma pe^{-pk} \\ &\approx \int_0^{\infty} (\frac{x}{\alpha\overline{S}})^r pe^{-px} dx \\ &= \frac{\Gamma(r+1)}{(\alpha p\overline{S})^r}, \end{aligned} \tag{4.17}$$

We notice that, under the deterministic model, H_i=$1/p$, and $N_{di} = H_i / \overline{S}$. Thus, for the deterministic model, we have $E[\Delta S_i] = (\frac{1}{\alpha p\overline{S}})^r$. To be TCP-compatible, the realization should work for both the deterministic and stochastic model. Combining with (4.14) [95], we get the constraint on the parameters (α, β and γ) as:

$$\alpha \geq \frac{(r+1-r\beta)\Gamma(r+1)^{(1/r)} S_0^{((\gamma-1)/\gamma)}}{1.5(r+1)(\beta)^{(1/r)}(1-\beta)^{((r-1)/r)}}, \tag{4.18}$$

For all the combinations of α, β, and γ for which (4.18) is satisfied, the trinomial protocol is *TCP-compatible*.

4.2.4 Intra-Protocol Fairness

We already demonstrate that the trinomial protocol is intra-protocol fairness convergent in Section 4.1. Here we give a simple proof. In this thesis, we use the fairness index proposed in [63] as

$$f(x) = (\sum_{i=1}^{n} x_i)^2 / (n \sum_{i=1}^{n} x_i^2), \tag{4.19}$$

where a system allocates resources to n contending users and x_i is the resource allocated to the i^{th} user. $f(x)$ is the fairness index which belongs to [$1/n$, 1]. If all users get the same amount of resource allocation, *i.e.* x_i*'s* are all equal, then the fairness index $f(x)$ is one that indicates 100% fair. As the disparity increases, fairness decreases. A scheme favouring only one selected user will lead to a fairness index equal to $1/n$.

For simplicity, we consider the case where only two trinomial users share the resources and the two users adopt the same trinomial algorithm. It is easy to extend the results to multi-user cases. For a 2-user application $\{x_1, x_2\}$, the fairness index can be expressed as

$$f(x_1, x_2) = \frac{(x_1 + x_2)^2}{2(x_1^2 + x_2^2)}$$

To show that the trinomial protocol is intra-fairness convergent, we only need to show that $f(x_1, x_2)$ is monotonically increasing or $f(x_1, x_2) \to 1$ as $t \to \infty$.

Proof: we assume that feedbacks are synchronous [58]. In each epoch, the decrease rule does not affect fairness as we show in Figure 4.2. We only need to consider the increase rule. For each epoch, let $x_1(t) = S_{01} + (\frac{t}{a})^r$ and $x_1(t) = S_{02} + (\frac{t}{a})^r$, we get

$$f^{'}(x_1, x_2) = \frac{\frac{r}{\alpha^r} t^{r-1} \left(S_{01} + S_{02} + 2(\frac{t}{\alpha})^r \right) (S_{01} - S_{02})^2}{\left(S_{01}^2 + S_{02}^2 + 2(S_{01} + S_{02})(\frac{t}{a})^r + 2(\frac{t}{\alpha})^{2r} \right)^2} \tag{4.20}$$

If $S_{01} = S_{02}$, $f^{'}(x_1, x_2) = 0$ and x_1 is always equal to x_2, which means $f(x_1, x_2) = 1$ (100% fair). Otherwise, $f'(x_1, x_2) > 0$ which means that $f(x_1, x_2)$ is monotonically increasing during the increase phase of the epoch.

Since the decrease rule does not affect fairness, $f(x_1,x_2)$ increases monotonically in each epoch. Thus, we conclude that the trinomial protocol is *intra-protocol fairness convergent.*

4.2.5 Efficiency

The efficiency of a data transport protocol is defined as the closeness of the total load on the bandwidth to the maximum capability of the Internet. Efficiency relates only to the total allocation and thus two different allocations can both be efficient as long as the total allocation is close to the maximum capability of the Internet.

To be efficiency convergent, the protocol should increase total load when there is extra bandwidth available, and decrease total load when the Internet is overloaded [27].

Like in previous sections, for simplicity, we consider the case where only 2 trinomial users share the resources and the two users adopt the same trinomial algorithm. It is easy to extend the results to multi-user applications.

Let B_{ss} is the optimal state, from the increase rule, we can get the upper boundary (B_U) of the bandwidth taken by the two users as:

$$B_U = B_{ss} + \frac{2}{\alpha^r}\left((T_{ss}+1)^r - T_{ss}^r\right), \text{ where, } T_{ss} = \alpha\left(\frac{B_{ss} - S_{01} - S_{02}}{2}\right)^{(1/r)},$$

From the decrease rule, we have the lower boundary (B_L) of the bandwidth that the two users use as:

$$B_L = (1-\beta)B_U .$$

Due to the binary nature of feedback, the trinomial protocol does not stay on B_{ss}, the optimal state. Instead, the system first converges to B_{ss} and then oscillates around B_{ss} within the boundaries [B_L, B_U]. The magnitude of fluctuation is mainly determined by the values of protocol parameters: α, β and γ.

4.2.6 Visualization of Trinomial Protocol Dynamics

To use the techniques described in [27] again, figure 4.6 shows a complete trajectory of the two-user system starting from point P_0 using the same trinomial protocol (α, β and γ are the same). Without loss of generality, we choose $x_1 < x_2$ at the starting position (P_0). Because P_0 is below the *efficiency line* (B_{ss}), there is extra bandwidth available. Both users take *extra bandwidth* at the same speed because α and γ are the same. The point (x_1, x_2) moves upright at an angle of 45° until gets to position P_1, which is above the *efficiency line*. The system is overloaded and congestion occurs. The two users are asked to reduce their transmission rates and they do so according to (4.6). This action corresponds to moving towards the origin on the line joining P_1 and it brings the point (x_1, x_2) to position P_2, which is below the *efficiency line*.

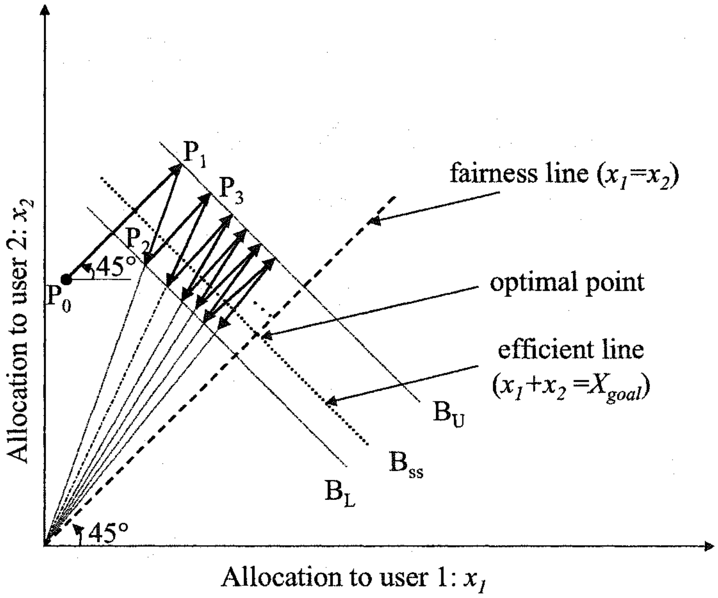

P1
P3
P2
45°
P0
fairness line (x_1=x_2)
optimal point
efficient line
(x_1+x_2 =X_{goal})
BU
Bss
BL
45°
Allocation to user 2: x_2
Allocation to user 1: x_1

Figure 4.6: The dynamics of the trinomial protocol for a two-user case

It is evident that P_2 has higher fairness than P_0 since P_2 is closer to the *fairness line* than P_0. Consequently, the fairness index increases slightly with each evolving epoch and eventually the bandwidth allocation point (x_1, x_2) converges to the *fairness line* where $x_1 = x_2$.

Once the system converges to the *efficiency line* by getting to P_1 from whatever initial state, for every evolving cycle thereafter, the system will oscillate around the *efficiency line* within the range, $[B_L, B_U]$.

Combining above two tendencies, the system will eventually converge to the optimal point in the sense that it keeps oscillating around that point. The trajectory is similar to the AIMD algorithm [27] in a 2-dimenitional space.

It should be noticed that the performance of efficiency convergence is largely determined by the constraints on responsiveness and TCP-compatibility.

4.3 Summary

In this chapter, we first tentatively consider a spectrum of linear transport protocols. The dynamics of the protocols is examined and the system state transitions are visualized as a trajectory through a 2-dimenional vector space. From the visualization, we derive the constraints on the protocols to be convergent to fairness and efficiency.

Based on the constraints on new transport protocols and requirements of the data transmission of teleoperation and e-service robotic systems, we introduce a novel end-to-end rate-based transport protocol. Since there are three parameters (α, β and γ) that are adjustable and designable in the protocol, we call it the trinomial protocol. By a proper combination of α, β and γ, the transient response of the trinomial protocol to network bandwidth variation is quick while the transmission rate is smooth in the steady state.

The trinomial protocol is responsive since it slows down its transmission rate when the network is congested. The constraint on the parameters, α, β and γ, for the protocol to be TCP-compatible (inter-protocol fair) is derived based on both the deterministic and stochastic model. The trinomial protocol is intra-protocol fairness convergent and efficiency convergent, which are shown theoretically and illustrated visually.

Chapter 5

Internet Delay Analysis and Estimation

Among the several aspects of Internet path dynamics, delay dynamics plays a key role in determining transmission quality for teleoperation and e-service robotic systems. Each packet generated by a source is routed to the destination via a sequence of intermediate nodes. The end-to-end delay is thus the sum of the delays experienced at each hop on the way to the destination. Generally, the delay of an Internet path consists of three components: propagation delay, computing delay and queuing delay. Propagation delay is at the speed of light, computing delay is introduced by packet routing and data transcoding, and queuing delay is packet-waiting time in the shared queue buffer before being served. The first two components are usually constant or vary in a very small range for packets with fixed size. However, queuing delay can change in a dramatically large range depending on the queue buffer size, and the network traffic intensity and characteristics. Consequently, queuing delay essentially determines the dynamics of Internet end-to-end delays.

Several reasons cause queuing delay inevitable in the Internet:

- Network traffic is irregular in nature and buffers are necessary to accommodate the irregularity.
- Mismatch among the link speeds of router's interfaces requires the slow links reserving certain buffers to store the traffic impulse fed by the fast links.
- Traffic flows from multiple links merge together to feed one link, causing further mismatch of link speeds.

In this chapter, we first study Internet roundtrip delay (or called roundtrip time) (RTT) statistically. Especially we examine whether RTT time series possesses any structures. In other words, we try to answer this question: whether there are any correlations in the RTT data, or the RTT time series is completely random. Then we present a novel algorithm for RTT and roundtrip timeout (RTO) estimation, which are an integral component of the implementation of the trinomial protocol.

5.1 Statistical Analysis of RTT

Although some work was done to model Internet RTT in the early 1990s, because of rapid changes in the number of users and connection types, the network model and parameters available in the literature may not be applicable to today's Internet. So far, little work has been done to explicitly investigate RTT from the point of view of RTT estimation. Moreover, previous research is focused on low-rate data exchange, whereas teleoperation and e-service robotic systems are concerned with fast-rate connections. For these reasons, it is still necessary to visit this topic. In this section, we employ a statistical approach to characterize RTT.

5.1.1 Related Work

A few studies on packet delays in various network environments are reported in the literature. These studies can be categorized into analytic [125,96], simulation [99-101] and experimental approaches [26,102-104,126].

Since the dynamics of path delays is dominated by queuing delay, conventional analytic approaches to characterizing delays concentrate on queuing theory or further delay distributions under certain strict assumptions about the incoming traffic models, for example, Markovian (exponential distribution), r-stage Erlangian [125] and Markov modulated Poisson Process (MMPP) [96]. However, queuing models cannot incorporate the features of real-world networks, such as the correlations introduced when traffic streams merge and split, the regulation of traffic by routing and congestion control mechanisms or the packet losses due to buffer overflow. Empirical studies [97, 98] demonstrate the shortcomings of these models for the modern Internet. Due to the complexities of the real Internet, most studies of packet delays are conducted with simulation and experimental approaches.

Regarding simulation approaches, it is concluded that both the link state and distance vector routing yield similar statistics of average packet delays in an NSFNET-like network [99]. The dynamic behavior of the delay of TCP connections is also investigated in [100,101].

Many researches have taken the experimental approaches. Systematic measurements of packet delays and losses were carried out on the ARPANET as early as 1971 [126]. People examine the variations of packet delays for different paths, different times of a day and different days of a week, etc. Other measurements are conducted to determine how delays across the ARPANET are influenced by packet length. The results are used to assess whether TCP performance could be improved by including a dependence on packet length in the retransmission timeout algorithm [102]. Several other studies address timeout adjustment in TCP and they propose improvements to take into account packet losses, packet retransmissions and the variances of RTT's [26, 103]. The NSFNET replaced the ARPANET in 1990. Studies have measured the delay and loss behaviors in the NSFNET and more generally in the Internet.

All above studies can be summarized into two categories of purposes: 1) understanding network performance; and 2) understanding Internet workload. For both of them, packet delays are obviously an effective indicator of the status of a network.

Our objective in this thesis is to investigate whether there are some structures, and what structures if any, in RTT time series. In [108], it is shown that the characteristics of a measured RTT sequence are determined by many factors, such as the scale of sampling periods, network types, application types and network sizes etc. Thus, it is difficult to find a uniform analytical modeling framework, which can accommodate all the cases [104, 108].

In this section, rather than pursuing a general analytic RTT model, we use both linear and nonlinear statistical analysis methods to investigate whether certain patterns, and what patterns if any, exist in RTT time series when the sampling interval is in tens to hundreds milliseconds, around which range the teleoperation and e-service robotic systems mostly may fall.

5.1.2 Data Collection

The ICMP (Internet Control Message Protocol) based *Pinger* is chosen as the tool to collect RTT data. In the experiments, unfragmented ICMP packets are sent at a fixed interval from the *Pinger* host, art333g.ee.ualberta.ca, at the University of Alberta, Canada, to the *Pingee* host, www.sohoo.com. One probing packet is sent out every 200 milliseconds. Of the 107,700 packets sent out, we get 106,790 RTT samples and 1910 timeouts that are removed from the RTT time series. The RTT time series sample is shown in Figure 5.1.

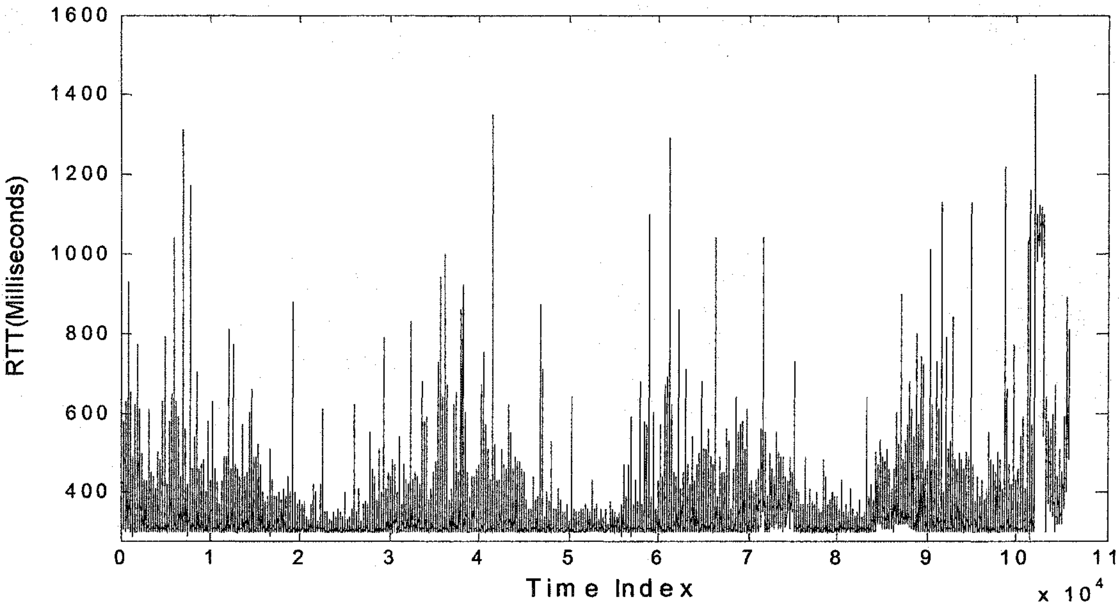


Figure 5.1: RTT time series sample

5.1.3 Histogram

The histogram of the RTT time series is shown in Figure 5.2.

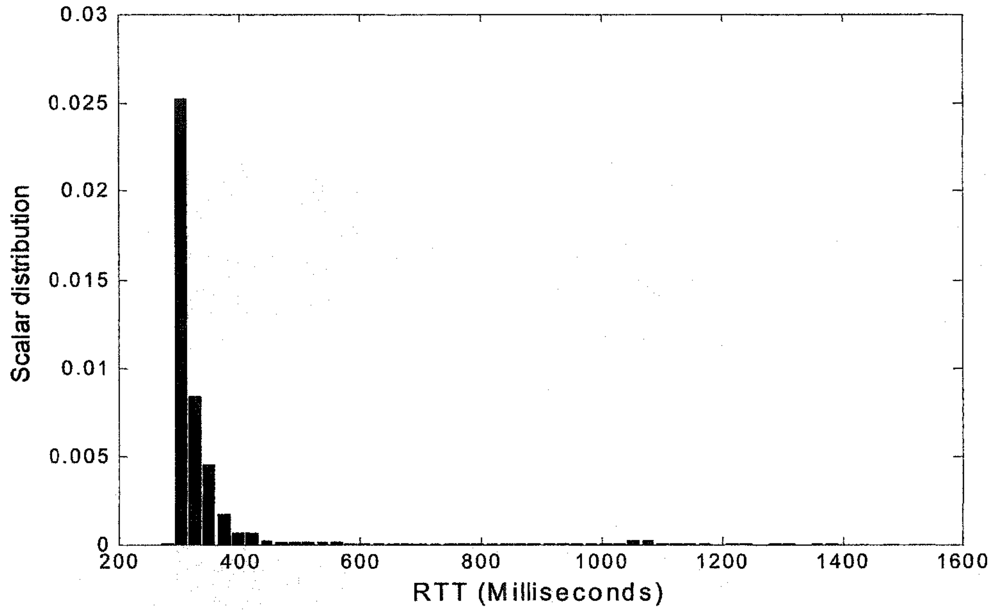


Figure 5.2: Histogram of RTT data set

This is a "skewed right" distribution for which the tail is on the right hand side. The RTT's are distributed within the scope of 281ms to 1452ms and the average is 334.31ms. The peak value is between 304ms and 328ms. There is a lower boundary for the RTT's, which is largely determined by the propagation and computing delay. The distribution of the RTT data is best fitted by a constant plus a gamma distribution.

The unsymmetrical "skewed right" characteristic indicates that the RTT time series is not a Gaussian or uniform white noise. The tail actually reflects the congestion of varying intensities in router buffer [173].

5.1.4 Autocorrelation

Autocorrelation is a commonly used tool for checking randomness in a data set. Randomness is ascertained by computing autocorrelations for data values at varying time lags. If random, such autocorrelations should be near zero for any and all time-lag separations. On the other hand, if non-random, one or more of the autocorrelations should be significantly non-zero [107].

Given a time series, $x_1, x_2, \cdots, x_N$ at time lag $1, 2, \cdots, N$, the lag τ autocorrelation function is defined as

$$C(\tau) = \frac{\sum_{i=1}^{N-\tau}(x_i - \bar{x})(x_{i+\tau} - \bar{x})}{\sum_{i=1}^{N}(x_i - \bar{x})^2}, \tag{5.1}$$

where $\bar{x}$ is the average of $x_1, x_2, \cdots, x_N$.

From the autocorrelation plot given by Figure 5.3, we can see that it starts with an autocorrelation of about 0.925 at lag 1 and then decreases gradually. Such a pattern shows that the RTT time series is not random, but rather *linearly correlated* between adjacent and near-adjacent observations [139].

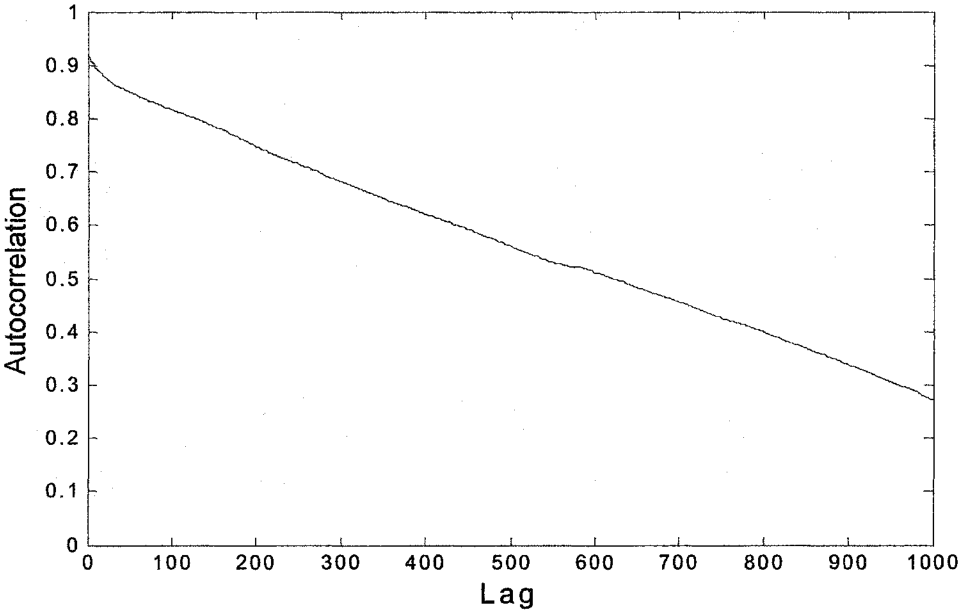


Figure 5.3: Autocorrelation plot

5.1.5 Power Spectrum

Power spectrum is a smoothed Fourier transform of the autocovariance function and a graphical technique for examining cyclic structures in the frequency domain [105,106]. The frequency is measured in cycles per unit time where the unit time is defined as the distance between 2 points. A frequency of 0 corresponds to an infinite cycle while a frequency of 0.5 corresponds to a cycle of 2 data points. Its curve shows how the variance of a stochastic process is distributed with frequency. The Logarithm power spectrum plot of the RTT time series is shown in Figure 5.4.

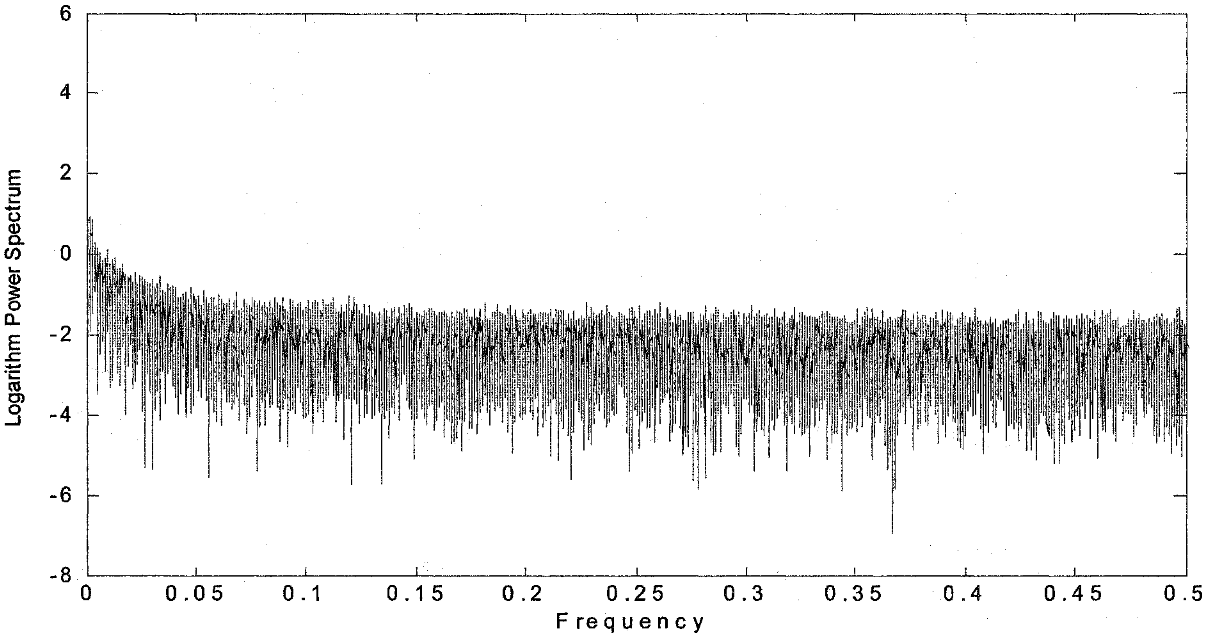


Figure 5.4: Power spectrum plot of RTT time series

The spectrum plot starts with a dominant peak (≈5.2) near frequency zero and rapidly decays to under zero for other higher frequencies. This is the spectral plot signature of a process with positive autocorrelation [137, 138].

5.1.6 Mutual Information

Mutual information is also a measure of the statistical independence between measurement x_i and time-lagged measurement $x_{i+\tau}$ in a time series [140]. However, unlike the autocorrelation function and power spectrum in which only linear dependence is incorporated, the mutual information takes into account both linear and nonlinear correlations. The average mutual information of a time series is defined as:

$$I(\tau) = \sum_{i,j} p_{ij}(\tau) \ln \frac{p_{ij}(\tau)}{p_i p_j}, \tag{5.2}$$

where τ is time lag; p_i is the probability to find a time series value in the i-th interval; and $p_{ij}(\tau)$ is the joint probability that an observation falls into the i-th interval and the observation τ later falls into the j-th. If the correlation in a time series is nonlinear, the mutual information should be *significantly* greater than zero in small lags [172]. The average mutual information plot is given by Figure 5.5.

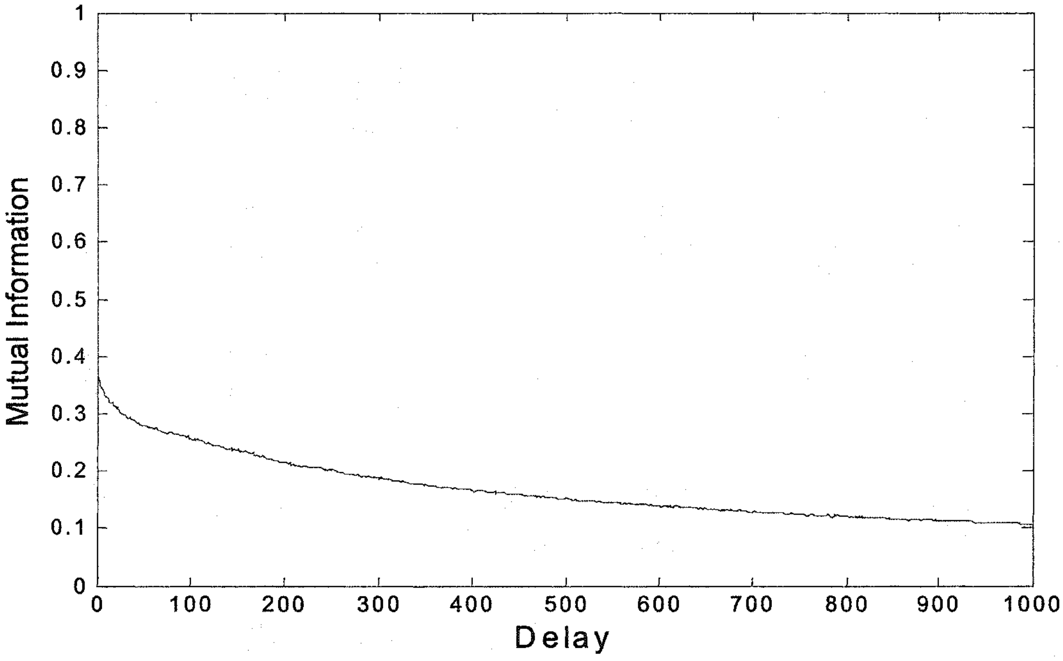


Figure 5.5: Mutual information plot

From Figure 5.5, it can be inferred that there are *no* or only *weak* *nonlinear* correlation among observations because $I(\tau)$ is less than 0.35 for lag τ larger than 0. Otherwise we would expect much higher values of mutual information for τ lager than and close to 0.

5.1.7 Conclusions on RTT Analysis

From the histogram plot, there is a lower boundary for the RTT time series, which explains partially why continuous command control in teleoperation and e-service robotic systems is difficult for tasks with small time constants. The "skewed right" characteristic means a nonsymmetrical distribution, and so, the RTT data are not Gaussian or uniform white noises.

The power spectrum and autocorrelation plots indicate that the RTT time series is *linearly correlated* between adjacent and near-adjacent observations. Thus, the RTT sequence is a process with some structures rather than a random one.

From the mutual information plot, there are no high values of mutual information for time lag τ lager than and close to 0. Intuitively, it can be inferred that there are weak or no nonlinear correlations among observations.

Based on theses findings, we reach the conclusion that the RTT time series is a process with certain structures rather than a random process. To estimate the value of next RTT based on past and current observations, the RTT time series should be characterized by linear models. Next, we discuss RTT and RTO estimations.

5.2 RTT and RTO Estimation

5.2.1 Reasons for RTT and RTO Estimation

RTT and RTO estimations are an integral component of the implementation of the trinomial protocol.

The trinomial protocol is rate-based, which is determined by the nature of teleoperation and e-service robotic systems. Like other rate-based transport protocols, the transmission rate of the trinomial protocol is controlled by adjusting the value of the *inter-packet-gap* (IPG) in the source. This relies on an estimation (or prediction) of next RTT to determine the value of IPG, the gap between two succeeding packets. For example, if the estimated value of next RTT is 1 second and the transmission rate is 5 packets per RTT, then the value of IPG is 0.2 seconds. Consequently, smaller IPG means higher transmission rate while larger IPG indicates lower transmission rate. In other words, too large RTT estimate causes a low transmission rate and thus a conservative protocol. On the other hand, too small RTT estimate leads to a high transmission rate and hence an aggressive protocol.

As mentioned in previous chapters, today's Internet signals congestion implicitly. Transport protocols detect network congestion from indirect congestion signals, such as packet losses or delay variances. Like TCP and most other transport protocols, the trinomial protocol uses packet losses as the only indication of network congestion. This mechanism partially depends on a proper timer (estimated RTO) to determine whether packets are lost. If the RTO estimate is too small, false congestion is alarmed, which causes unnecessary back-off in transmission rate. On the other hand, if the RTO estimate is too large, bandwidth is wasted and the time delay is elongated unnecessarily.

The RTT and RTO estimations are also believed useful for the design of teleoperation control algorithms. It would make it much easier to design a teleoperation control scheme should the roundtrip time and its boundary were known in advance.

5.2.2 Current Approaches

Currently, the widely adopted method for RTT estimation is based an autoregressive, moving average (ARMA) model with *fixed* parameters. The ARMA model roots from the timer estimation in TCP implementation. Another ARMA model is used to estimate the mean deviation (RTTVAR) between the SRTT and the measured RTT. The roundtrip timeout (RTO) (or called delay boundary) is then calculated as the sum of SRTT and RTTVAR multiplied by a constant factor, k =4 [19,26]. The low-pass filter used to estimate SRTT is shown in (5.3)

$$\hat{R}_{n+1} = (1-g)\hat{R}_n + gR_n, \tag{5.3}$$

where $\hat{R}_{n+1}$ is the estimated delay at time n+1 (SRTT), R_n is the delay measurement at time n, g is a constant in the range of [0.1, 0.25]. By doing some mathematical manipulations, the ARMA model shown in (5.3) is found based on the assumption that the RTT time series is modelled by (5.4) [108].

$$R_n = M + e_n + g\sum_{k=1}^{\infty} e_{n-k}, \tag{5.4}$$

where M is a constant representing the mean RTT and $\{e_i,\ i= -\infty, \ldots, n, n+1\}$ is a sequence of independent, identically distributed (iid) random variables with zero mean and a certain variance.

From (5.4), we can see that when $g \to 0$, we actually model the delays as a white noise; when $g \to 1$, we model the delays as a sequence of Brownian motion, a non-stationary process; and when $0 < g < 1$, we model the delays as the sum of a white noise and a weighted non-stationary process [108].

While the algorithm shown in (5.3) is widely used, further studies reveal that this parameter-fixed ARMA-based SRTT predictor has some limitations. With such a small and fixed g, e.g. 0.1~ 0.2, in most current TCP implementations, this method presumes implicitly that the dynamics of RTT is dominated by a slow stationary process and independent of concurrent load, routing and path etc. From recent studies [108,109] and our experiments, it is shown that this is not always true. Actually, RTT dynamics is quite complicated. It evolves with time. For a sequence of measured RTT's, it might be dominated by a base plus a noise process in some intervals, but with strong structures in others. It may be as a stationary process in some intervals while a nonstationary one or a complex combination of both in other intervals. This *multi-structured nonstationary* characteristic is *not* well fitted by a model with *fixed* parameters. For example, if one window of a RTT time series is dominated by a base plus a noise process, we should choose a much small g for (5.3) because the mean is the best estimate of next RTT at this case. On the opposite, we may need a large g for packets clustered together (a burst of packets or the network is congested) since the delays during this period may be characterized by a non-stationary Brownian motion.

One main reason that current ARMA model works very well in TCP implementations is that the inaccuracy of RTT prediction can be absorbed by the RTO estimation. TCP implementation does not utilize RTT estimates directly. It only uses estimated RTT to calculate RTO.

However, for the trinomial protocol, it is very important to have an accurate estimate of RTT because it directly affects protocol performance as we discuss at the beginning of this section. The estimated RTT determines the magnitude of IPG (transmission rate) directly.

5.3 RTT Estimation Using the MEP Algorithm

The shortcomings of current *parameter-fixed* ARMA model indicate that the RTT estimation algorithm needs to be further investigated. As we discuss in the last section, due to the complexities of the Internet, it is not likely to derive an explicit analytic model for RTT. RTT evolves under the direct and indirect influences of innumerous factors.

In this chapter, we employ a *parameter-varying* adaptive algorithm for RTT estimation based on the information theory and maximum entropy principle (MEP) which is first introduced by Klan and Darbellay [110]. Next, we introduce how this algorithm works.

5.3.1 Proposed Model

The RTT estimation problem can be expressed as: given the knowledge of n lately RTT measurements, $\{R_i, i=n, n-1, \ldots, n-k+1\}$, what is the most likely value of next RTT (at instant $n+1$), $\hat{R}_{n+1}$. This is a univariate short-term (one step ahead) forecasting problem. The predicted RTT, $\hat{R}_{n+1}$, can be regarded as a function of past n observations,

$$\hat{R}_{n+1} = f(R_n, R_{n-1}, \cdots, R_{n-k+1}), \tag{5.5}$$

where f is unknown and k is the order, or 'memory', of the model. The forecasting problem is actually to choose an optimal f based on some criteria so that the estimated $\hat{R}_{n+1}$ is as close as possible to R_{n+1}, which has not occurred.

Let us first review the *global* autoregressive linear model. The estimation for the model is

$$\hat{R}_{n+1} = a_1 R_n + a_2 R_{n-1} + \cdots + a_k R_{n-k+1}, \tag{5.6}$$

where $a_1, a_2, \cdots, a_k$ are constants. Once an estimation model is identified, the prediction can start. The coefficients $a_1, a_2, \cdots, a_k$ or f in (5.5) are fixed, thus we call it the global autoregressive linear model. From what we discuss in Section 5.2.2, the global (parameter-fixed) model does not work for RTT estimation.

We now generalize above model by introducing *time-varying* coefficients. The proposed tentative model is defined as a convex combination (CC) of past data as shown in (5.7),

$$\hat{R}_{n+1} = a_{n1} R_n + a_{n2} R_{n-1} + \cdots + a_{nk} R_{n-k+1}, \tag{5.7}$$

where, $a_{ni}, i = 1, 2, \cdots, k$, are the coefficients of the model at instant n. The lower index n expresses the *time-dependence* of the coefficients. Thus (5.7) can be viewed as a *local autoregressive linear model*. These coefficients satisfy the condition shown in (5.8)

$$\sum_{i=1}^{k} a_{ni} = 1, \quad a_{ni} \geq 0, \tag{5.8}$$

The inappropriate of the model shown in (5.7) is evident. From (5.8) we know that $\hat{R}_{n+1}$ can lie only in the interval [a_n, b_n], where $a_n = \min\{R_n, R_{n-1}, \cdots, R_{n-k+1}\}$ and $b_n = \max\{R_n, R_{n-1}, \cdots, R_{n-k+1}\}$. Hence, the interval [$a_n$, b_n] must be extended to be practicable. For this purpose, two more terms, R_n^L and R_n^H, such that $R_n^L \le a_n$ and $R_n^H \ge b_n$, are added into (5.7). The model then becomes

$$\hat{R}_{n+1} = a_{n1}R_n + a_{n2}R_{n-1} + \cdots + a_{nk}R_{n-k+1} + a_{n(k+1)}R_n^L + a_{n(k+2)}R_n^H, \tag{5.9}$$

The condition for the coefficients now becomes

$$\sum_{i=1}^{k+2} a_{ni} = 1, \quad a_{ni} \ge 0, \tag{5.10}$$

The only assumption is that the coefficients of the model shown in (5.9) still possess validity when they are used to calculate current RTT, R_n, with the real time series {*R_i*, *i*=*n*-1, *n*-2, ..., *n*-*k*}. The reflex condition presented in (5.11) is used for this purpose.

$$R_n = a_{n1}R_{n-1} + a_{n2}R_{n-2} + \cdots + a_{nk}R_{n-k} + a_{n(k+1)}R_n^L + a_{n(k+2)}R_n^H, \tag{5.11}$$

The reflex condition is used to calculate the coefficient $a_{ni}, i = 1, 2, \cdots, k+2$. Next, we discuss how to calculate the coefficients.

5.3.2 The MEP Algorithm

The key point of the model shown in (5.9) is that all its coefficients, $a_{ni}, i = 1, 2, \cdots, k+2$, are positive numbers and the sum of them is 1, which is shown in (5.10). If this is the case, $a_{ni}, i = 1, 2, \cdots, k+2$ can be regarded as a set of probability distributions over $R = \{R_{n-1}, R_{n-2}, \cdots, R_{n\text{-}k}, R_n^L, R_n^H\}$. Thus, the reflex condition (5.11) can be understood as the constraint on the set of probability distributions, $a_{ni}, i = 1, 2, \cdots, k+2$. The reflex condition is the only relevant information available on next RTT (R_{n+1}) at time instant *n*, and there are infinite distributions satisfying (5.11). According to the information theory, the maximum entropy principle (MEP) is the most unbiased prescription to choose $a_{ni}, i = 1, 2, \cdots, k+2$, for which the Shannon entropy, *i.e.*, the expression

$$H(a_n) = -\sum_{i=1}^{k+2} a_{ni} \ln a_{ni}, \tag{5.12}$$

is maximal under the reflex condition specified in (5.11). Using the Lagrange multipliers, the solution to (5.12) is [112]:

$$a_{ni} = \frac{e^{-\beta_n R_{n-i}}}{\sum_{j=1}^{k+2} e^{-\beta_n R_{n-j}}}, \quad i = 1,2,\cdots k+2, \tag{5.13}$$

where β_n is the solution to the equation (5.14)

$$\sum_{j=1}^{k+2} (R_{n-j} - R_n) e^{-\beta_n (R_{n-j} - R_n)} = 0, \tag{5.14}$$

In (5.13) and (5.14), $R_{n-(n+1)} = R_n^L$ and $R_{n-(n+2)} = R_n^H$.

The maximum entropy probability distribution represents the least biased judgment possible for $a_{ni}, i = 1, 2, \cdots, k+2$. This claim is justified in [113,114] and thereafter. For example, the MEP contains the classic principle of insufficient reason (PIR) as a special case. Indeed, in the absence of any reason, *i.e.*, in the case that no or only trivial constraints are imposed on the probability distribution, $H(a_n)$ is maximal when all probabilities are equal, which corresponds to a uniform distribution.

(5.14) is an exponential equation. Its solution has to be obtained numerically. It is however enlightening to consider an approximate analytical solution to the calculation of the coefficients a_{ni}. We can write

$$a_{ni} = \frac{1}{k+2} + \varsigma_{ni}, \tag{5.15}$$

where ς_{ni}, $i = 1, 2, \cdots, k+2$, are real numbers and $\sum \varsigma_{ni} = 0$. This means that the coefficients fluctuate around a constant value, 1/(*k*+2). The entropy of the associated distribution at instant *n* becomes:

$$H(a_n) = -\sum_{i=1}^{k+2} (g + \varsigma_{ni}) \ln(g + \varsigma_{ni}),$$

where $g = 1/(k+2)$. Below, to simplify the notations, we drop the index *n*. The logarithm can be written as

$$\ln(g+\varsigma_i)=\ln((g)(1+\frac{\varsigma_i}{g}))=\ln(g)+\ln(1+\frac{\varsigma_i}{g})$$

If the set of coefficients is close to the uniform distribution, we have $|\varsigma_i/g|<1$. Then we can approximate the last term as

$$\ln(1+\frac{\varsigma_i}{g})\approx\frac{\varsigma_i}{g}-\frac{\varsigma_i^2}{2g^2}$$

The entropy becomes

$$H(a)\approx-\sum_{i=1}^{k+2}(g+\varsigma_i)[\ln(g)+\frac{\varsigma_i}{g}-\frac{\varsigma_i^2}{2g^2})]$$

$$\approx-\sum_{i=1}^{k+2}g\ln(g)-\sum_{i=1}^{k+2}\varsigma_i+\sum_{i=1}^{k+2}\frac{\varsigma_i^2}{2g}-\sum_{i=1}^{k+2}\varsigma_i\ln(g)-\sum_{i=1}^{k+2}\frac{\varsigma_i^2}{g}$$

to the second order in ς_i. Since $\sum_i^{k+2}\varsigma_i=0$ and g is a constant, we obtain

$$H(a)=-\sum_{i=1}^{k+2}g\ln(g)-\sum_{i=1}^{k+2}\frac{\varsigma_i^2}{2g}.$$

The maximum entropy principle requires that we find

$$\max_a\{-\sum_{i=1}^{k+2}(\frac{1}{k+2}\ln(\frac{1}{k+2}))-\frac{k+2}{2}\sum_{i=1}^{k+2}\varsigma_i^2\}$$

However, since the first sum is a constant, this is equivalent to minimising

$$\sum_{i=1}^{k+2}(a_i-\frac{1}{k+2})^2$$

over the set a. This means that, under the approximation up to the second order in ς_i, the maximum entropy principle boils down to the least square minimisation.

To take the constraints into account, we introduce the Lagrange function,

$$L=\sum_{i=1}^{k+2}(a_i-\frac{1}{k+2})^2-\alpha(\sum_{i=1}^{k+2}a_i-1)-\beta(\sum_{i=1}^{k+2}(a_iR_{n-i}-R_n)),$$

where α and β are the Lagrange multipliers corresponding to the constraints (5.10) and (5.11). Putting the first order partial derivatives equal to zero we get

$$\frac{\partial L}{\partial a_i} = 2a_i - \frac{2}{k+2} - \alpha - \beta R_{n-i} = 0, \quad i = 1, 2, \cdots, n+2,$$

$$\frac{\partial L}{\partial \alpha} = 1 - \sum_{i=1}^{k+2} a_i = 0,$$

$$\frac{\partial L}{\partial \beta} = R_n - \sum_{i=1}^{k+2} a_i R_{n-i} = 0.$$

Reintroducing the index *n* for a_{ni}, the final solution is

$$a_{ni} = \frac{1}{k+2} - \frac{1}{k+2} \frac{R_n - \frac{1}{k+2}\sum_i R_{n-i}}{\sum_i R_{n-i}^2 - \frac{1}{k+2}(\sum_i R_{n-i})^2} \sum_i R_{n-i} + \frac{R_n - \frac{1}{k+2}\sum_i R_{n-i}}{\sum_i R_{n-i}^2 - \frac{1}{k+2}(\sum_i R_{n-i})^2} R_{n-i}, \tag{5.16}$$

where, $i = 1, 2, \cdots, k+2$. Thus, the coefficients of the system model, a_{ni}, can be easily calculated from the inputs, current *i*+2 observations, $R = \{R_{n-1}, R_{n-2}, \cdots, R_{n-k}, R_n^L, R_n^H\}$.

Since (5.16) is only a polynomial equation, the computing overhead of updating the coefficients is quite small compared to other adaptive algorithms, such as the Deviation-Lag Function (DLF) scheme proposed in [108] for which computation is too intensive to be practical.

Because the coefficients of (5.9) are updated every RTT measurement, the model parameters are *time-varying* and the model is adaptive. Thus, it can track RTT dynamics quickly.

5.3.3 RTO Estimation

We employ the same method as the one in TCP implementation to estimate the roundtrip timeout (RTO) based on the RTT estimate. This scheme is shown as follows:

$$RTO_{n+1} = \hat{R}_{n+1} + \beta \times \hat{\sigma}_{n+1}, \tag{5.17}$$

where $\hat{\sigma}_i$ stands for the estimate of RTT variation for the i^{th} data packet and β is a constant factor which usually takes the value of 4 [26]. The RTT variations are estimated according to (5.18):

$$\hat{\sigma}_{n+1} = (1-\alpha)\hat{\sigma}_n + \alpha \times | R_n - \hat{R}_n |, \tag{5.18}$$

where α is a constant factor and usually takes the value of 0.25 [26].

5.4 Experiments

The MEP algorithm is tested by using the data collected from the real Internet links and compared with the traditional ARMA methods.

For the MEP algorithm used in the experiments, the values of R_n^L and R_n^H in (5.9) are determined by (5.19) and (5.20) respectively:

$$R_n^L = (1-\gamma) \times \min\{R_n, R_{n-1}, \cdots, R_{n-k+1}\}, \tag{5.19}$$

$$R_n^H = (1+\gamma) \times \max\{R_n, R_{n-1}, \cdots, R_{n-k+1}\}, \tag{5.20}$$

where γ=0.2. The order of the model, k that is shown in (5.9), is 4.

5.4.1 Experimental Data Collection

In the experiments, unfragmented ICMP packets are sent out at regular intervals from the *Pinger* host, *meng.ee.ualberta.ca* at the University of Alberta, Canada, to the *Pingee* hosts. Due to the diversities of networks and the unbelievable numbers of hosts on the Internet, it is impossible to study delays for all possible connections. For a fair representative, one *Pingee* host was selected randomly from each combination of the locations and types given by Table 5.1. As a result, we had totally 9 links for the experiments.

Table 5.1: Data collection configuration

Collection Date	Sept. 24, 2001 (Mon)	Sept. 25, 2001 (Tue)	Sept. 24, 2001 (Wed)
Collection Time	10:00am	14:00pm	20:00pm
Pingee Host Location	Northern America	Asian	Europe
Pingee Host Type	Academy	Business	Government

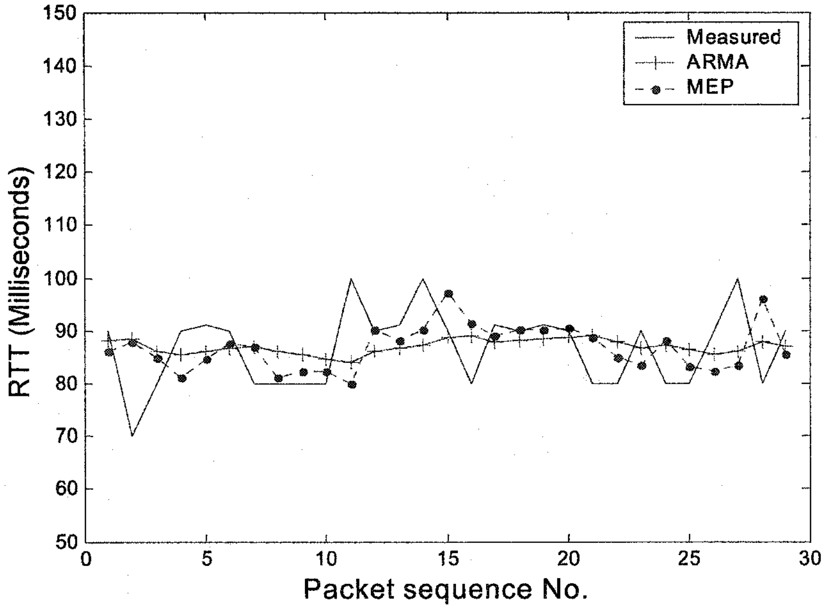
The time scheduled for data collection in Table 5.1 is the Mountain Time (US & Canada). Data was collected for 3 successive working days and we had totally 81 RTT time series samples. In the experiments, the size of probing data packets was 69 bytes and a large RTO (5 seconds) was adopted. In each experiment, 50 successive data packets were sent out from the *Pinger* to the *Pingee* and the *Pinger* received packets of the same size returned from the *Pingee* if the packet was not lost in the link. If the *Pinger* received a returned packet within the time period of RTO, the corresponding RTT was known and counted. Otherwise, the corresponding data position was removed from the RTT time series.

5.4.2 RTT Estimation Results

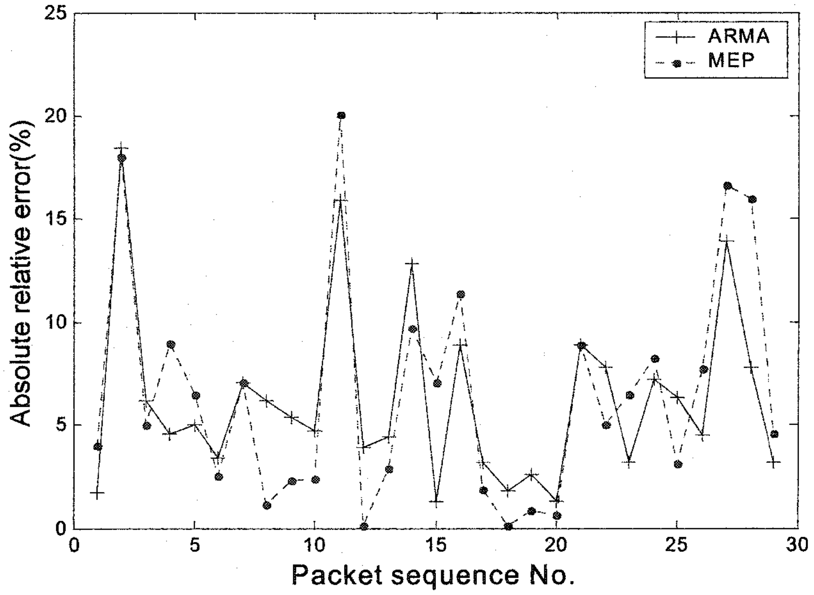
Figure 5.6(a), 5.7(a), 5.8(a) and 5.9(a) show the curves of the measured RTT, estimated RTT by using the ARMA method and estimated RTT by using the MEP algorithm for some typical scenarios from the set of experiment results. Figure 5.6(b), 5.7(b), 5.8(b) and 5.9(b) are the plots of the absolute relative estimation errors respectively.

In the scenario shown in Figure 5.6, the RTT time series is dominated by a base plus a noise process with weak patterns. This is the situation on which the traditional ARMA method is targeted. From the error plot, ARMA and MEP have similar performance on RTT estimation. We cannot determine which scheme is superior.

However, in the scenarios depicted in Figure 5.7, 5.8 and 5.9, the RTT time series show apparent structures. RTT evolves rapidly and significantly. The adjacent observations are strongly correlated. It can be inferred that the network is congested in these scenarios and thus it is particularly important that network transport protocols should perform properly. In other words, the accuracy of RTT estimation is especially relevant. From the estimation plots, it is clear that ARMA cannot track the RTT dynamics because its dynamics is too slow. On the contrary, MEP works very well and much better than ARMA. This point is also reflected by the plots of estimation errors. Although the MEP algorithm may show a peak estimation error higher than that of ARMA in the point where the RTT dynamics changes suddenly, the average errors of the MEP scheme are much smaller than those of ARMA.

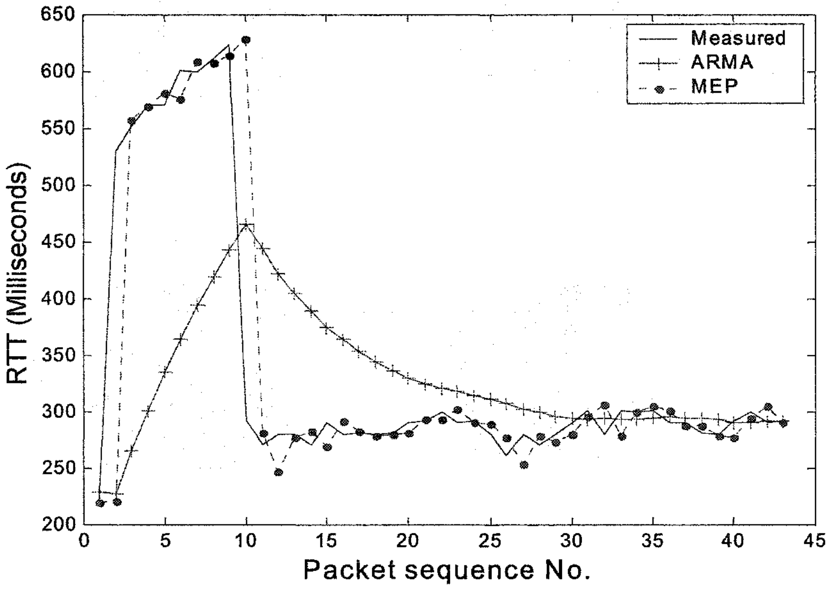


(a) RTT estimation

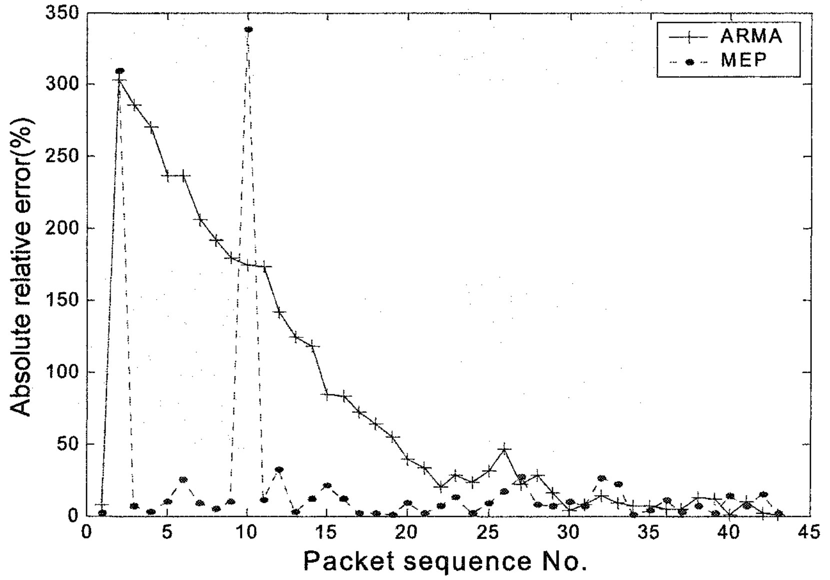


(b) RTT estimation error

Figure 5.6: Comparison of RTT estimation between ARMA and MEP--Scenario I

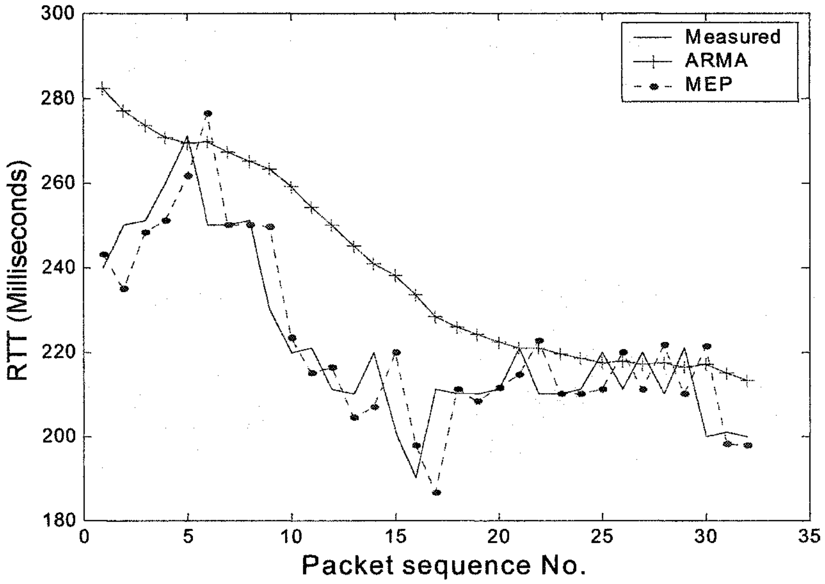


(a) RTT estimation

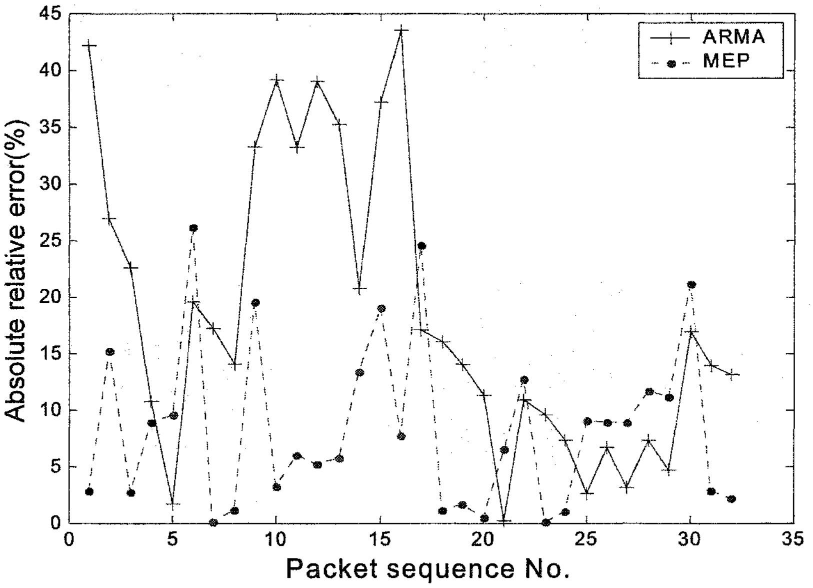


(b) RTT estimation error

Figure 5.7: Comparison of RTT estimation between ARMA and MEP--Scenario II

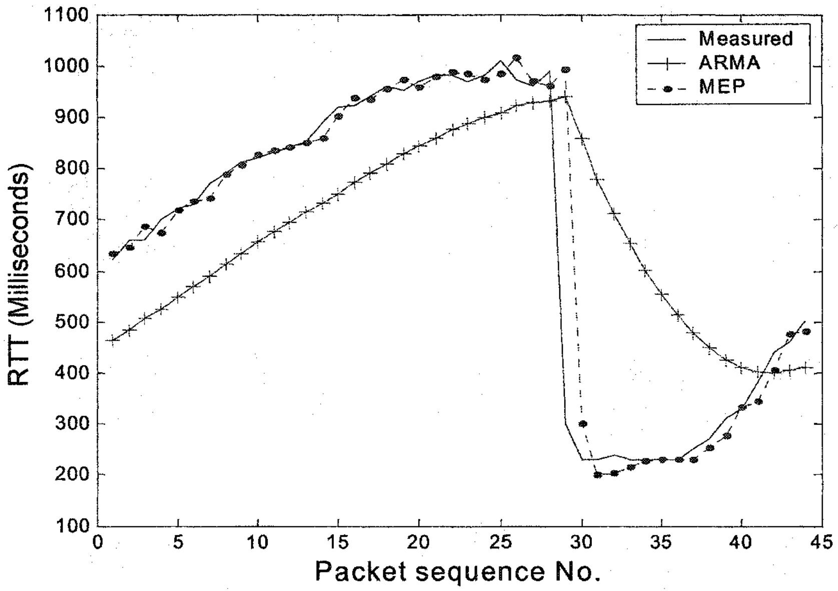


(a) RTT estimation

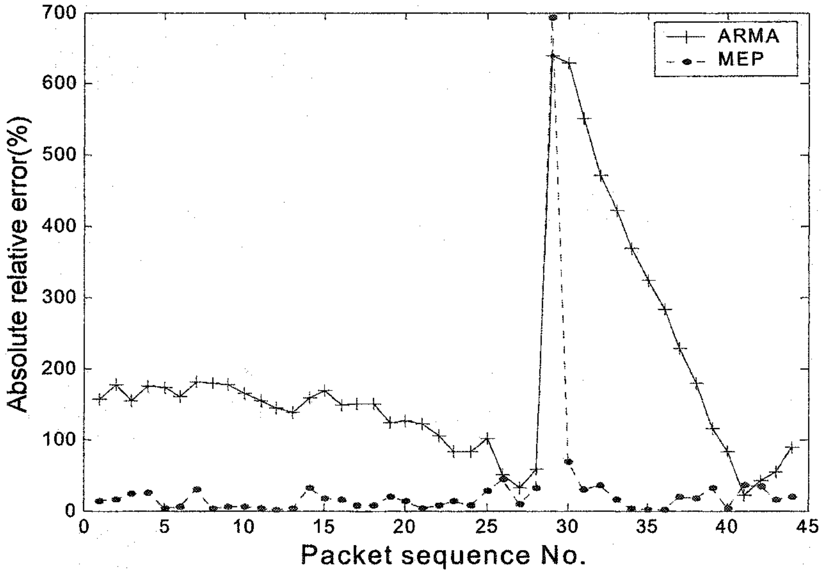


(b) Estimation error

Figure 5.8: Comparison of RTT estimation between ARMA and MEP--Scenario III



(a) RTT estimation



(b) RTT estimation error

Figure 5.9: Comparison of RTT estimation between ARMA and MEP--Scenario IV

5.4.3 RTO Estimation Results

We use two criteria, percentage overhead (*OH*) and percentage error (*ERR*), for the comparison of RTO estimation between ARMA and MEP. Their definitions are as follows:

$$OH = \frac{\sum_i (RTO_i - R_i)}{\sum_i R_i}, \tag{21}$$

$$ERR = \frac{n}{N}, \tag{22}$$

where n is the number of cases in which RTO_i (estimated round trip timeout) is smaller than R_i (measured RTT) and N is the total number of returned data packets the *Pinger* receives.

From above definitions, *OH* can be viewed as the indicator of the overhead of waiting time for the system to determine whether packets get lost. If the *RTO* estimate is too large, we have a large *OH* and thus the bandwidth is wasted by unnecessary waiting. On the other hand, *ERR* is the ratio of the number of packets that are falsely alarmed lost to the total number of returned packets that are received by the *Pinger*. If the *RTO* estimate is too small, a relatively large number of packets will be alarmed lost falsely and thus unnecessary back-off in transmission rate is induced.

The experimental results of the RTO estimation by using both the ARMA and MEP algorithms are summarized in Table 5.2. From the results, it is clear that MEP has small OH's and ERR's compared to ARMA. The detailed results of comparison for the experiments of three successive days are shown in Table 5.3, Table 5.4 and Table 5.5.

Table 5.2: Summary of comparison of RTO estimation between ARMA and MEP

	OH (%)		ERR (%)	
	ARMA	MEP	ARMA	MEP
Monday	19.06	16.38	3.75	1.81
Tuesday	25.51	16.31	1.98	1.31
Wednesday	20.26	19.71	3.73	1.69

Table5.3: Experimental result I of RTO estimation (Date: Monday, Sept. 24, 2001)

Pingee Location	Pingee Host	10:00am				14:00pm				18:00pm			
		OH (%)		ERR (%)		OH (%)		ERR (%)		OH (%)		ERR (%)	
		ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP
Northern America	164.109.71.245	19.94	22.84	0.00	0.00	20.80	31.34	3.57	3.57	18.08	25.90	5.00	2.50
	18.181.0.31	21.09	16.11	0.00	2.00	16.80	15.14	0.00	0.00	13.05	13.48	2.00	2.00
	128.231.56.110	26.20	35.13	2.00	0.00	36.51	37/00	6.00	0.00	14.38	18.35	10.00	0.00
Asia	202.106.68.19	66.14	23.10	4.17	4.17	13.72	14.00	2.38	0.00	41.47	40.19	2.63	0.00
	202.112.144.70	17.64	12.28	2.56	0.00	6.06	8.62	5.40	2.70	6.46	9.70	8.57	2.85
	202.108.249.206	8.60	9.85	0.00	0.00	16.15	14.08	2.77	2.77	35.10	10.75	0.00	2.70
Europe	212.160.117.137	23.65	32.65	5.40	2.7	9.52	14.95	2.86	2.86	11.96	7.33	0.00	0.00
	148.88.2.37	3.79	5.63	12.00	4.00	3.70	5.52	12.00	4.00	4.77	7.68	10.00	6.00
	130.192.239.1	41.96	28.03	2.04	0.00	8.33	9.70	0.00	2.00	8.74	9.99	0.00	2.00
Average		25.45	20.62	3.13	1.43	14.62	12.59	3.89	1.99	17.11	15.93	4.24	2.01

Table5.4: Experimental result II of RTO estimation (Date: Tuesday, Sept. 25, 2001)

Pingee Location	Pingee Host	10:00am				14:00pm				18:00pm			
		OH (%)		ERR (%)		OH (%)		ERR (%)		OH (%)		ERR (%)	
		ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP
Northern America	164.109.71.245	51.89	20.79	0.00	2.08	29.88	34.82	8.70	4.34	22.80	26.16	4.26	2.13
	18.181.0.31	21.20	27.07	0.00	2.00	20.56	20.49	0.00	0.00	19.24	18.30	0.00	0.00
	128.231.56.110	35.01	28.14	4.17	0.00	26.56	28.17	6.00	2.00	18.73	20.73	0.00	2.00
Asia	202.106.68.19	109.19	19.53	2.04	2.04	70.30	36.20	2.04	0.00	6.97	4.58	2.08	2.08
	202.112.144.70	12.20	12.27	3.70	0.00	6.18	5.93	0.00	2.38	7.61	8.67	0.00	0.00
	202.108.249.206	4.92	4.27	0.00	0.00	9.05	8.41	2.00	2.00	8.85	6.44	0.00	2.17
Europe	212.160.117.137	50.73	13.61	2.56	0.00	84.36	19.04	0.00	0.00	14.69	14.15	2.08	2.08
	148.88.2.37	7.97	12.01	0.00	0.00	5.37	5.58	8.00	4.00	8.72	7.84	2.00	0.00
	130.192.239.1	12.28	14.34	2.00	4.00	10.51	12.49	2.04	2.04	13.08	10.53	0.00	0.00
Average		33.93	16.89	1.60	1.12	29.20	19.01	3.20	1.86	13.41	13.04	1.16	0.94

Table 5.5: Experimental result III of RTO estimation (Date: Wednesday, Sept. 26, 2001)

Pingee Location	Pingee Host	10:00am				14:00pm				18:00pm			
		OH (%)		ERR (%)		OH (%)		ERR (%)		OH (%)		ERR (%)	
		ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP	ARMA	MEP
Northern America	164.109.71.245	12.39	16.74	6.38	4.26	19.79	29.16	2,78	0.00	25.57	20.11	2.27	0.00
	18.181.0.31	14.80	20.41	2.00	0.00	15.60	21.32	0.00	0.00	15.77	14.84	0.00	2.00
	128.231.56.110	39.51	49.67	2.00	2.00	37.42	40.60	8.00	2.00	24.41	24.99	6.00	0.00
Asia	202.106.68.19	7.78	5.53	2.00	0.00	31.60	16.15	7.14	2.38	20.28	24.15	0.00	2.08
	202.112.144.70	10.30	11.73	3.57	0.00	12.62	12.64	0.00	2.70	48.46	34.30	3.48	3.48
	202.108.249.206	12.24	8.7	0.00	2.00	4.29	3.37	4.65	2.32	9.36	8.38	2.22	0.00
Europe	212.160.117.137	106.37	72.72	6.67	3.33	6.72	11.32	5.71	0.00	30.39	29.53	10.53	5.26
	148.88.2.37	4.69	7.94	8.00	4.00	4.61	4.50	0.00	0.00	8.90	10.44	8.00	6.00
	130.192.239.1	7.71	11.43	4.00	2.00	7.43	7.74	2.00	0.00	8.07	13.78	6.00	0.00
Average		23.98	22.76	3.85	1.94	15.56	16.31	3.06	1.04	21.25	20.06	4.28	2.09

Figure10, 11, 12, and 13 show the curves of the measured RTT, estimated RTO by using ARMA and estimated RTO by using MEP for some typical scenarios from the se of experiment results.

For the scenario shown in Figure 10, the performance of ARMA and MEP are close. However, for Figure 11, 12 and 13, the OH's of MEP are much smaller than those of ARMA while the ERR's are approximately at the same level.

In summary, on the whole, the MEP algorithm generates tighter correct delay boundary estimations than ARMA. Thus, network bandwidth could be used more efficiently.

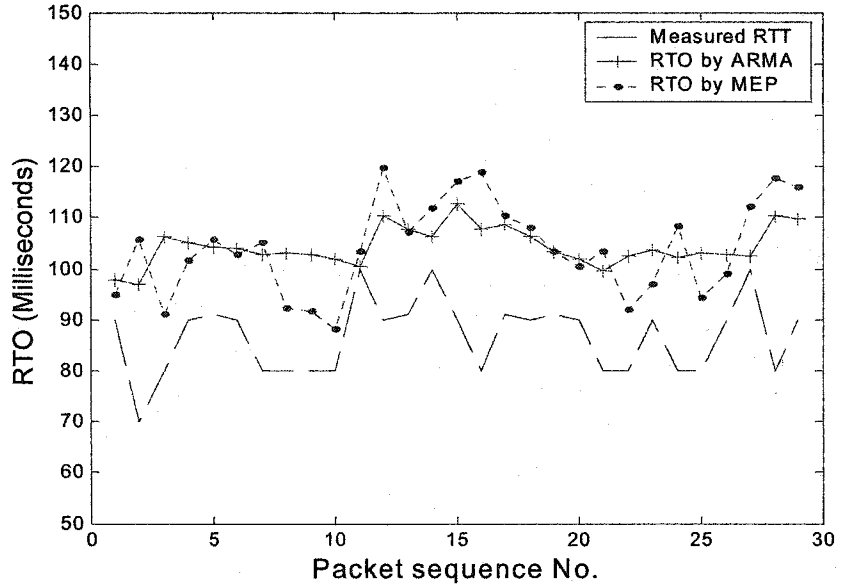


Figure 5.10: Comparison of RTO estimation between ARMA and MEP--Scenario I

OH_{ARMA}=20.38, ERR_{ARMA}=2.78; OH_{MEP}=19.51, ERR_{MEP}=0

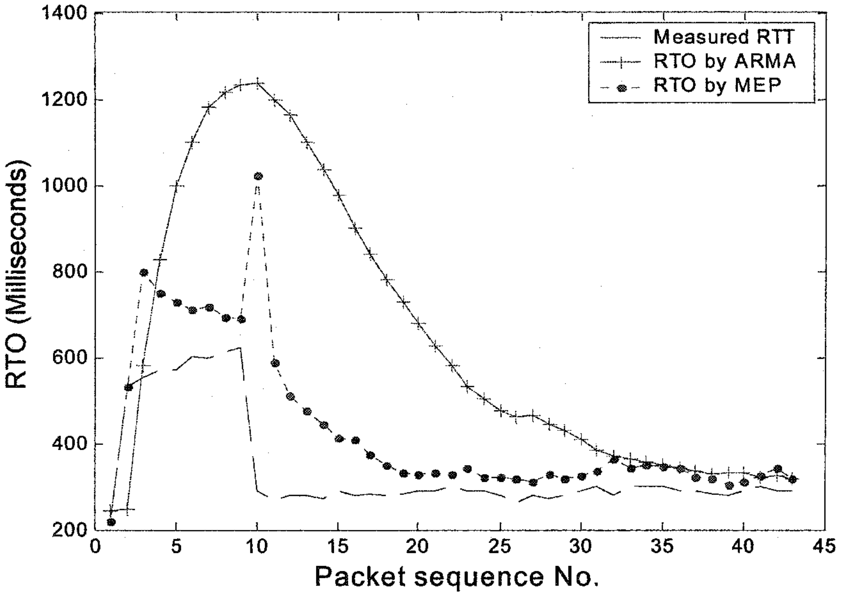


Figure 5.11: Comparison of RTO estimation between ARMA and MEP--Scenario II

OH_{ARMA}=89.27, ERR_{ARMA}=2.1; OH_{MEP}=27.54, ERR_{MEP}=0

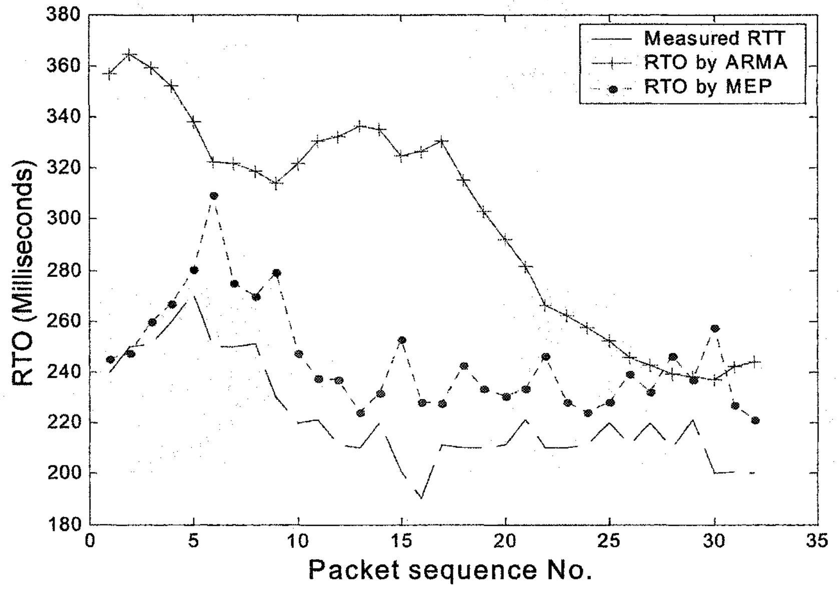


Figure 5.12: Comparison of RTO estimation between ARMA and MEP--Scenario III

OH_{ARMA}=35.22, ERR_{ARMA}=0; OH_{MEP}=10.52, ERR_{MEP}=2.7

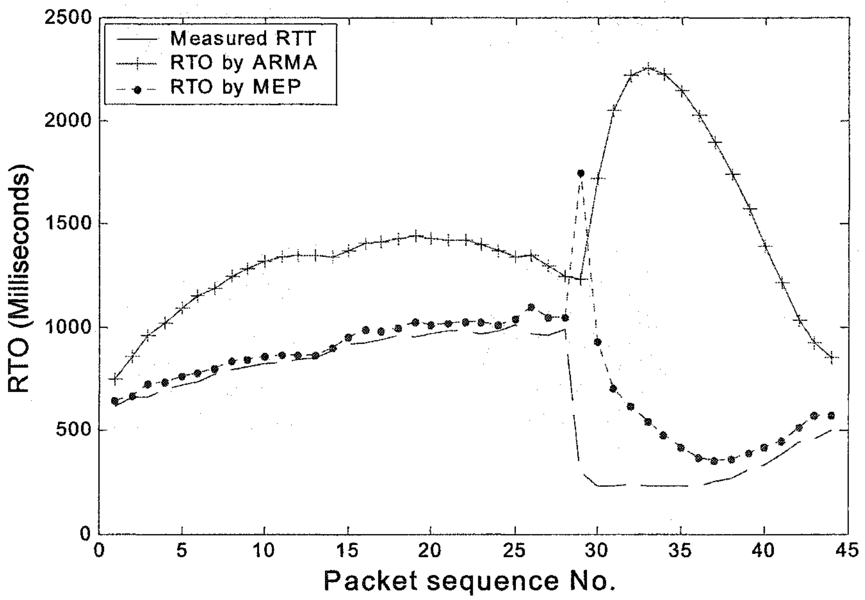


Figure 5.13: Comparison of RTO estimation between ARMA and MEP--Scenario IV

OH_{ARMA}=113.45, ERR_{ARMA}=0; OH_{MEP}=20.06, ERR_{MEP}=0

5.5 Summary

In this chapter, we analyze Internet roundtrip time (delay) statistically and introduce a novel algorithm for RTT and RTO estimations.

First, we examine RTT time series using both linear and nonlinear statistical methods. In particular, we investigate whether RTT time series possesses any structures or it is completely random. From the experimental results, the histogram of the RTT time series is a nonsymmetrical distribution that is best fitted by a constant plus a gamma distribution. The "skewed right" characteristic indicates that the RTT time series is not a Gaussian or uniform white noise. From the power spectrum and autocorrelation plots, it is found that the RTT time series is linearly correlated between adjacent and near-adjacent observations. The mutual information plot does not show high values of mutual information when time lag is lager than and close to 0. Intuitively, we infer that there are weak or no nonlinear correlations among observations.

Second, we revisit the traditional parameter-fixed ARMA model that is employed for RTO estimation in TCP implementation. The ARMA method presumes implicitly that the dynamics of RTT is dominated by a slow stationary process and independent of concurrent load, routing and path etc. However, the RTT dynamics of the real Internet is quite complicated. It evolves with time. For a measured RTT sequence, it might be dominated by a base plus a noise process in some intervals, but show strong structures in others. It may be a stationary process in one time while a nonstationary one or a complex combination of both in other times. This multi-structure characteristic is not well fitted by models with fixed parameters.

Then, we introduce a novel parameter-varying adaptive algorithm, called MEP, for RTT and RTO estimations based on the information theory and maximum entropy principle. In the model, the estimated roundtrip delay is regarded as a linear combination of the past observations. The estimation problem becomes how to find out the coefficients of the model. The key idea of the MEP algorithm is that the coefficients of the model are positive numbers and the sum of them is one. As a result, the coefficients can be regarded as a set of probability distributions and the maximum entropy principle can be used to calculate the coefficients. Because the coefficients are updated every ACK received, the model is adaptive and is able to track RTT dynamics quickly.

Finally, the performances of the MEP and ARMA methods are compared by using the RTT data collected from nine Internet links during three successive working days. The experimental results show that the MEP algorithm works better than the traditional ARMA method in both RTT and RTO estimations.

Chapter 6

Implementation Issues and Simulation Studies

This chapter addresses the implementation issues and simulation studies of the trinomial protocol. In Chapter 4, we claim that the trinomial protocol is an end-to-end rate-based transport protocol, which is specifically developed for teleoperation and e-service robotic systems. The trinomial protocol meets with the data transmission requirements of teleoperation and e-service robotic systems, such as minimized delay and delay jitter, smooth transmission rate in the steady state, and quick transient response to network bandwidth variations. In the mean time, the protocol satisfies the constraints on transport protocols, such as responsiveness, inter-protocol fairness convergence, intra-protocol fairness convergence and efficiency convergence. In this chapter, we explore how to implement the trinomial protocol on the Internet and evaluate its performances through extensive simulation and comparison studies.

6.1 Implementation Issues

The machinery of the trinomial protocol is mainly implemented at the source. A trinomial source sends data packets with sequence numbers and the trinomial sink acknowledges each packet, providing end-to-end feedback. Each acknowledgment (ACK) packet contains the sequence number of the corresponding delivered data packet. Using the feedback, the trinomial source can detect packet losses and sample roundtrip times (RTT's).

In previous chapters, we already know that the transmission rate of the trinomial protocol is adjusted depending on whether the network is congested. If the network is *on congestion*, the transmission rate is decreased immediately. Otherwise, the transmission rate is increased every roundtrip time (RTT). To implement a transport protocol, three issues must be addressed: congestion determination, rate adjustment algorithm, and rate adjustment frequency [57].

6.1.1 Congestion Signaling

There are two principal approaches to determining whether the network is congested [22]: loss-based scheme [26] and Explicit-Congestion-Notification-based mechanism (ECN-based) [60,122].

In the ECN-based mechanism, the congested routers signal congestion to end hosts *explicitly*. Routers mark packets along the congested links, and the receiver returns these congestion marks to the sender in a transport-specific manner. To signal congestion, the routers set the Congestion Experienced (CE) state in the IP header of the ECN-capable packets. The receiver returns this signal to the sender by setting the ECN-Echo (ECE) flag in the data packet header of subsequent acknowledgements. In TCP implementation, for example, to ensure reliable delivery of this signal, the receiver continues to set the ECE flag in acknowledgements until a Congestion Window Reduced (CWR) flag is received, implying the sender has reacted to the congestion. Unfortunately, the design of ECN requires routers and receivers to explicitly and correctly participate in the congestion control loop.

Although the ECN-based mechanism sounds more accurate than the loss-based scheme, most routers in today's Internet still adopt the DropTail queue management scheme and do not support the ECN-based mechanism. On the contrary, the loss-based mechanism does not rely on any explicit congestion signal from the network.

Consequently, in today's Internet, packet losses seem to be the most feasible *implicit* feedback signal to determine whether the network is congested. Signalling congestion via packet drops has been proven to be a simple and robust mechanism by the success of TCP implementations. Hence, in the trinomial protocol implementation, we still adopt the loss-based mechanism to determine whether the network is congested.

6.1.2 Congestion Determination

The trinomial protocol uses both timeouts (timer-based) and gaps (ACK-based) in the ACK sequence space to detect whether a packet is lost [22].

The trinomial protocol maintains an estimate of the roundtrip time (RTT) based on the MEP algorithm shown in Chapter 5 and calculates the roundtrip timeout (RTO) based on the Jacobson's algorithm [22,26]. The trinomial protocol couples the timer-based loss detection to the packet transmission. The source maintains a record for each transmitted packet, containing the sequence number, departure time, transmission rate and status flag. The collection of records for outstanding packets is called the transmission history. Before sending a new packet, the source checks for a potential timeout among the outstanding packets using the updated value of the RTO estimate. Then it traverses through the transmission history and detects all the timeout losses.

The ACK-based loss detection mechanism in the trinomial protocol is based on the same intuition as the fast-recovery scheme in TCP implementations. To limit the amount of overshoot during the increase phase, a trinomial source needs to detect congestion (*i.e.* packet loss) as early as possible. If the trinomial source receives an ACK that implies delivery of three packets after the missing one, the packet is considered lost.

It should be mentioned that, to determine network congestion, the ACK-based mechanism functions for the most part. The timer-based mechanism is mainly used to detect clustered losses at once. Thus, the timeout scheme works as a backup for critical scenarios such as a burst of losses.

6.1.3 Rate Adjustment Algorithm

As we have already known in Chapter 5, the data transmission rate of the trinomial protocol is controlled by adjusting the inter-packet-gap (*IPG*) in the source [49]. In the absence of packet losses, it is considered that there is extra bandwidth available in the network and the transmission rate is increased periodically. To increase sending rate according to (4.5), *IPG* is updated iteratively based on (6.1):

$$IPG_{i+1} = \frac{IPG_i \times W}{IPG_i + W}, \qquad (6.1)$$

where IPG_i is the interval between the succeeding packets for the i^{th} updating of the transmission rate and W is the desired increase rate in *packets per packet/second* [26, 49]. Thus, W actually has the dimension of time and it determines the value of increase step height.

If the transmission rate is updated every T seconds and the number of packets sent during each step is increased by k every step, we can get

$$W = T/k, \qquad (6.2)$$

According to (4.6), to reduce transmission rate when congestion occurs, the value of *IPG* should be updated based on (6.3).

$$IPG_{i+1} = IPG_i/(1-\beta), \qquad (6.3)$$

6.1.4 Rate Adjustment Frequency

Rate adjustment frequency specifies how often the rate is changed. The optimal adjustment frequency depends on feedback delays. The feedback delay is the time between changing the rate and detecting the network's reaction to that change. Feedback delays in ACK-based schemes are in the order of one RTT. In [40], it is suggested that *rate-based protocols should adjust transmission rate not more than once per RTT.* Changing the rate too often leads to unnecessary oscillations whereas infrequent change results in an unresponsive behaviour [22,49].

According to (4.5), the trinomial protocol adjusts the transmission rate once every RTT using (6.1). The time between two subsequent adjustment points is called a step. The data transmission rate is controlled by adjusting the inter-packet-gap (*IPG*) in the source and the rate updates once every RTT. This is realized relying on an estimation of next RTT to determine the value of inter-packet-gap. Too large RTT estimate causes a conservative protocol while too small RTT estimate results in an aggressive protocol. We have already discussed how to estimate RTT in Chapter 5.

At the beginning of each step, a timer, called step-timer, is set to the recent estimate of next RTT and the value of *IPG* is calculated based on (6.1). The value of *IPG* remains unchanged until the step-timer expires or a packet loss occurs. If no loss is detected, *IPG* is decreased and a new increasing step is started. Adjusting the value of *IPG* once every RTT has a nice property: packets sent during one step are likely to be acknowledged during next step, which allows the source to observe the reaction of the network to the previous adjustment before making a new adjustment.

As mentioned earlier, *W* in (6.1) has the dimension of time and is the only parameter that controls the increasing rate of transmission rate. One immediate question is how to calculate *W*. According to [26], if *IPG* is updated once every RTT and we choose *W* to be equal to RTT, the number of packets sent during each step is increased by 1 every step. Similarly, if *IPG* is updated once every *T* seconds and we choose *W* to be *T/k*, the number of packets sent during each step is increased by *k* every step. Combining with the increase rule of the trinomial protocol, (4.5), we get:

$$W = R(\frac{\alpha}{i})^r \tag{6.4}$$

where *R* is the value of RTT and *i* is the number of RTT's in the increasing phase.

So far, we have known how to adjust the trinomial sending rate in implementation.

6.2 Simulation Studies

In this section, we evaluate the performances of the trinomial protocol through simulations. We use the widely adopted network simulation tool, *ns-2*, which was developed by DARPA through the VINT project at LBL, Xerox PARC, UC Berkeley, and USC/ISI [124], for simulation studies. First, we examine the individual capabilities of the trinomial protocol, such as delay, delay jitter, packet loss rate, smoothness of transmission rate in the steady state, and transient response speed to network bandwidth variation. Then, we explore the social behaviours, e.g., responsiveness, inter-protocol fairness (TCP-compatibility), intra-protocol fairness and efficiency. In the simulations, the protocol parameter γ takes the value of 3 and β=0.05. The value of α depends on the initial sending rate in each increase phase and is updated according to (4.18).

6.2.1 Network Topologies

The two network topologies used in the simulations are shown in Figure 6.1 [22] and 6.2[95].

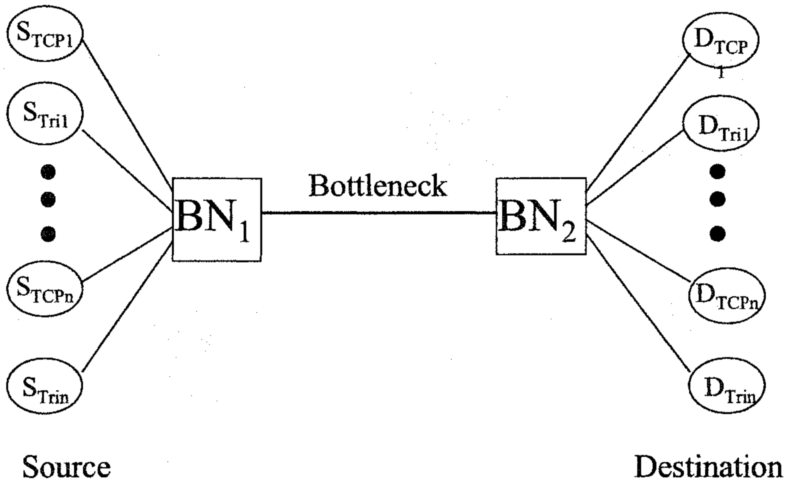


Figure 6.1: Network topology I

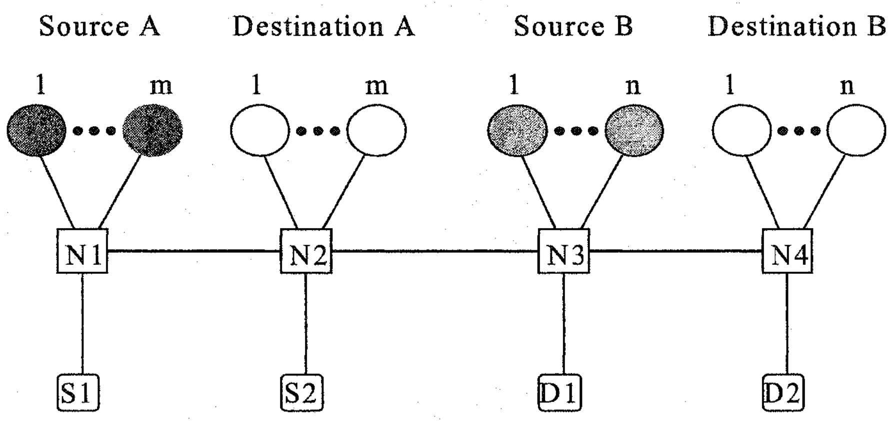


Figure 6.2: Network topology II

In topology I, the link between node BN_1 and node BN_2 is always the bottleneck of traffic. All other links (side links) have higher bandwidth than the bottleneck. The switches (routers) implement the FIFO scheduling and Drop-Tail queuing except in the RED simulations. .

In topology II, at the beginning of simulations, there are m+1 connections sharing link N_1--N_2, two connections sharing link N_2--N_3 and n+1 connections sharing link N_3--N_4. By controlling the number of connections sharing the links, we can create situations, such as sudden bandwidth increase/decrease, for monitored flows.

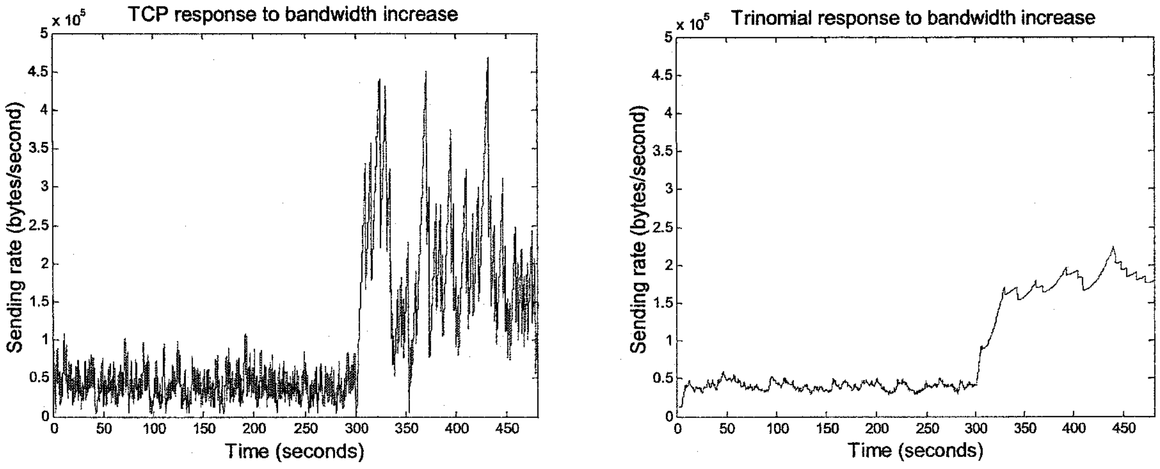
6.2.2 Transmission Rate

Topology I is deployed for the simulations to evaluate the smoothness, responsiveness and transient response of the transmission rate. The simulation parameters are given by Table 6.1.

Table 6.1: Simulation parameters for transmission rate studies

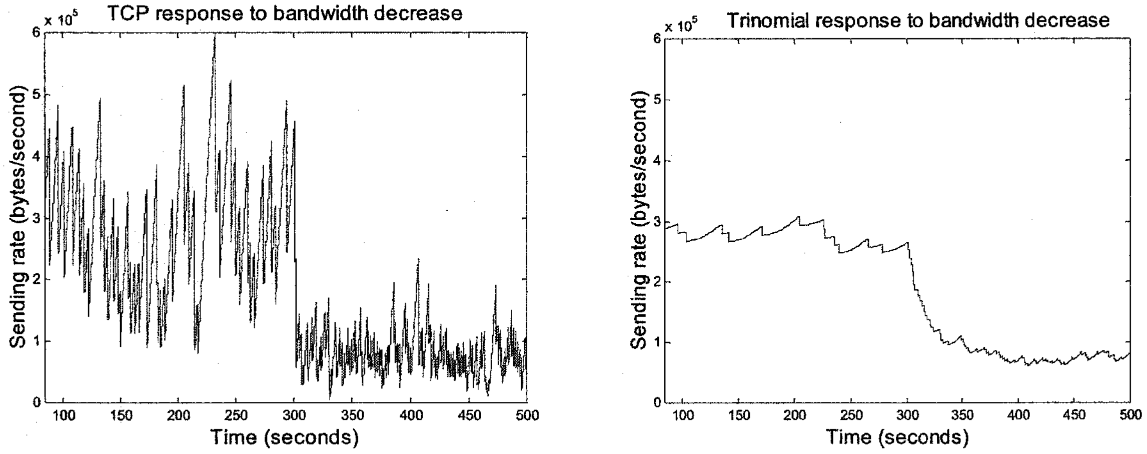
Parameter	**Value**
Packet size	1000 Bytes
ACK size	40 Bytes
Bottleneck delay	20ms
Bottleneck bandwidth	100Mbps
Delay of side links	2ms
Bandwidth of side links	100Mbps
Simulation length	1000seconds
TCP maximum window	1000
TCP source	FTP

For the scenarios shown in Figure 6.3, at the beginning of the simulation, one monitored flow (TCP or trinomial) shares the bottleneck, BN_1--BN_2, with 20 background TCP flows. When time goes to 300 seconds, all the background flows are gone. For the scenarios shown in Figure 6.4, at the beginning of the simulation, only the flow (TCP or trinomial) to be examined is in the bottleneck link. As time goes to 300 seconds, 20 background TCP flows come up suddenly to share the bandwidth of the bottleneck, BN_1--BN_2.



TCP Trinomial

Figure 6.3: Response to bandwidth increase



TCP Trinomial

Figure 6.4: Response to bandwidth decrease

From the results shown in Figure 6.3 and Figure 6.4, both the TCP and trinomial protocols respond to network bandwidth variation properly. Both protocols are responsive to network congestion. When there is an increase in network bandwidth, both protocols take extra bandwidth quickly as shown in Figure 6.3 while back off when there is a decrease in network bandwidth as shown in Figure 6.4.

The transient response of the trinomial protocol to network bandwidth variations is a little bit slower than that of TCP, however, the oscillation of TCP's transmission rate is too large and too fast. Compared to TCP, the transmission rate of the trinomial protocol is much smooth in the steady state.

The transmission rate of UDP is the smoothest because it could be constant. However, UDP is unresponsive to congestion and cannot adapt to network bandwidth variation because it has no mechanisms to detect network states such as extra bandwidth availability and congestion. This point is already shown by the simulation in Chapter 1.

6.2.3 Delay, Delay Jitter and Packet Loss Rate

We use topology I for the simulation to compare time delay, delay jitter and packet loss rate among TCP, UDP and the trinomial protocol. Table 6.2 gives the simulation parameters.

For time delay simulation, three flows (1 TCP, 1 UDP and 1 trinomial) to be studied share the bottleneck with a few background TCP flows. In the cases of loss rate study, the number of background TCP flows varies from 1 to 100, and the sending rate of the UDP flow is 0.005Mbps.

From the simulation results of delay and delay jitter given by Figure 6.5, we can see that the trinomial protocol has similar performances to UDP. Their delays are almost constant and the averages of their delays are much smaller than that of TCP. It is clear that the trinomial protocol is much better than TCP in terms of both delay and jitter since the trinomial protocol has no retransmission mechanism.

Table 6.2: Simulation parameters for delay and loss rate studies

Parameter	**Value**
Packet size	1000 Bytes
Trinomial ACK size	40 Bytes
Bottleneck delay	200ms

Bottleneck bandwidth	1Mbps
Delay of side links	100ms
Bandwidth of side links	5Mbps
Simulation length	200seconds
TCP maximum window	1000
TCP source	FTP
UDP source	CBR
Queue	RED
Queue Size	60
Background TCP flows	50

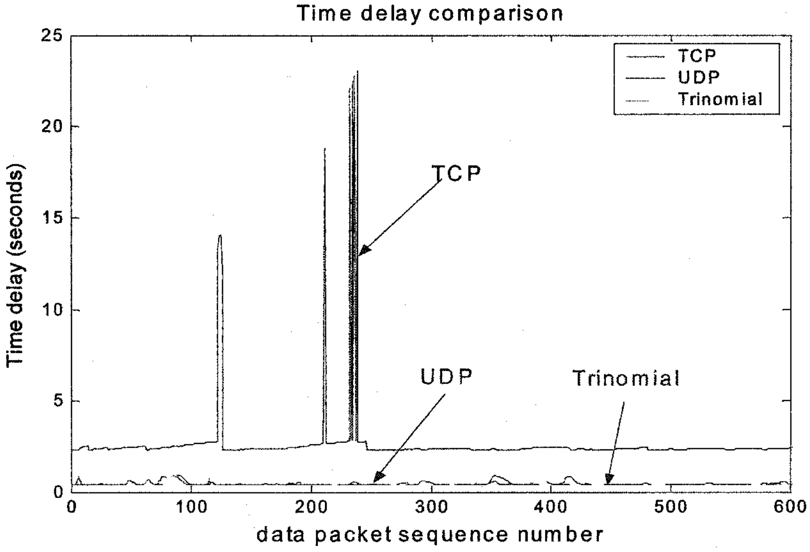


Figure 6.5: Time delay and delay jitter comparison among the trinomial protocol, TCP and UDP

From Figure 6.6, the trinomial protocol also has similar performance to UDP on packet loss rate, which is better than TCP. This point was not expected before simulation. We guess that this might be because they have different implementation mechanisms: TCP is window-based while both the trinomial protocol and UDP are rate-based.

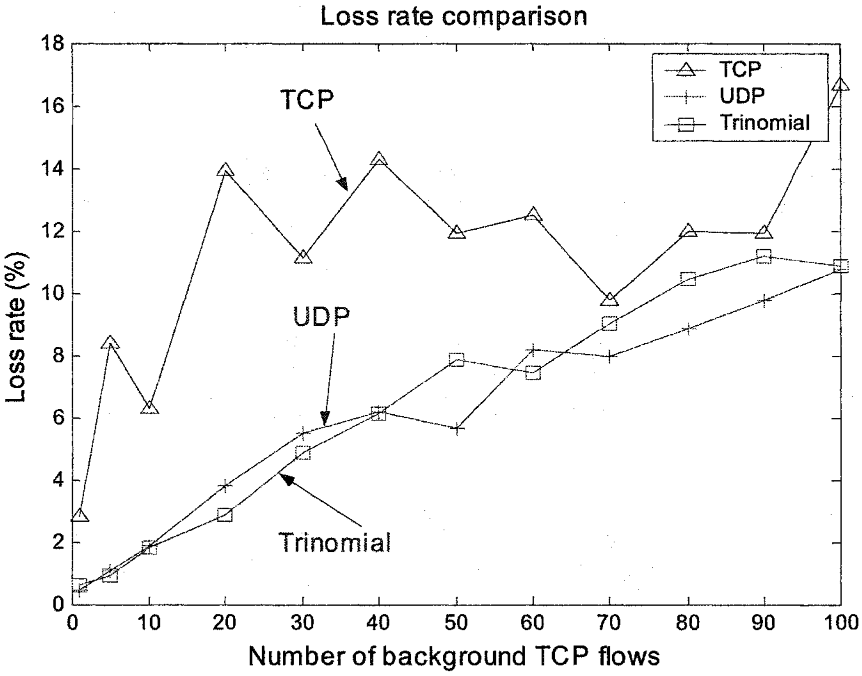


Figure 6.6: Packet loss rate comparison among the trinomial protocol, TCP and UDP

6.2.4 Inter-Protocol Fairness

To estimate inter-protocol fairness, which is the most important metric to characterize the social behaviour of a protocol, we introduce the notion of *throughput.*

The *average throughput* of a flow f using protocol p, T_{pf}, is defined as the total amount of data being received by the receiver during the whole measurement interval over time. In order to analyze the throughput distribution of different flows, we introduce *normalized throughput*. The normalized throughput of flow f using protocol p, T_{pf}^N, is defined as

$$T_{pf}^N = \frac{T_{pf}}{T^N},$$

where $T^N = \frac{\sum_{p\in P}\sum_{f\in F_p} T_{pf}}{\sum_{p\in P}|F_p|}$, where $|F_p|$ is the number flows using protocol p , F_p is the set of flows using protocol p, and P is the set of protocols in the traffic.

To be able to compare the throughput of different types of protocols in a direct way, we introduce *mean normalized throughput.* The mean normalized throughput of protocol p, $\overline{T}_p^N$, is thus defined as

$$\overline{T}_p^N = \frac{\sum_{f\in F_p} T_{pf}^N}{|F_p|}.$$

For inter-protocol fairness simulation studies, we still employ topology I shown in Figure 6.1. The simulation parameters are given in Table 6.3.

Table 6.3: Simulation parameters for TCP-Compatibility studies

Parameter	**Value**
Packet size	1000 Bytes
ACK size	40 Bytes
Bottleneck delay	20ms
Bottleneck bandwidth	15Mbps
Delay of side links	20ms
Bandwidth of side links	100Mbps
Simulation length	1000seconds
TCP maximum window	512
TCP source	FTP

As mentioned earlier, since today's Internet applications are dominated by TCP, we only show how the trinomial protocol coexists with TCP. In other words, we examine whether the trinomial protocol is *TCP-compatible* (*TCP-friendly*). In the simulation, n trinomial and n TCP flows share the bottleneck link (BN1---BN2) and n is varied from 1 to 100. All the sources are put on one end of the bottleneck link and all the destinations on the other side. The bandwidth of the side links is large enough compared to the one of the bottleneck link so that there is no congestion occurring in the side links.

To evaluate TCP-compatibility of the trinomial protocol in a wide range of network conditions, we have conducted simulations in various conditions in terms of different combinations of TCP implementations (Tahoe, Reno, NewReno, Sack, Vegas and Fack), queue managements (RED and DropTail), and traffic sizes. Figure 6.7, 6.8 and 6.9 are three typical examples of the simulation results showing how the trinomial protocol coexists with TCP.

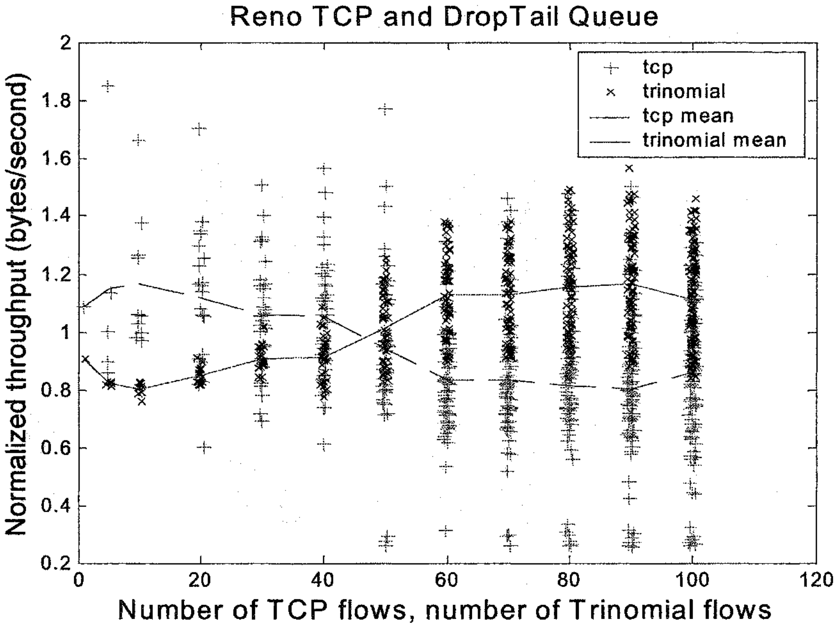


Figure 6.7: The trinomial protocol coexists with Reno TCP

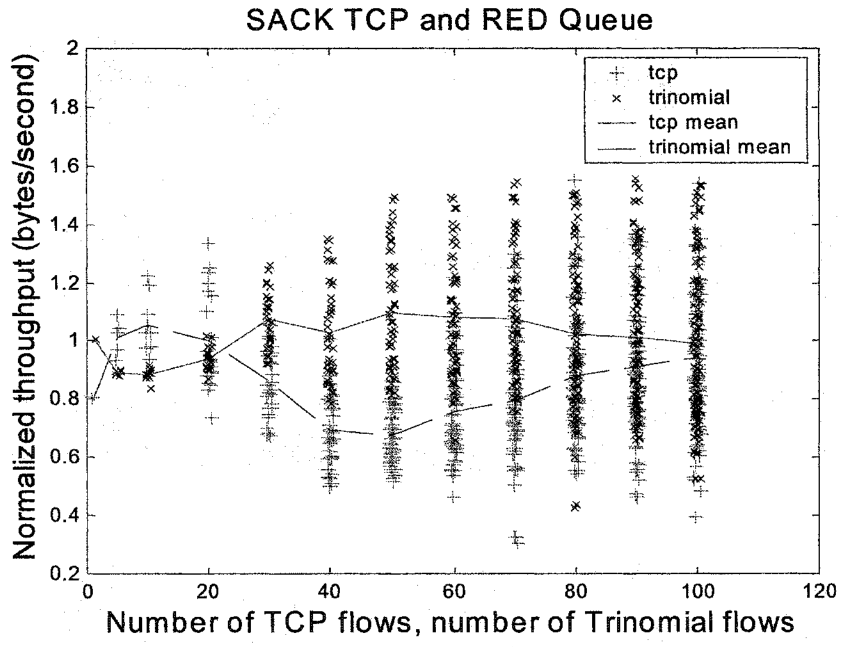


Figure 6.8: The trinomial protocol coexists with SACK TCP

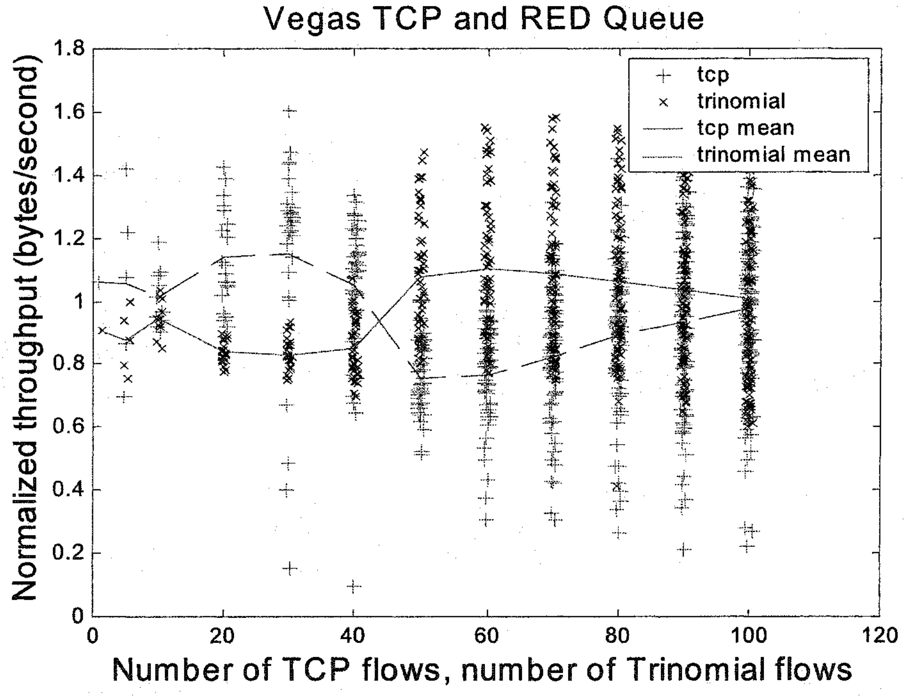


Figure 6.9: The trinomial protocol coexists with Vegas TCP

From the simulation results, we can see that although the variances of normalized throughputs for individual flows could be quite high, the mean normalized throughputs of both TCP and the trinomial protocol mostly fall into the range of [0.8, 1.2] when the number of flows, *n*, varies from 1 to 100. For Figure 6.8 and 6.9, when the number of flows, *n*, increases, the mean normalized throughputs of both TCP and the trinomial protocol converge to one. Compared to other *TCP-friendly* transport protocols, such as RAP [49], TFRC [75] and LDA [71], the trinomial protocol is reasonably fair to TCP in a wide range of traffic.

As shown by Figure 1.2 in Chapter 1, this harmony with TCP is *not* provided by UDP.

The reasons that the mean normalized throughputs of the trinomial protocol and TCP are not the same (100% fair) for all numbers of flows may mainly exist in the following four facts:

1) For simplicity, we employ the deterministic model when we model TCP transmission rate instead of the stochastic model that gives a more accurate description of TCP;
2) The slow-start process of TCP is not taken into account. In the context of long-lived sessions, the performance of the start-up phase is not crucial because the duration of this period is negligible in comparison to the session length. However, for short-lived and interactive sessions, the slow-start process might be relevant;
3) Errors in RTT estimation are unavoidable. As mentioned in previous chapters, the value of next RTT estimate determines the transmission rate of the trinomial protocol directly. A larger RTT causes a lower transmission rate while a smaller RTT results in a higher transmission rate;
4) The trinomial protocol and TCP use different implementation mechanisms. TCP is window-based while the trinomial protocol is rate-based.

In addition, the TCP retransmission and error-recovery mechanisms are not considered either.

6.2.5 Intra-Protocol Fairness

Topology II is used to examine the intra-protocol fairness property of the trinomial protocol. The simulation parameters are given in Table 6.4.

Table 6.4: Simulation parameters for intra-fairness convergence and efficiency convergence

Parameter	**Value**
Packet size	1000 Bytes
ACK size	40 Bytes
Bottleneck delay	10ms
Bottleneck bandwidth	10Mbps
Delay of side links	5ms
Bandwidth of side links	10Mbps
TCP maximum window	512
Queue	RED
Simulation length	200seconds
TCP source	FTP

In the simulation, *m* background TCP flows (source A→ destination A) and 1 flow to be studied (TCP or trinomial S1→D1) share link N1---N2. Similarly, *n* background TCP flows (source B→ destination B) and 1 flow to be studied (TCP or trinomial S2→D2) share link N3---N4. The two monitored flows (either two TCPs or two trinomials) share link N2---N3. At the beginning of the simulation, *m*=20 and *n*=0. So, the sending rate of the flow (S2 →D2) is higher than that of the flow (S1→D1). For the flow (S1→D1), link N1---N2 is the bottleneck while for the one (S2→D2), link N2---N3 is the bottleneck.

When time goes to 100 seconds, all the background TCP flows (source A→ destination A) terminate. So, for both flows to be studied (S1→D1 and S2→D2), the bottleneck becomes link N2---N3.

From the results shown in Figure 6.10, it is clear that the two monitored trinomial flows converge to the same sending rate, the *fairness line*, although their initial sending rates are different. Consequently, we can conclude that the trinomial protocol is intra-protocol fairness convergent. From Figure 11, we can see that the two TCP flows also converge to the *fairness line*, however, the oscillation in their transmission rates is too much. UDP flows do *not* converge to an intra-protocol fairness state since UDP flows are unaware of Internet dynamics as we see in Chapter 1.

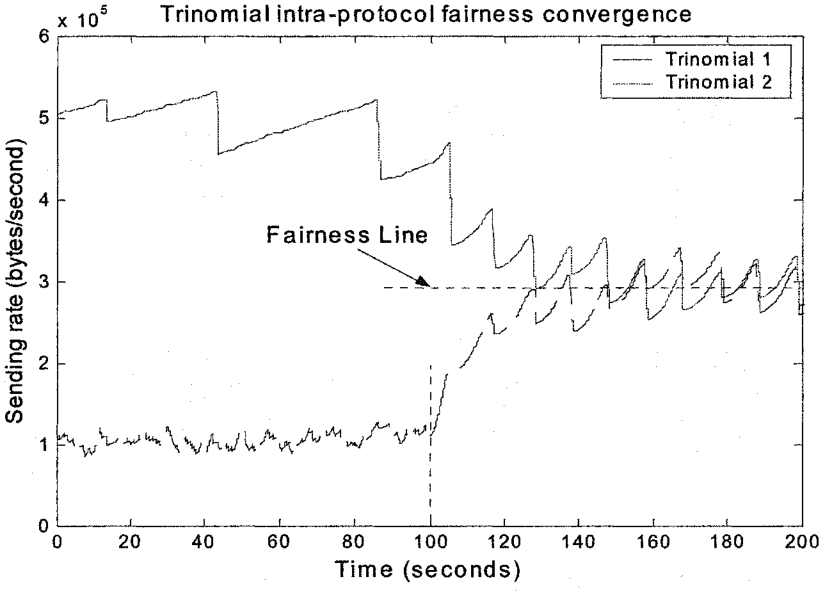


Figure 6.10: Intra-protocol fairness convergence of the trinomial Protocol

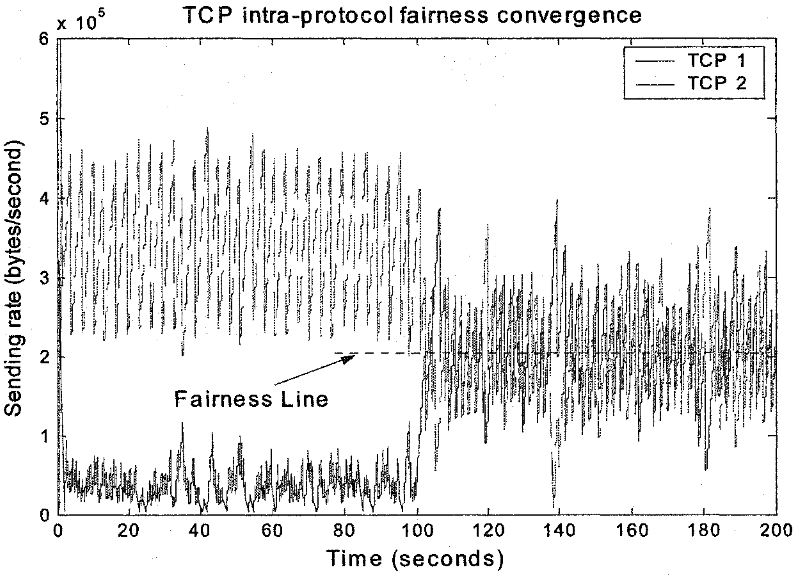


Figure 6.11: Intra-protocol fairness convergence of TCP

6.2.6 Efficiency

In this part, we use topology II shown in Figure 6.2. At the beginning of the simulation, m=15 and n=5, thus link N1---N2 and N3---N4 are bottlenecks for the flows of (S1->D1) and (S2->D2) respectively. When time goes to 100 seconds, all the background TCP flows are gone. Thus, there is sudden extra bandwidth available and link N2---N3 becomes the bottleneck for the two flows to be examined. The simulation parameters are the same as given in Table 6.4.

From the results shown in Figure 6.12 and 6.13, we can see that the two TCP flows and the two trinomial flows increase their sending rates quickly from 100 seconds until get around the new steady states, the *efficiency lines* (higher sending rates), and then oscillate around the new steady states. In other words, both TCP and the trinomial protocol are efficiency convergent and bounded. As mentioned earlier, the raw UDP protocol does not converge since it has no mechanisms to probe extra bandwidth availability.

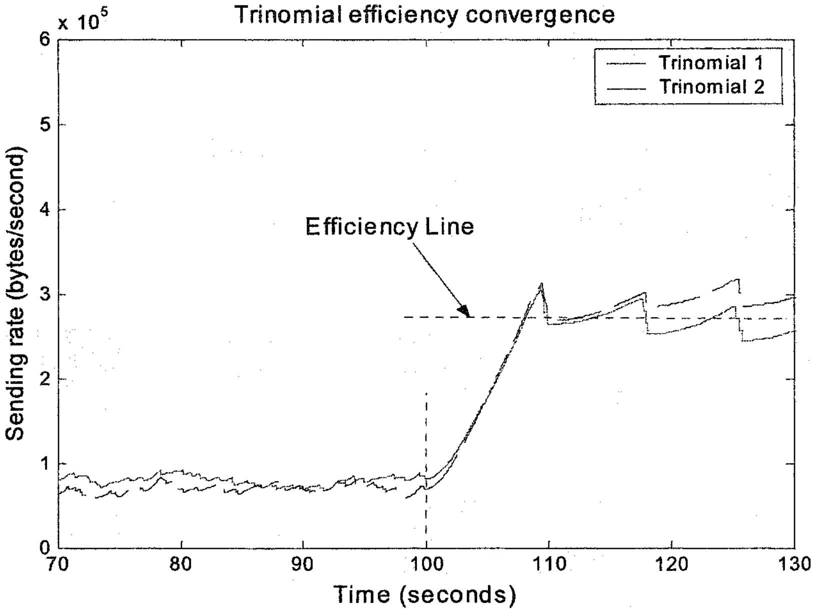


Figure 6.12: Efficiency convergence of the trinomial protocol

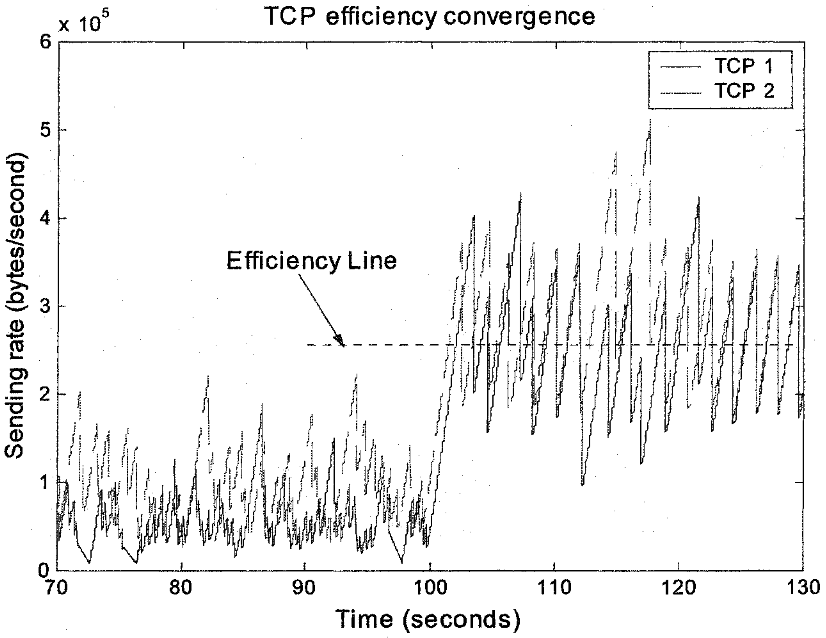


Figure 6.13: Efficiency convergence of TCP

6.3 Summary

This chapter discusses the implementation issues and evaluates the performances of the trinomial protocol through extensive simulation and comparison studies.

First, it is elaborated how to implement the trinomial scheme in the real Internet. The trinomial algorithm is mainly implemented at the source. It is a closed-loop mechanism in which the trinomial source sends data packets with sequence numbers and the trinomial sink acknowledges each packet, providing end-to-end feedback. Each acknowledgment (ACK) packet contains the sequence number of the corresponding delivered data packet. Using the feedback, the trinomial source detects losses and samples RTT's. The three key issues on protocol implementation: congestion determination, rate adjustment algorithm, and rate adjustment frequency, are discussed in details.

- The trinomial protocol adopts both the ACK-based and timer-based loss detection mechanisms to determine congestion. The ACK-based mechanism functions for the most part and the timer-based mechanism is mainly used to detect clustered losses, acting mainly as a backup for critical scenarios such as a burst of losses.
- The transmission rate of the trinomial protocol is controlled by adjusting the value of inter-packet-gap (IPG). In the absence of packet losses, it is considered that there is extra bandwidth in the network and the value of IPG is decreased periodically. When packet losses are detected, the value of IPG is increase immediately. The increase/decrease of IPG is based on (6.1) and (6.2).
- The trinomial protocol adjusts its transmission rate once every RTT. At the beginning of each step, a timer, called step-timer, is set to the recent RTT estimate and IPG is decreased. IPG remains unchanged until the step-timer expires or a packet loss occurs. If no loss is detected, IPG is decreased and a new step is started.

Second, the trinomial protocol is simulated extensively and compared with both TCP and UDP by using the *ns-2*, a widely accepted network simulator. To examine TCP-compatibility in a wide range of network conditions, we have carried out simulations in various conditions in terms of different combinations of TCP implementations (Tahoe, Reno, NewReno, Sack, Vegas and Fack), queue managements (RED and DropTail), and traffic sizes. For other performance evaluations, we have simulated both the RED and DropTail queues. The simulation results validate the theoretical findings shown in Chapter 4.

- The trinomial protocol minimizes delays and delay jitter as UDP does because it has no retransmission mechanisms. Its delays are almost constant and the average is much lower that that of TCP. Its packet loss rate is also much lower than that of TCP. In the steady state, its transmission rate is rather smooth compared to that of TCP. When available network bandwidth varies, it adapts to the variation quickly. TCP also responds to network bandwidth changing quickly, however, the fluctuation in its transmission rate is too large and too fast. UDP does not show any response to network bandwidth changing at all because it has no mechanisms to detect network states.
- The trinomial protocol meets with all the constraints on transport protocols: it is responsive to network congestion as TCP does; it is inter-protocol fair, i.e., reasonably compatible to TCP; it is intra-protocol fairness convergent; and it is efficiency convergent. On the other hand, UDP does not satisfy *any* of these constraints.

Chapter 7

A Platform for E-Service Mobile Robotic Systems

In this chapter, we introduce a new modular platform for e-service mobile robotic systems. On this platform, advanced remote compensation control algorithms, interface designs and applications can be tackled for the long-term goal of the research: a network-enabled system that can provide real-world applications such as tele-monitoring, tele-assistance and tele-embodiment etc.

7.1 System Hardware Architecture

The system hardware configuration is shown in Figure 7.1.

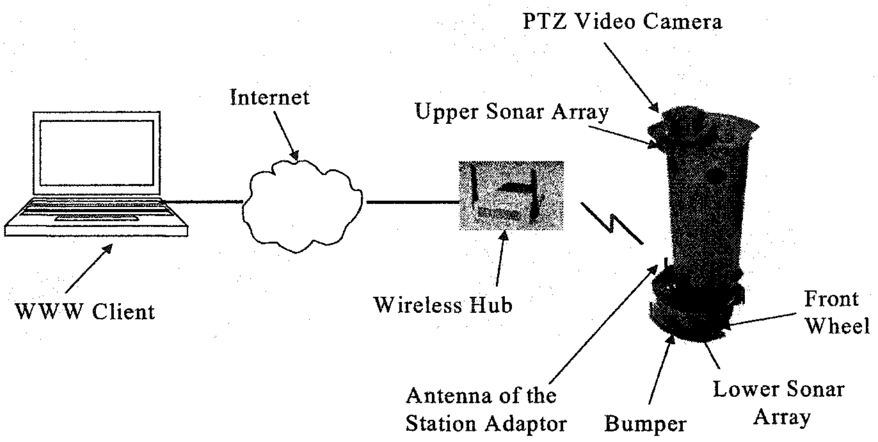


Figure 7.1: System hardware configuration

The system hardware mainly consists of a commercial *Pioneer 2 PeopleBot* (P2PB) mobile robot. The P2PB mobile robot is connected to the Internet through a pair of wireless LAN adaptors. Users can control the mobile robot remotely on any regular PC with a web browser and Internet access.

7.1.1 Mobile Robot

The P2PB mobile robot, as shown in Figure 7.2, is provided by the *ActiveMedia Robotics* [132]. It contains the basic components for motor control, sensing and navigation, including battery power, drive motors and wheels, position/speed encoders, bumper switches, integrated sonar ranging sensors and a visual system. They are all managed by an on-board micro-controller and the corresponding server software. The P2PB mobile robot uses a 20MHz Siemens 88C166-based microcontroller, with independent motor/power and sonar controller boards for a versatile operating environment. The controller has two RS232-standard communication ports and an expansion bus to support various accessories.

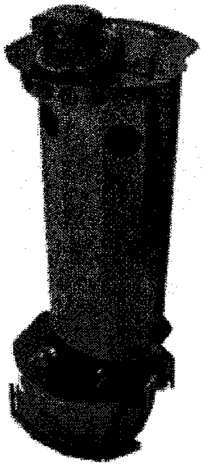


Figure 7.2: P2PB mobile robot

The drive system (Figure 7.3) of the P2PB mobile robot uses high-speed, high-torque, reversible-DC motors. It has a caster wheel and two drive wheels. It can move with an approximate maximum speed of 60 cm per second. Each front drive motor includes a high-resolution optical quadrature shaft encoder that provides 9,850 ticks per wheel revolution (19 ticks per millimeter) for precise position and speed sensing and advanced dead reckoning. The error in distance is about 1 cm per meter; the error in rotation is up to 8 degrees per revolution.

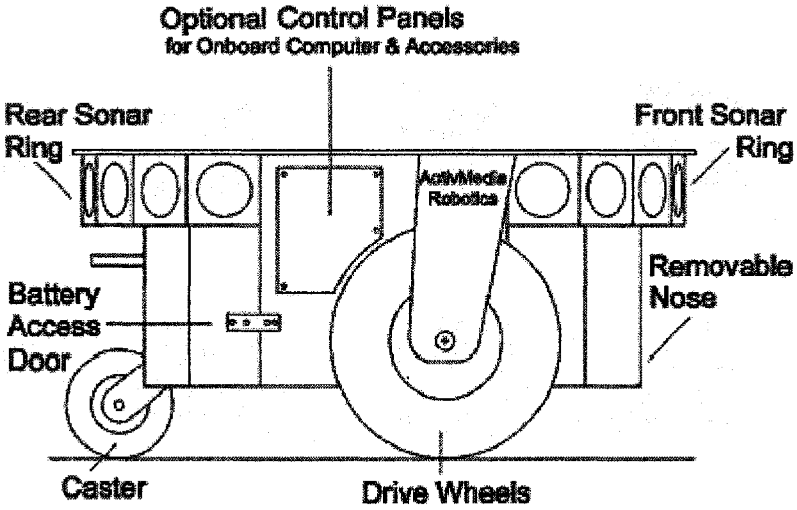


Figure 7.3: P2PB drive system

The P2PB mobile robot provides two range-finding sonar arrays. One array, affixed under the front of the Deck and atop the Nose, provides forward and side-range sensing. The other, an optional sonar array is attached just beneath the rear Deck and provides rearward, as well as side sensing. Each array contains eight sonars. Totally up to sixteen sonars surround the robot. The sonar positions are fixed in both arrays: one on each side, and six facing outward at 20-degree intervals, together providing 360 degrees of nearly seamless sensing. The sonar sensitivity ranges from 10 centimetres (6 inches) to more than 5 meters (16 feet). (Objects closer than 10 centimetres are not detected.). The sonar's firing pattern may also be controlled through software; the default is left-to-right in sequence for the forward array and right-to-left on the rear. One sonar from each array *pings* simultaneously. The sonar array is shown in Figure 7.4.

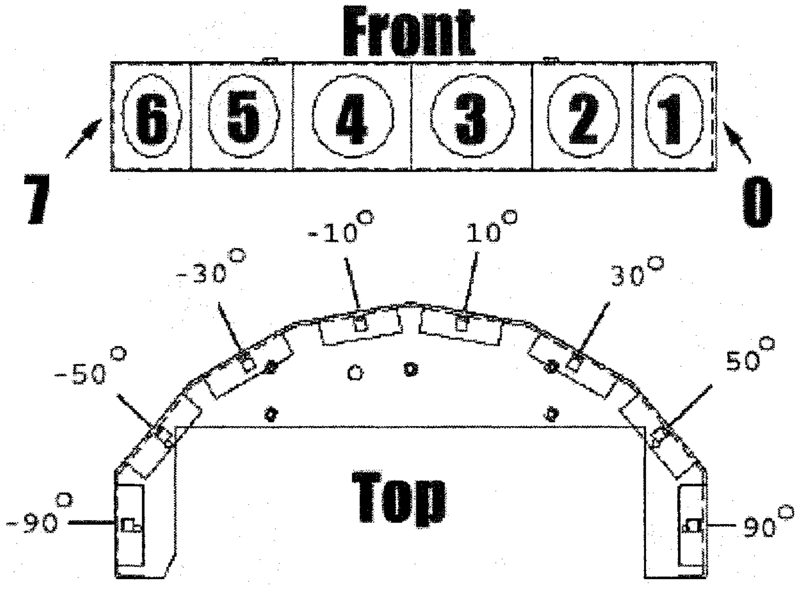


Figure 7.4: P2PB sonar array

The P2PB mobile robot has a console that consists of a liquid-crystal display (LCD), MOTORS and RESET control buttons and indicators, and an RS232-compatible serial port at the 9-pin DSUB connector on the front and top of the Deck. The layout of the console is shown in Figure 7.5. The P2PB's standard electronics reside on three main boards: The micro-controller is mounted under the console deck; a power/motor controller board is mounted to the battery box inside the robot; and a sonar controller (one for each array) is mounted in the base of the sonar array.

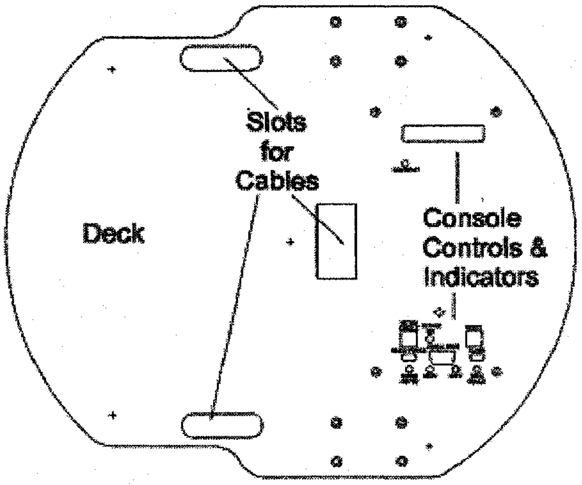


Figure 7.5: P2PB console and deck layout

7.1.2 Visual System

The visual system is detachable and mounted on the head of the P2PB mobile robot. It mainly consists of a Sony D30/31 pan-tilt-zoom (PTZ) color camera (shown in Figure 7.6) and PTZ system software, which is for camera pan-tile-zoom angle control and image grabbing. .

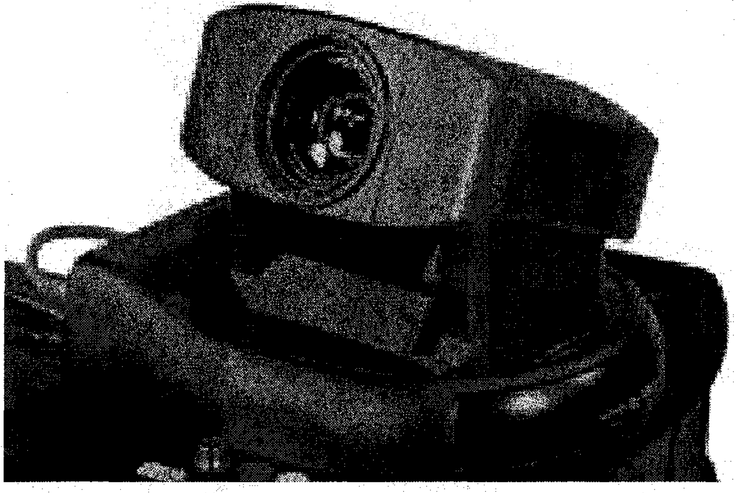


Figure 7.6: Sony D30/31 PTZ video camera

7.1.3 Wireless Ethernet Adapter

A pair of BreezeCom's BreezeNet PRO 11 indoor wireless Ethernet adaptors (as shown in Figure 7.7) is employed to connect the P2PB mobile robot to the Internet. The BreezeNet PRO.11 indoor adaptors adhere to the IEEE 802.11 standard, working seamlessly with other 802.11 Frequency Hopping wireless LAN products. By operating in the 2.4 GHz unlicensed ISM band, it offers data communication rate of up to 3 Mbps, at distances of up to 150m (500ft) indoors, and roaming speeds up to 100km/h (60mph). In the platform, an AP-10 adaptor (access point to hub slot) and a SA-10 adaptor (station adaptor installed inside the P2PB mobile robot) are used.

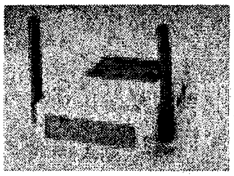


Figure 7.7: Wireless hub

7.2 System Software Architecture

The system software employs a client-server architecture for robot control and feedback information display. In the model, there are two servers, *Video Server* and *Control Server*, and the two corresponding clients, *Video Applet* and *Control Applet*. The web browser is any *Java-enabled* web browser and the web server is a *Linux Apache web server*.

On the client side -- the web browser, the Video Applet is responsible for live image decoding and display etc., and the Control Applet is for intercepting and interpreting human control commands (mouse-click events in the control panel) and display of other information feedback, such as robot position, speed and ranging etc.

On the server side -- the P2PB mobile robot, the Video Server is in charge of video grabbing, compression, encoding and transmission, while the Control Server takes care of motor control, sonar firing, camera pan-tile-zoom control, and other sensor sampling and encoding.

The corresponding server and client pairs exchange information through the web server, the Linux Apache web server.

A brief functional software structure of the platform is shown in Figure 7.8.

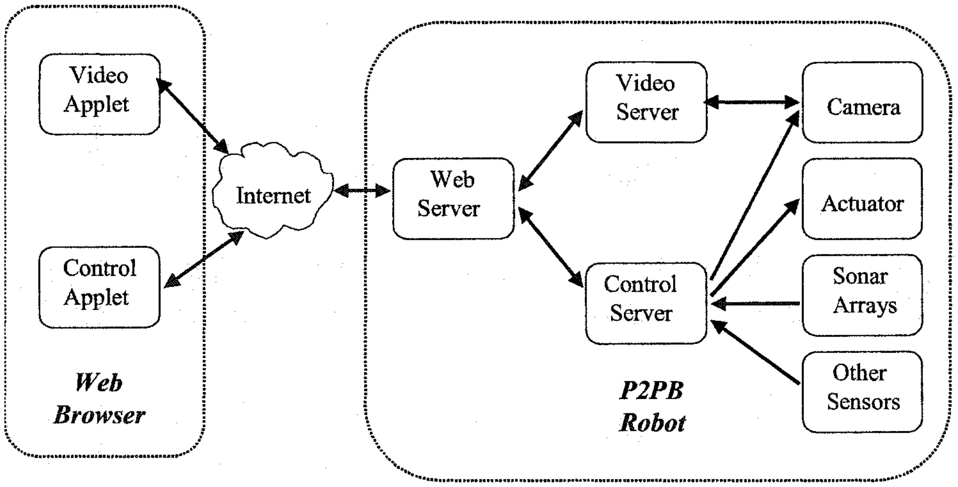


Figure 7.8: System software architecture

7.2.1 Data Types and Corresponding Transport Protocols

Various messages need to be exchanged between the robot and the human operator: non real-time administration and configuration data, control commands, live scene images, robot states such as position and speed, and sonar ranging data. The variety of information requires different transfer modes. Generally, there are four classes of data need to be transferred as follows:

- Administrative data (such as access control, configuration data etc.): small packet size, once-for-all transfer, no real-time constraints, and requiring reliable delivery;
- Control commands (such as desired translation velocity and rotation angle etc.): small packet size, random timing in nature, real-time, and the most current data are preferred should packets were lost;
- Image data (the most important and costly information feedback): periodic transfer, real-time, requiring significant bandwidth, and the most current data are preferred should packets were lost; and
- Other scene and robot state information: (such as position, speed and ranging data): small packet size, periodic transfer, real-time, and the most current data are preferred should packets were lost.

From above categorization, it is clear that data communications in e-service mobile robotic systems require *real-time* delivery except for the once-for-all administrative data. Consequently, in the implementation, we adopt the TCP protocol for the communications of administrative data. A TCP connection is opened when a teleoperation session is started and closed once the teleoperation is geared up. The trinomial protocol proposed in this thesis is employed for the transmission of information on live scene images, sonar ranging, and robot positions and speeds. Since the human operator issues control commands based on his/her personal judgments and decisions, the timing of these control signals is random in nature and controlled solely by the human operator. The transmission frequency depends on how often the specific user controls the mobile robot. Thus, the data transmission rate is largely controlled by the human operator instead of the transport protocol. As a result, we still utilize UDP for control command delivery. The usage of UDP is generally acceptable because control commands are small-size packets and it should work if the human operator does not issue a burst of commands when the network is heavily congested.

7.2.2 Server-Client Control Structure

In the server-client control model, the robot's Control Server—*the Pioneer 2 Operating System* (P2OS)—works to manage all the low-level details of the mobile robot system. These include operating the motors, firing the sonars, collecting sonar and wheel encoder data, managing the battery power, controlling the pan-tilt-zoom of the camera, and so on -- all on command from and reporting to the corresponding client application, the Control Applet, through the Linux Apache web server.

With this client-server architecture, the client application – the Control Applet, is insulated from the lowest level of controls. The Control Applet communicates with the Control Server, P2OS, via the P2OS client-server interface -- the communication packet protocol. The server-client control structure is shown in Figure 7.9.

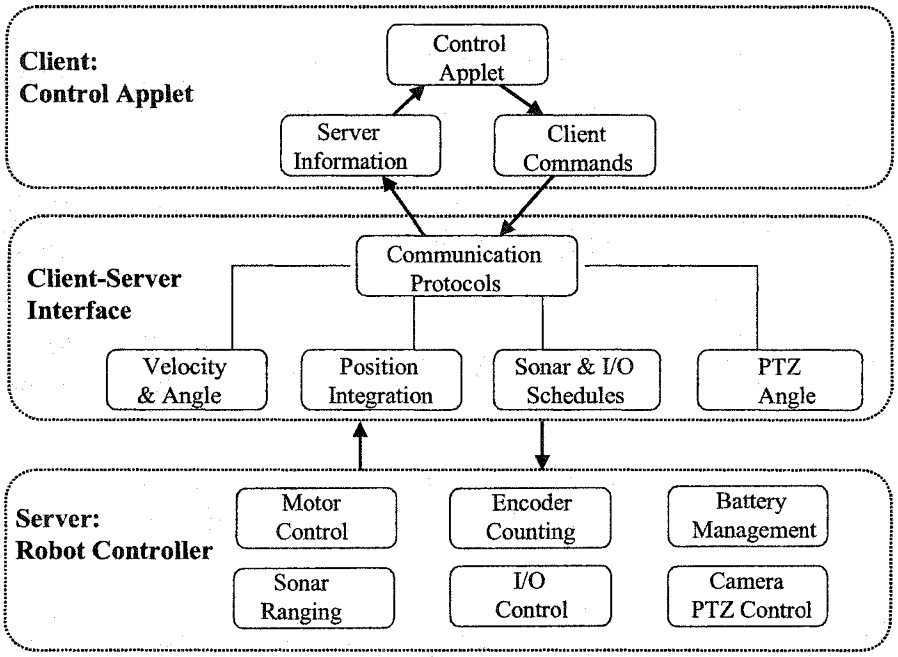


Figure 7.9: Server-client control structure

7.2.2.1 Communication Packet Protocol

The Control Server, P2OS, communicates with the client application, the Control Applet, using a special packet protocol for both command packets from client to server and server information packets (SIP's) from server to client. Both are bit streams consisting of four main elements: a two-byte header, a one-byte count of the number of subsequent packet bytes, the client command and its arguments or the server information data bytes, and, finally, a two-byte checksum. The main elements of the communication packet protocol are shown in Table 7.1.

Table 7.1: Main elements of the communication packet protocol

Component	**Bytes**	**Value**	**Description**
Header	2	0xFA, 0xFB	Packet header; same for client and server
Byte Count	1	N + 2	Number of subsequent data bytes, including the Checksum word, but not the Byte Count. Maximum 200 bytes
Data	N	Command or SIP	Client command or server information packet (SIP)
Checksum	2	Computed	Packet integrity checksum

7.2.2.2 Packet Data Types

Client commands and server information packets (SIPs) contain several data types, as defined in Table 7.2.

Table 7.2: Data types of the communication packet

Data Type	**Bytes**	**Order**
Integer	2	b0 low byte; b1 high byte
Word	4	b0 low byte; b1 high byte
String	Up to ~ 200, Length-prefixed	b0 length of string; b1 first byte of string

7.2.2.3 Server Information Packet (SIP)

The standard P2OS Server Information Packet (SIP) informs the client about a number of operating states and readings, using the order and data types described in Table 7.3.

Table 7.3: Server information packet (SIP)

Name	**Data Type**	**Description**
Header	Integer	Exactly 0xFA, 0xFB
Byte Count	Byte	Number of data bytes + 2 (checksum); must be less than 201 (0xC9)
Status/Packet Type	Byte = 0x3S; where S	Motors status
	2	Motors stopped
	3	Robot moving
Xpos	Unsigned integer (15 ls-bits)	Wheel-encoder integrated coordinates; multiply by *DistConvFactor* to convert to millimeters.
Ypos	Unsigned integer (15 ls-bits)	
Thpos	Signed integer	Orientation in platform-dependent units— multiply by *AngleConvFactor* for degrees
L vel	Signed integer	Wheel velocities (respectively Left and Right); multiply by *VelConvFactor*—currently 1.0, — to convert into millimeters per second
R vel	Signed integer	
Battery	Byte	Battery charge in tenths of volts
Stall and Bumpers	Integer	Motor stall and bumper accessory indicators. Bit 0 of the lsbyte is the left wheel stall indicator =1 if stalled. Bits 1-5 of that same byte correspond to the bump switch states (1=on) for the rear bumpers accessory. Bit 0 of the msbyte is the right wheel stall; the bits 1-5 of that same msbyte correspond to the front bumpers switch states
Control	Signed integer	Set point of the server's angular position servo—multiply by *AngleConvFactor* for degrees
FLAGS	Signed integer	b0 – motors flag (1=motors enabled) b1 – sonar flag: enabled if 1
Compass	Byte	Compass heading in 2-degree units

Sonar readings	Byte	Number of new sonar readings included in information packet; readings follow:
Sonar number	Byte	Sonar number
Sonar range	Unsigned integer	Sonar range; multiply by *RangeConvFactor*—currently 0.268 —for millimeters
...rest of the sonar readings...		
Timer	Unsigned int	Selected analog port number 1-5
Analog	Byte	User Analog input (0-255=0-5 VDC) reading on selected port
Digin	Byte	User I/O digital input
Digout	Byte	User I/O digital output
Checksum	Integer	Checksum

7.2.2.4 Client Command Packet (CCP)

The Control Server, P2OS, has a structured command format for receiving and responding to the directions from the client for control and operation of the P2PB robot. The number of client commands per second depends on how frequent the user control the mobile robot. Commands are processed at a maximum rate of one per millisecond. The elements of the client command packet are shown in Table 7.4.

Table 7.4: Client command packet (CCP)

Component	**Bytes**	**Value**	**Description**
Header	2	0xFA,0xFB	Packet header; same for client and server
Byte Count	1	N+2	Number of following command bytes plus Checksum's two bytes, but not including Byte Count. Maximum of 200
Command Number	1	0-255	Client command number
Argument Type	1	0x3B or 0x1B or 0x2B	Required data type of command argument: positive integer, Negative integer or absolute value, or string
Argument	N	Data	Command argument; integer or string
Checksum	2	Computed	Packet integrity checksum

A P2OS command is comprised of a one-byte command number optionally followed by, if required by the command, a one-byte description of the argument type and the argument value. There are four types of P2OS client command arguments: none, unsigned integers two bytes long, signed integers two bytes long, and NULL-terminated strings consisting of as many as 196 characters. The byte order, where applicable, is least-significant byte first. Negative integers are transmitted as their absolute value (unlike information packets, which use sign extension for negative integers). The argument is an integer, a string, or nothing, depending on the command.

7.2.3 Server-Client Video Structure

The server-client video structure is shown in Figure 7.10.

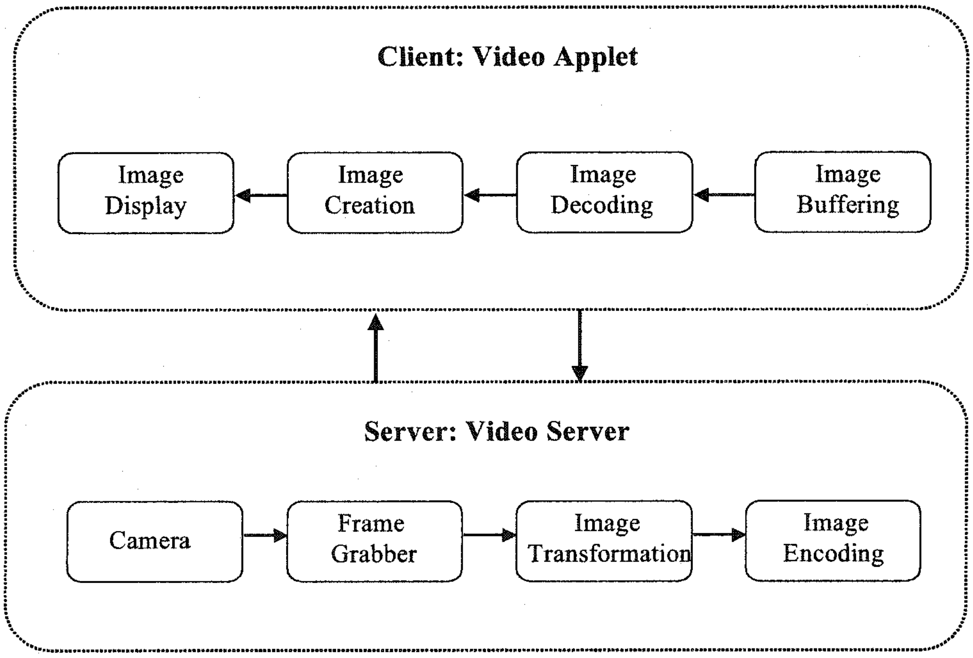


Figure 7.10: Server-client video structure

The live video stream of the mobile robot's environment is sent back to the human operator by the Video Server in the P2PB mobile robot. The sending rate is controlled by the trinomial protocol based on network states. The images are captured by a bt848 chipset-based frame grabber and then compressed before transferring to the client application via the Internet. On the client side, the Video Applet buffers, decodes and creates the image when it receives the entire frame and displays it.

For the data communications of teleoperation and e-service robotic systems, video transmission consumes most of the bandwidth and is very costly. Video data are usually compressed before transmission. Rather than employing conventional uniform-resolution compression methods, we adopt the wavelet-based image foveation technology to compress the video data further. The compression rate in the implementation is achieved as high as 30:1 with an acceptable resolution.

The idea of image foveation is to mimic human nonuniform-resolution vision system. Ordinary images are sampled, stored and transmitted with a uniform resolution, but human vision system does not require uniform detail in the field of vision. Less detail is needed on the periphery of the vision field than at the focal point. The concept of image foveation is to take advantage of this feature of human natural vision. A foveated image has non-uniform resolution. A spatially variant filter is applied to the image. This filter maintains high fidelity around the point of interest (For example, the ground floor in front of the robot for mobile robot tele-navigation and the edges of the object for fine tele-manipulation etc) while reduces spatial resolution away from the region of interest according to the sensitivity function of the human visual system. Thus, substantial savings in bandwidth are achieved compared to traditional uniform-resolution image compression algorithms.

Different from uniform-resolution images, foveated ones characterize themselves by implementing multi-resolution sub-regions in the same image. No matter what coding and transmission algorithm that may be employed, a function to describe the spatial sensitivity of the human visual system must be determined first. In [133], Schwartz presents a logarithmic resolution function, which would be the most appropriate from the biology point of view. However as argued in [134], this sensitivity function is difficult to implement due to singularities along one axis. Therefore, various simplifications are proposed in the literature, among which two are widely adopted. One is a pyramid image representation presented by Geisler and Perry in [135]. The other one is a wavelet-based approach introduced by Chang and Yap [136]. The advantage of the later approach over the former one is that no redundant information is added in order to build the pyramid. In the implementation of the platform, we employ the wavelet-based approach.

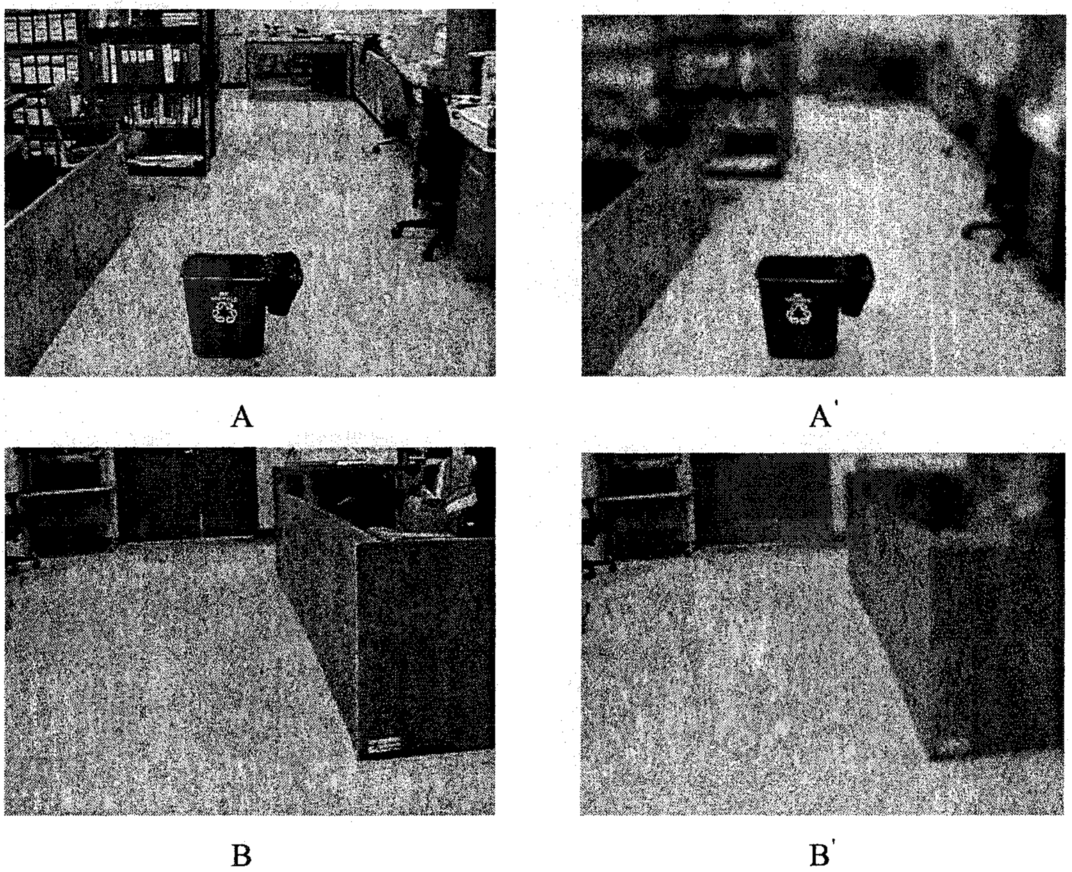
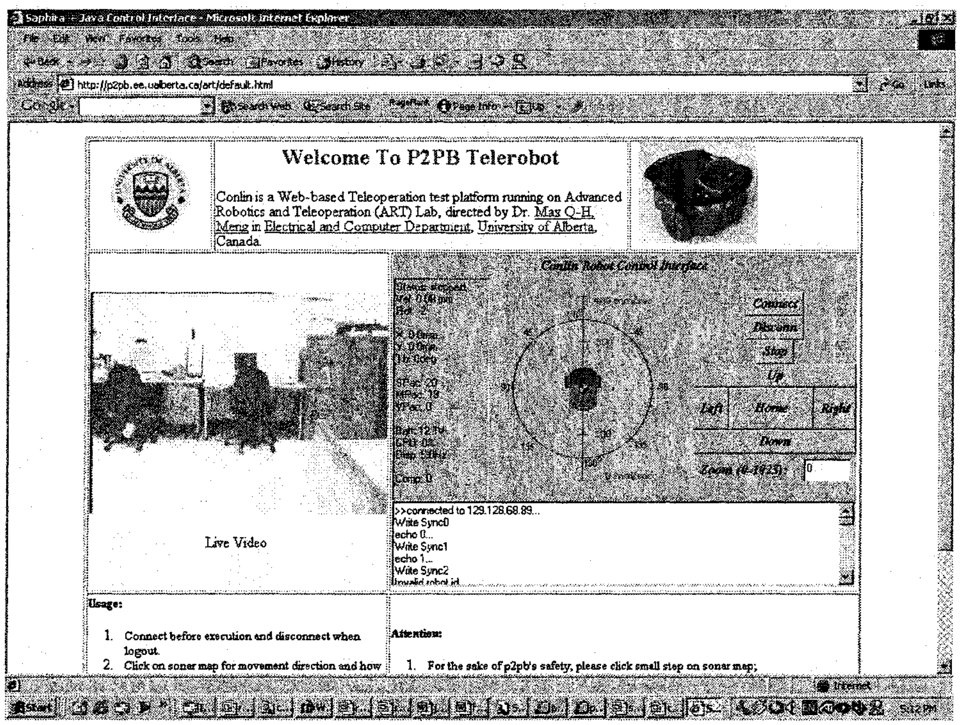


Figure 7.11: Illustration of image foveation

As an illustration example, Figure 11 shows the idea of image foveation. Figure 11-A and 11-B are the original uniform-resolution images, and Figure 11-A' and 11-B' are the foveated ones respectively. For the foveated image shown in Figure 11-A', the bottom part of the garbage bin is the point of interest and the resolution around this region is the highest. The resolution fades away from the fovea, the bottom part of the garbage bin. Similarly, for Figure 11-B', the left part of the image, the floor, is the fovea.

7.3 User Interface

The user interface (UI) is designed in orientation to human operators-- making it easy to interact with the remote mobile robot. The outlook of feedback data display, user manipulating platform, operation convenience and feasibility are taken into account when the UI is designed. A snapshot of the teleoperation UI is shown in Figure 7.12.



Saphira -- Java Control Interface - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://p2pb.ee.ualberta.ca/art/default.html
Go Links
Search Web Search Site PageRank Page Info Up
Conlin Robot Control Interface
Connect
Disconn
Stop
Up
Left
Home
Right
Down
Zoom (0-1923):
0
Live Video
Internet

Welcome To P2PB Telerobot

Conlin is a Web-based Teleoperation test platform running on Advanced Robotics and Teleoperation (ART) Lab, directed by Dr. Max Q-H. Meng in Electrical and Computer Department, University of Alberta, Canada.

```
>>connected to 129.128.68.89...
Write Sync0
echo 0...
Write Sync1
echo 1...
Write Sync2
Invalid robot id
```

Usage:

1. Connect before execution and disconnect when logout.
2. Click on soner map for movement direction and how

Attention:

1. For the sake of p2pb's safety, please click small step on sonar map;

Figure 7.12: Teleoperation user interface

The UI mainly consists of two Java applets: Control Applet and Video Applet. It works on any Java-enabled web browser. Brief instructions on how to use the interface to control the remote mobile robot are also provided at the bottom part of the UI.

7.3.1 Live Video Display

Live visual images are the principal source of information feedback on which the human operator relies to have knowledge of the environment of the remote robot. For a reliable teleoperation, real-time image feedback is usually mandatory although image data transmission on the Internet is very costly. A live video screen is placed on the left-centre of the interface, as shown in Figure 7.12. The human operator can expect a higher resolution display of his/her interested area by mouse clicking and dragging.

7.3.2 Control Panel

The control panel, as placed on the centre of the UI, is an immediate environmental map of the robot. Because of the "skewed right" characteristic and variation of Internet delays, continuous rate control is difficult. Consequently, we use the discrete-command control scheme and a computer mouse rather than a joystick is used as the command-signalling device. The user navigates the robot by clicking the mouse on the control panel to indicate the orientation and velocity of next robot movement. Because of low confidence and low reliability of sonar readings, the sonar readings are displayed in the robot's immediate environmental map only for reference. The user acquires depth information mainly by manipulating the pan-tilt angle of the robot's on-board camera.

7.3.3 Camera Control and Other Information Display

To the immediate left of the control panel is a text area for the display of the robot's other state information feedback, such as speed, position and battery voltage etc. At the right side of the control panel, there are some control buttons for camera control. By clicking the mouse on these buttons, the human operator can adjust the pan-tile angle of the video camera to have a best view of the robot's immediate environment.

7.4 Experiments

To demonstrate the validity and feasibility of the proposed platform for e-service mobile robotic systems described above, we have carried out many experiments. Figure 7.13 shows the setup of the environment that the P2PB mobile robot moves through. The goal of the experiments is to remotely guide the P2PB mobile robot from the starting point O_0 to the objective point O_F via the Internet.

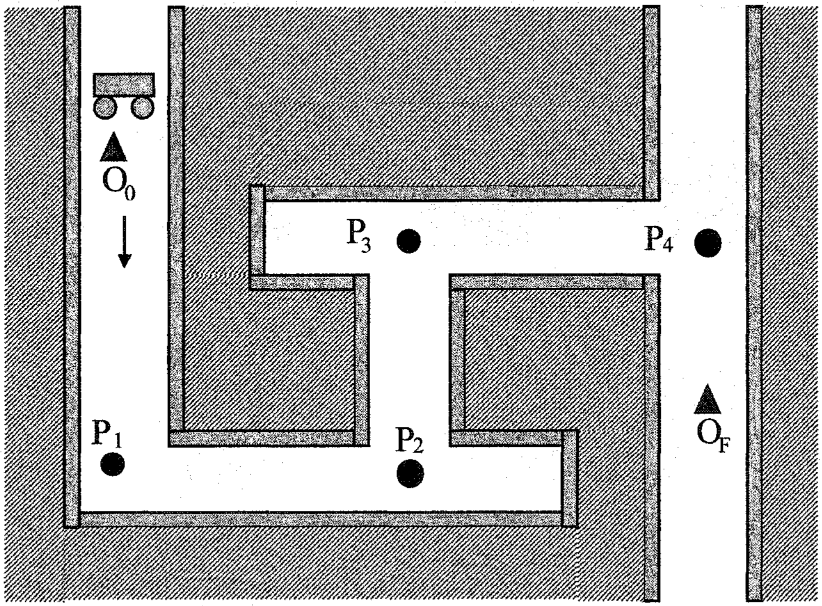
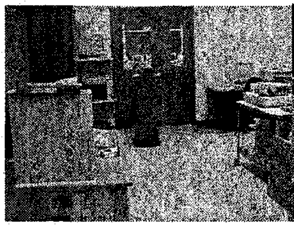


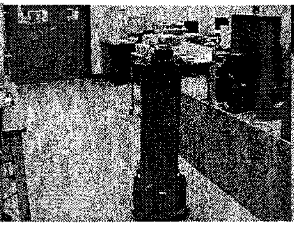
Figure 7.13: The objective of teleoperation is to navigate the mobile robot from point O_0 to O_F. The obstacles (walls) are represented by solid squares.

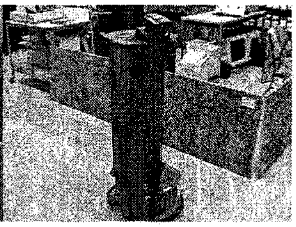
In the experiments, by using any Java-enabled web browser, such as Microsoft Internet Explorer or Netscape Communicator, on a regular PC, the users successfully navigated the P2PB mobile robot from point O_0 to point O_F via the Internet. Figure 7.14

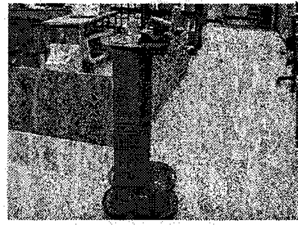
shows a sequence of the snapshots of the P2PB mobile robot when it was remotely guided via the Internet to move from point O_0 to point O_F in the ART (Advanced Robotics and Teleoperation) Lab at the University of Alberta, Canada.


1

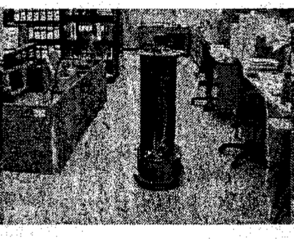
2

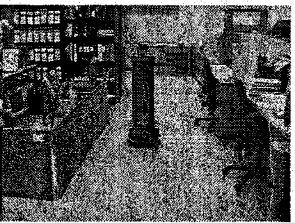

3

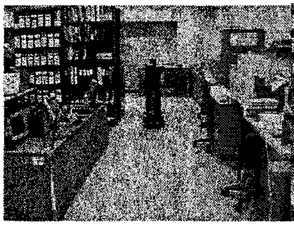

4


5

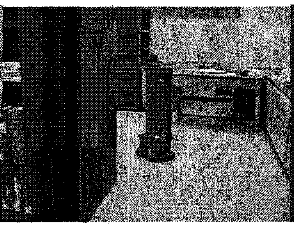
6

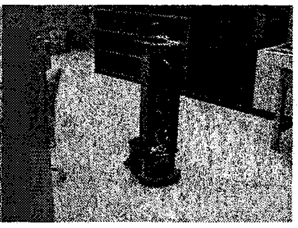

7

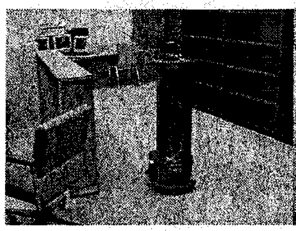

8

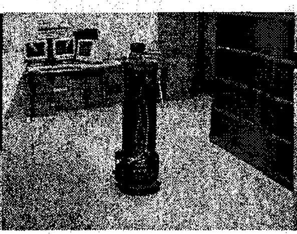

9

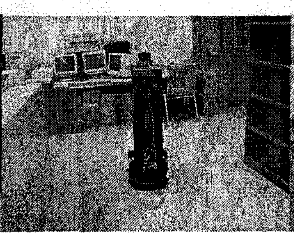
10

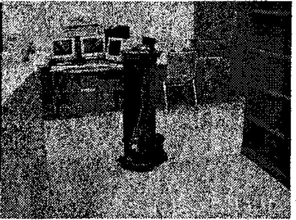

11


12


13


14


15


16

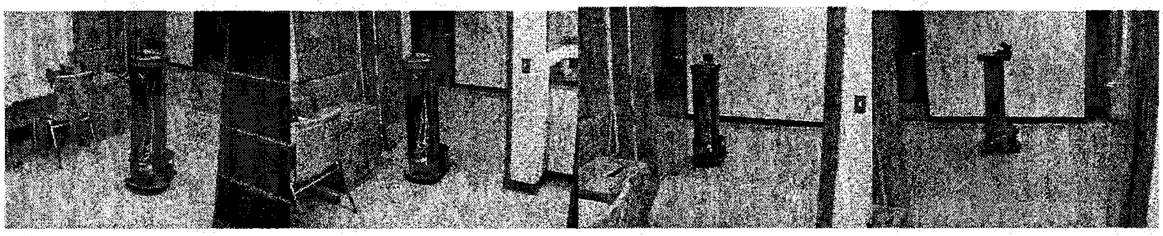


Figure 7.14: A sequence of robot images showing how the P2PB mobile robot is teleoperated over the Internet by the user to move through the ART Lab at the university of Alberta

We have conducted the experiments in different day times -- morning, noon, afternoon and night -- trying to capture the "rush hour" of the Internet traffic. Every time, the user succeeded to navigate the mobile robot through the ART Lab.

7.5 Summary

As the first step towards a real-world network-enabled robotic system, a new modular platform for e-service mobile robotic systems is developed in this chapter. The system enables Internet users to guide a P2PB mobile robot remotely via the Internet by using any Java-enabled web browser.

The system hardware configuration mainly consists of a commercial Pioneer 2 PeopleBot (P2PB) mobile robot. On the head of the robot, a Sony PTZ video camera provides live visual information feedback on robot's immediate environment. There are two sonar arrays for range measurement. Each array contains eight sonars. The robot also has some lower bumper switches for collision avoidance. In addition to visual and range information, the mobile robot provides position and speed information feedback. The

P2PB mobile robot is connected to the Internet through a pair of wireless LAN adaptors. Human operators can control the mobile robot with a web browser on an ordinary PC.

The system employs a client-server software architecture for robot control and feedback information display. In the model, there are two servers, Video Server and Control Server, and the two corresponding clients, Video Applet and Control Applet. The web browser is any Java-enabled web browser and the web server is a Linux Apache web server:

- On the client side -- the control station, the Video Applet is responsible for live image decoding and display, and the Control Applet is for mobile robot control, camera control and display of other sensor information feedback such as robot position and speed.
- On the server side -- the P2PB mobile robot, the Video Server is in charge of video grabbing, compression, encoding and transmission, while camera pan-tile-zoom control, motor control and sensor information sampling etc. are taken care of by the Control Server.
- The corresponding server and client exchange information through the Linux Apache web server.

The great benefit of this client-server architecture is that the client software is insulated from the lowest level details of the mobile robot. Thus, it is very easy to implement and test new advanced teleoperation control algorithms, interface designs and applications on this platform without large programming work.

The teleoperation user interface is designed with the intension of making it easy for users to interact with the remote mobile robot. Live scene video images are displayed on the left-center of the screen and the control panel is put on the right-center of the screen for mobile robot control, camera control and other sensor feedback display. The user controls the mobile robot easily by clicking the mouse in the control panel. Mouse clicking events are intercepted and interpreted by the Control Applet as the indication of the orientation and velocity of next robot movement. The user can also control the pan-tilt angles of the camera to have a best view of the remote environment by clicking the corresponding control buttons. Brief instructions on how to use the interface to control the remote mobile robot are provided by the UI.

Direct sonar readings are presented on the robot's immediate environmental map for reference. Because of low confidence and low reliability of sonar readings, the user mainly relies on the live scene video images to have knowledge of the robot's environment.

To achieve larger savings of network bandwidth, the wavelet-based image foveation technology is adopted to compress video data before transmission.

Experiments have been carried out to test the feasibility of the platform. In the experiments, by using any Java-enabled web browser such as Microsoft Internet Explorer or Netscape Communicator, on a regular PC, the users successfully guided the P2PB mobile robot through a laboratory environment remotely over the Internet.

In summary, the preliminary experimental results of the platform look promising. Future research on remote compensation control algorithms, teleoperation interface designs and applications towards a real-life e-service robotic system can be easily tested and implemented on this platform.

Chapter 8

Conclusions and Future Work

In this chapter, we summarize the research presented in this thesis and then outline the future work related to our study.

8.1 Conclusion

In this thesis, we describe a novel rate-based transport protocol named the trinomial (α, β, γ) for Internet-based teleoperation and e-service robotic systems. Based on the protocol, a new modular platform for e-service mobile robotic systems is proposed, developed and implemented.

First, we describe and justify the reasons for new network transport protocols for Internet-based teleoperation and e-service robotic systems for which the most challenging and distinct difficulties are transmission delays, delay jitter and bandwidth limitation. The core of the network is relatively fixed. Most of current routers in the Internet still implement the FIFO scheduling and DropTail queuing. Today's Internet does not support any source reservation mechanisms or Quality of Service (QoS). Consequently, to deal with transmission delays, delay jitter and bandwidth limitation in data transmission, the most feasible approach is to take them into account in transport protocols. Currently, most researches employ one of two current transport protocols, i.e., TCP and UDP. However, neither TCP nor UDP is appropriate for Internet-based teleoperation and e-service robotic systems, which is demonstrated by our simulations. TCP induces large average delays and great delay jitter. Also, the transmission rate of TCP fluctuates severely. On the other hand, UDP is unresponsive and does not behave properly since it has no mechanisms to detect network states, such as congestion and extra bandwidth availability. As a result, UDP may be inefficient, impair network stability and give rise to unfairness problems among competing flows. Consequently, we argue that new transport protocols are needed if we allow today's Internet to support teleoperation and e-service robotic systems.

Second, we introduce a novel rate-based transport protocol that is specifically developed for teleoperation and e-service robotic systems. We first limit the design space based on the requirements and constraints that are imposed by the application and the network. Then we introduce the trinomial protocol for which there are three parameters (α, β, γ) that are adjustable based on specific applications. By a proper combination of the values of α, β and γ, the trinomial protocol meets with the requirements of teleoperation and e-service robotic systems and the constraints on network transport protocols. Since there are no retransmission mechanisms, delays and delay jitter are minimized. In the steady state, the transmission rate is smooth. When available network bandwidth increases/decreases, the trinomial protocol tracks these variations quickly. The protocol is responsive since it reduces its transmission rate when the network is congested. A trinomial flow achieves a throughput that is reasonably close to that of a TCP flow traveling over the same network path. Two trinomial applications using the same route reach the same sending rate no matter how different their initial sending rates may be. The trinomial protocol always converges to the efficient state regardless of initial states. We extensively examine the performances of the protocol and compare with both TCP and UDP in a wide range of network parameters through simulations. For example, to assess TCP-compatibility, we examine the interaction of the trinomial protocol with different TCP implementations, such as Tahoe, Reno, NewReno, Sack, Vegas and Fack. For all the evaluations and comparisons, we have conducted simulations for both the RED and DropTail queues. The simulation results indicate that: the trinomial protocol has similar minimized delays and delay jitter to UDP; its loss rate is unexpectedly better than that of TCP; in the steady state, its transmission rate is much smoother than that of

TCP while its transient response to network bandwidth changes is very quick; it is responsive, inter-protocol fair (TCP-compatibility), intra-protocol fair and efficient.

Third, we investigate Internet delays by using both linear and nonlinear statistical approaches and then introduce a novel adaptive algorithm called the MEP scheme for RTT and RTO estimations, which are an integral component of the implementation of the trinomial protocol. The experimental statistical results show that RTT time series is not random, but linearly correlated between adjacent and near-adjacent observations. It is also revealed that there are only weak or no nonlinear correlations among RTT observations. Based on these findings, we conclude that, to estimate next RTT value from past and current observations, RTT time series should be characterized by linear models. Due to the multi-structure characteristic of RTT dynamics, the traditional parameter-fixed ARMA model is not capable of tracking RTT dynamics constantly. Thus, we introduce a novel parameter-varying adaptive algorithm for RTT estimation based on the information theory and maximum entropy principle. Since the coefficients of the model are updated every ACK received, the model is adaptive and is able to catch up with RTT dynamics quickly. Since the solution to the coefficient is only a polynomial, the computing overhead of updating the coefficients is much small compared to other adaptive algorithms. The experimental results show that the MEP algorithm outperforms the ARMA method in terms of both RTT and RTO estimations

Fourth, as the first step towards a real-world network-enabled robotic application system, a new modular platform for e-service mobile robotic systems is developed based on the trinomial protocol. The system enables Internet users to guide a mobile robot through an office/laboratory environment remotely via the Internet by using any Java-enabled web browser. The system hardware architecture mainly consists of a commercial Pioneer 2 PeopleBot mobile robot including a Sony PTZ video camera, two sonar arrays, lower bumper switches, and speed and distance encoders. The mobile robot is connected to the Internet by using a pair of wireless LAN adaptors. The system software employs a client-server architecture for robot control, camera control and feedback information display. In this architecture, there are two servers, Video Server and Control Server, and two corresponding clients, Video Applet and Control Applet. The web browser is any Java-enabled web browser and the web server is a Linux Apache web server. The great benefit of this client-server architecture is that the client application software is insulated from the lowest level details of the mobile robot. As a result, it is very easy to implement and test new advanced teleoperation control algorithms, interface designs and applications on this platform without large programming work. To achieve a large saving of network bandwidth, the wavelet-based image foveation technology is adopted to compress video data before transmission. A compression rate as high as 30: 1 is achieved. The control user interface is designed with the intension to make it easy for users to interact with the remote mobile robot.

Experiments have been carried out to examine the validity and feasibility of the platform. In the experiments, by using any Java-enabled web browser on an ordinary PC, the users successfully guided the P2PB mobile robot through a laboratory environment over the Internet. The preliminary experimental results look promising.

8.2 Future Work

The work presented in this thesis can only be considered preliminary, since many challenging and possibly more important real-world problems have not been touched in this thesis. Towards our long-term research goal -- a real-world network-enabled robotic application system, in this section, we propose some problems as possible future work.

- *Real-world experiments*: The performance evaluation on the trinomial protocol is mainly carried out through simulations. The platform developed for e-service mobile robotic systems is only tried on the local Internet that is within the campus of the University of Alberta. For real-life applications, both the protocol and the platform should be extensively tested on the real Wide-Area Network with great diversity, irregularity and heterogeneity. For example, most of the simulations in the thesis assume long-lived TCP flows (FTP traffic). However, a reasonable portion of today's Internet traffic consists of short-lived TCP connections generated by web-based applications. TCP-compatibility should be examined in the presence of the more realistic short-lived web-based background traffic. The real-world experiments would validate our simulation results and help us to identify some of the actual issues that exist in the Internet and cannot be easily captured in simulation environments.

- *Differentiating congestion losses from wireless losses*: Like most other transport protocols, the trinomial protocol relies on packet losses to determine whether the network is congested. If packets get lost, the network is considered congested. For wired network connections, this inference mechanism holds very well. However, for wireless connections, this inference does *not* work very well because packet losses in wireless links do not necessarily mean congestion. Current proposals to differentiate congestion losses from wireless losses require cooperation from the intermediate nodes on the path, which is infeasible in most cases [178]. The desirable approach should be an end-to-end mechanism that does not require intermediate hosts to take any specific actions. From the literature, this problem has not been well solved [175,176] and it is one topic of my future research.
- *Teleoperation compensation control algorithms and interfacing*: In this thesis, we do not consider control algorithms and teleoperation interfaces deliberately. However, for the end of a real-world network-enabled robotic application system, time delays, delay jitter and bandwidth limitation problems should be incorporated into all aspects, such as compensation control algorithms, interface designs, and data transmission and presentation. We have developed an adaptive scaling control algorithm and the concept of SoftHaptics for interfacing. However neither of them has been implemented on current platform. In the future, we will try to apply them to the platform.

- *Real-world applications*: We plan to apply the e-service mobile robotic system proposed in the thesis to real-world applications, for example, for home healthcare purposes. Nowadays, population aging is an urgent societal problem around the world. Inexpensive assistive devices or systems are becoming demanding, which allow healthcare centers or the relatives to remotely monitor and help senior people who stay at home via the Internet. These applications are very promising and possible as the Internet technologies continue to advance.

Bibliography

[1] B. Giordano and R. Buckley, “Trends in colour imaging on the Internet,” in *the Proceedings of 9th International Congress of the AIC*, SPIE Vol. 4421-4422, June 2001, Rochester, USA

[2] K. Taylor and J. Trevelyan, “Australia’s telerobot on the Web”, in *the 26th International Symposium on Industrial Robots,* Singapore, October 1995

[3] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter and J. Wiegley, “Desktop teleoperation via the World Wide Web,” in *IEEE International Conference on Robotics and Automation*, Nagoya, Japan, 1995

[4] R. Simmons, R. Goodwin, R. Haigh, S. Koenig and J. O’Sullivan, “A modular architecture for office delivery robots,” in *Autonomous Agents 1997*. February 1997, p245-252

[5] W. Burgard, A Cremers, D. Fox, D. Haehnel, G. Lakemeyer, D. Schulz, W. Steiner and S. Thrun, “Experiences with an interactive museum tour-guide robot”, in *Artificial-Intelligence,* v114, n1, 1999, p3-55

[6] E. Paulos, and J. Canny, “Ubiquitous tele-embodiment: applications and implications,” in *International Journal of Human Computer Studies*, n 46, 1997, p861-877

[7] O. Michel, P. Saucy and F. Mondada, “ ‘KhepOnTheWeb’: An experimental demonstrator in telerobotics and virtual reality,” in *Proceedings of the Annual International Conference on Virtual Systems and Multimedia, VSMM*, 1997, IEEE, Los Alamitos, CA, USA, p 90-98

[8] K. Han, S. Kim, Y. Kim and J. Kim, "Internet Control Architecture for Internet-Based Personal Robot," in *Autonomous Robots*, 10, 2001, pp135-147

[9] H. Hu, L. Yu, P. Tsui and Q Zhou, "Internet-Based Robotic Systems for Teleoperation," in *International Journal of Assembly Automation*, Vol. 21, No. 2

[10] R. Luo and T. Chen, "Development of a Multibehavior-Based Mobile Robot for Remote Supervisory Control through the Internet, " in *IEEE/ASME Transactions on Mechatronics*, vol5, No.4, Dec. 2000

[11] R. Oboe and P. Fiorini, "A design and control environment for Internet-based telerobotics," in *International Journal of Robotics Research*, vol. 17, no. 4, April 1998, pp.433-449

[12] R. C. Goertz and R. C. Thompson, "Electronically controlled manipulator" in *Nucleonics*, 1954, pp46-47

[13] W. R. Ferrell, "Remote manipulation with transmission delay", in *IEEE Transaction Human Factors in Electronics HFE-6*, No.1, 1965

[14] B. Hannaford, L. Wood, D. McAffee and H. Zak, "Performance evaluation of a six-axis generalized force-reflecting teleoperator," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no.3 1991, pp620-633

[15] S. T. Venkataraman and S. Hayati, "Shared/Traded control of telerobots under time delay," in *Computers Elect. Engineering*, vol. 19, no. 6, 1993, pp481-494

[16] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*, The MIT Press, Cambridge, Massachusetts, USA, London, England, 1992

[17] D. Schulz, W. Burgard and A. B. Cremers, "Robust visualization of navigation experiments with mobile robots over the Internet," in *Proceedings of the 1999*

IEEE/RSJ International Conference on Intelligent Robots and Systems, 1999, pp942-947

[18] J.Tan, I.Belousov and G.Clapworthy, “A Virtual Environment Based User Interface for Teleoperation of a Robot Using the Internet,” in *Proc. of the Sixth UK VR-SIG Conference*, Salford, U.K., Sept. 13-15, 1999, pp.145-154.

[19] J. Postel, *RFC 793: Transport Control Protocol, DARPA Internet Program Protocol Specification*, September 1981

[20] J. Postel, *RFC 768: User Datagram Protocol*, August 28, 1980

[21] S. Floyd, *RFC 2914: Congestion Control Principles*, Best Current Practice, September 2000

[22] R. Rejaie, *An End-to-End Architecture for Quality Adaptive Streaming Applications in the Internet,* Ph.D. Dissertation, University of Southern California, 1999

[23] I. J. Wakeman, *Congestion Control for Packetised Video in the Internet*, Ph.D. Dissertation, University of London, 1995

[24] A.S. Tanenbaum, *Computer Networks*, Prentice Hall, Englewood Cliffs, 1981

[25] D.W. Davies, “The Control of Congestion in Packet Switching Networks,” in *IEEE Transactions on Communications*, Vol. 20, June 1972, pp546-550

[26] V. Jacobson, “Congestion avoidance and control,” in *ACM SIGCOMM*, pages 314-329, August 1988

[27] D.-M. Chiu and R. Jain, “Analysis of the increase and decrease algorithms for congestion avoidance in computer networks,” in *Computer Networks and ISDN Systems,* No.17, 1989, pp1-14

[28] K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with Connectionless Network Layer," in *Proc. SIGCOMM'88*, August 1988, pp. 303-313

[29] D. Comer, *Internetworking with TCP/IP Principles, Protocols and Architecture*, Prentice Hall, Englewood Cliffs, 1988

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

[31] R. Jain, K. Ramakrishnan, D. Chiu, "Congestion Avoidance in Computer Networks with a Connectionless Network Layer," in *DEC-TR-506*, reprinted in C. Partridge, Ed., *Innovations in Internetworking*, published by Artech House, October 1988

[32] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queuing algorithm," in *Proceedings SIGCOMM Symposium on Communications Architectures and Protocols*, September 1989,pp.1-12

[33] S. Floyd, "Connections with multiple congested gateways in packet-switched networks," in *Computer Communication Review*, October 1991, 21(5): 30-47

[34] L. Zhang, S. Shenker, and D. Clark, "Observations on the dynamics of a congestion control algorithm: The effect of two-way traffic," in *Proceedings of the ACM SIGCOMM*, September 1991

[35] T. V. Lakshman and U. Madhow, "Performance analysis of window-based flow control using TCP/IP: the effect of high bandwidth delay products and random losses," in *IFIP Transactions C26, High Performance Networking V*, North Holland, 1993

[36] K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks," in *ACM Transactions on Computer Systems*, 8(2):158-181, May 1990

[37] M. Mathis and J. Mahdavi, "Forward acknowledgment: Refining TCP congestion control," in *Proceedings of the ACM SIGCOMM*, August 1996

[38] L. S. Brakmo and L. L. Peterson, " TCP Vegas: End to end congestion avoidance on a global Internet," in *IEEE Journal of Selected Areas in Communication*, 13(8):1465-1480, October 1995

[39] J. S. Ahn, P. B. Danzig, Z. Liu, and L. Yan, "Experiences with TCP Vegas: Emulation and experiment," in *Proceedings of the ACM SIGCOMM*, Cambridge, MA., August 1995

[40] J. Mahdavi and S. Floyd, "TCP-friendly unicast rate-based flow control," in *Technical note sent to the end2endinterest mailing list*, January 1997

[41] A. Legout, and E. W. Biersack, "Beyond TCP-friendliness: A new paradigm for end-to-end congestion control," in *Eurecom Technical Report*, September 2000

[42] B. Braden *et al.*, *RFC2309: Recommendations on Queue Management and Congestion Avoidance in the Internet*, April 1998

[43] S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance," in *IEEE/ACM Transactions on Networking*, Vol.1, No 4, August 1993, pp. 397-413

[44] S. Floyd and V. Jacobson, "On traffic phase effect in packet-switched gateways," in *Internet working: Research and Experiences*, 3(3): 115--156, September 1992

[45] A. Demers, S. Keshav, and S. Shenker, “Analysis and simulation of a fair queuing algorithm,” in *Proceedings SIGCOMM Symposium on Communications Architectures and Protocols*, September 1989, pp.1-12

[46] C. Leffehocz, B. Lyles, S. Shenker, and L. Zhang, “Congestion control for best-effort service: why we need a new paradigm,” in *IEEE Network*, 10(1), January 1996

[47] I. Stoica, S. Shenker, and H. Zhang, “Core-stateless fair queuing: Achieving approximately fair bandwidth allocations in high speed networks,” in *Proceedings of the ACM SIGCOMM*, September 1998, pp118-130

[48] I. Stoica and H. Zhang, “ Providing guaranteed services without per flow management,” in *Proceedings of the ACM SIGCOMM*, September 1999

[49] R. Rejaie, M. Handley, and D. Estrin, “RAP: An end-to-end rate-based congestion control mechanism for real-time streams in the Internet,” in *Proc. IEEE Infocom*, March 1999

[50] J. Nagle, RFC970:*On Packet Switches with Infinite Storage*, Dec. 1985

[51] D. Waitzman, *RFC 2549: IP over Avian Carriers with Quality of Service*, April 1999

[52] K. Ramakrishan and Raj Jain, “ A binary feedback scheme for congestion avoidance in computer networks,” in *ACM Transactions on Computer Systems*, 8(2): 158--181, 1990

[53] J. Padhye, J. Kurose, D. Towsley and R. Koodli, "A TCP-friendly rate adjustment protocol for continuous media flows over best effort networks", in *UMass-CMPSCI Technical Report TR 98-04*, October 1998

[54] K. Nichols, *et al.*, *RFC 2474: Differentiated Services Field*, December 1998

[55] A. Mankin, "Random drop congestion control," in *Proceedings of the ACM SIGCOMM*, 1990

[56] D. Clark, M. L. Lambert, and L. Zhang, "NETBLT: A high throughput transport protocol," in *Proceedings of the ACM SIGCOMM*, August 1988

[57] R.. Jain, "A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks," *ACM Computer Communication Review*, 19(5): 56--71, October 1989

[58] S. Floyd, M. Handley, and J. Padhye, "A comparison of equation-based and AIMD congestion control," http://www.aciri.org/floyd/papers.html, May 12, 2000

[59] D. Clark, "The Design Philosophy of the DARPA Internet Protocols," in *Proc. ACM SIGCOMM'88*, pp106-114, Aug 1988

[60] K. Ramakrishnan and S. Floyd, *RFC 2481: A proposal to add explicit congestion notification (ECN) to IP*, January 1999

[61] V. Jacobson *et al., RFC1323: TCP Extensions for High Performance*, May 1992

[62] V. Jacobson *et al.*, *RFC2525: Know TCP Implementation Problems,* March 1999

[63] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," in *DEC Research Report TR-301*, September 1984

[64] J. W. Wong, F. P. Sauve, and J. A. Field, "A study of fairness in packet switching networks," in *IEEE Transactions on Communications*, COM-30, 2, February 1982, pp346-353.

[65] S. Keshav, *Congestion Control in Computer Networks*, Ph.D. Thesis, EECS, University of Berkeley, CA 94720,USA, September 1991

[66] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, 1987

[67] D. Clark, *RFC 817: Modularity and Efficiency in Protocol Implementation*, July 1982

[68] S. Jacobs and A. Eleftheriadis, “Providing video services over networks without quality of service guarantees,” in *Proceedings, World Wide Web Consortium Workshop on Real-time Multimedia and the Web*, Sophia Antipolis, October 1996

[69] N. Wakamiya, M. Murata and H. Miyahara “TCP-friendly video transfer” in *IEEE International Conference on Multimedia and Expo (II)*, 2000

[70] J-C. Bolot and T. Turletti, “Experience with rate control mechanisms for packet video in the Internet,” in *Computer Communication Review*, January 1998

[71] D. Sisalem and H. Schulzrinne, "The loss-delay adjustment algorithm: A TCP-friendly adaptation scheme," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, (Cambridge, England), pp. 215--226, Jul. 1998.

[72] Z. Chen, S. M. Tan, R. H. Campbell, and Y. Li, “Real time video and audio in the world wide web,” in *World Wide Web Journal*, Vol 1, 1996

[73] J. Padhye, J. Kurose, D. Towsley and R. Koodli, “A model based TCP-friendly rate control protocol,” in *Proceedings of NOOSDAV’99*, 1999

[74] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, “Modeling TCP throughput: A simple model and its empirical validation,” in *Proceedings of SIGCOM’98*, 1998

[75] M. Handley, J. Pahdye, S. Floyd and J. Widmer, "TCP friendly rate control (TFRC): Protocol specification," in *Internet draft draft-ietf-tsvwg-tfrc-03.txt*, work in progress, May 2001.

[76] S. Floyd, M. Handley, J. Padhye and J. Widmer, "Equation-based congestion control for unicast applications," in *SIGCOMM 2000*, August 2000

[77] W. Tan and A. Zakhor, "Real-time Internet video using error-resilient scalable compression and TCP-friendly transport protocol," in *IEEE Trans. Multimedia*, June 1999

[78] D. Bansal and H. Balakrishnan, "Binomial congestion control algorithms," in *IEEE INFOCOM 2001 proceedings*, vol.2, pp. 631-640, Anchorage, Alaska, April 22-24, 2001.

[79] P.Vasallo, "Variable packet size equation based congestion control," ftp.icsi.berkeley.edu, May 2000

[80] J. Padhye, J. Kurose, D. Towsley and R. Koodli, "A TCP-friendly rate adjustment protocol for continuous media flows over best effort networks," in *UMass-CMPSCI Technical Report TR 98-04*, October 1998

[81] R. Rejaie, M. Handley and D. Estrin, "Architectural considerations for playback of quality adaptive video over the Internet," in *Technical report 98-686*, CS-USC, November 1998

[82] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," in *Computer Communication Review*, 26(3), July 1996

[83] W. Stevens. *RFC 2001: CP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*, Jan 1997

[84] S. Floyd. “Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic,” in *ACM Computer Communication Review*, 21(5) pp30–47, Oct. 1991

[85] S. Shenker, “Fundamental design issues for the future Internet,” in *IEEE Journal of Selected Areas in Communication*, 13(7): 1176--1188, 1995

[86] M. Gilge and R. Gusella, “Motion video coding for packet-switching networks an integrated approach,” in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Boston, MA, November 1991

[87] H. Kanakia, P. P. Mishra, and A. Reibman, “An adaptive congestion control scheme fro real-time packet video transport,” in *Proceedings of the ACM SIGCOMM*, San Francisco, CA., September 1993, pp. 20--31

[88] D. LeGall, “MPEG: A video compression standard for multimedia applications,” in *Communications of the ACM*, 4(34), April 1991, pp47--58

[89] T. Turletti and C. Huitema, “Videoconferencing in the internet,” in *ACM/IEEE Transactions on Networking*, June 1996, pp. 340--351

[90] J. Bolot and T. Turletti, “A rate control mechanism for packet video in the Internet,” in *Proceedings of the IEEE INFOCOM*, June 1994, pp. 1216--1223

[91] *Recommendation H. 261: Video codec for audiovisual services at p×64 kb/s*, International Telecommunication Union (ITU-T), 1993

[92] K. Jeffy, D. L. Stone, T. Talley and F. D. Smith, “Adaptive, best-effort delivery of digital audio and video across packet-switched networks,” in *Workshop on Network and Operating System Support for Digital Audio and Video*, San Diego, CA., November 1992

[93] Z. Chen, SM Tan, R. H. Campbell and Y. Li, "Real time video and audio in the world wide web," in *Fourth International World Wide Web Conference*, December 1995

[94] S. Cen, C. Pu, and J. Walpole, "Flow and congestion control for Internet streaming applications," in *Proceedings Multimedia Computing and Networking*, January 1998

[95] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-friendly SIMD Congestion Control and Its Convergence Behavior," in *Proceedings of IEEE ICNP 2001*

[96] S. Li and C. Hwang, " Queue response to input correlation functions Continuous Spectral Analysis," in *IEEE/ACM Transactions on Networking*, 1(6), December, 1993, pp678-692

[97] A. Erramilli, O. Narayan and W. Willinger, "Experimental queuing analysis with long-range dependent packet traffic," in *IEEE/ACM Transactions on Networking*, 4(2): 209-223, April 1996

[98] W. Leland, M. Taqqu, and D. Wilson, " On the self-similar nature of Ethernet traffic (extended version), in *IEEE/ACM Transactions on Networking*, 2(1): 1:15, February, 1994

[99] A. Shankar *et al.*, "Performance comparison of routing protocols using MaRS: Distance vector versus link state," in *Proc. ACM Sigmetrics and Performance*, 1992, June 1992, pp181-1912

[100] L. Zhang and D. Clark, "Oscillating behavior of network traffic: A case study simulation," in *Internetworking: Research and Experience*, vol.1, no.2, pp101-112, Dec. 1990

[101] L. Zhang, S. Shenker and D. Clark, "Observations on the dynamics of a congestion control algorithm: the effects of 2-way traffic," in *Proc. ACM Sigcomm'91*, pp133-147, Sept 1991

[102] D. Mills, *RFC 889: Internet delay experiments*, RFC 889, December 1983

[103] P. Karn and C. Partridge, "Improving roundtrip time estimates in reliable transport protocols," in *ACM Transactions on Computer Systems*, vol.9, pp.364-373, Nov. 1991

[104] J. Bolot, "Characterizing Eng-to-end Packet Delay and Loss in the Internet," in *Journal of high-Speed Network*, vol.2, no.3, pp.305-323, December 1993

[105] G. Jenkins and D. Watts, *Spectral Analysis and Its Applications*, Holden-Day, 1968

[106] P. Bloomfield, *Fourier Analysis of Time Series*, John Wiley and Sons, 1976

[107] G. Box, and G. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, 1976

[108] Q. Li and D.L. Mills, "Jitter-based delay-boundary prediction of wide-area networks," in *IEEE-ACM-Transactions-on-Networking*, v 9 n 5 October 2001, p578-590

[109] M. S. Borella *et al.* "Self-similarity of Internet packet delay," in *Proceedings of IEEE ICC'97*, August, 1997

[110] P. Klan and G. Darbellay, "An Information theoretic adaptive method for time series forecasting," in *Neural Network World*, no. 2, 1997, pp227-238

[111] D. Kugiumtzis and M. A. Boudourides, "Chaotic Analysis of Internet Ping Data" in *SOEIS Meeting at Bielefeld*, March 27-28, 1998

[112] S. Guiasu and A. Shenitzer, "Maximum entropy principle," in *The Mathematical Intelligencer*, Vol. 7, No. 1, pp.42-48.

[113] E.T. Jaynes, "Information theory and statistical mechanics I," in *Physical Review* 106, pp620-630

[114] J. Uffink, "Can the Maximum Entropy Principle be explained as a Consistency Requirement?," in *History and Philosophy of Modern Physics* 26B (1995) 223-261.

[115] T. Pilioura and A. Tsalgatidou, "E-Services: Current Technology and Open Issues," in *Technologies for E-Services, Second International Workshop*, TES 2001, Rome, Italy, pp.1-15, September 14-15, 2001

[116] L. Delgrossi and L. Berger, *RFC 1819: Internet Stream Protocol Version2 (ST2), Protocol Specification- Version ST2+*, August 1995

[117] J. Saltzer, D. Reed, and D. Clark, "End-to-End Arguments in System Design," in *ACM Transactions on Computer Systems*, 2(4): 277-288, November, 1984

[118] S. Blake, *RFC2475: An Architecture for Differentiated Services*, December 1998

[119] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, "RSVP: A New Resource ReSerVation Protocol," in *IEEE Network*, 8(5): 8-20, 1993

[120] D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms," in *ACM SIGCOMM Conference*, August 1992

[121] V. Jacobson, K. Nicols, and K. Poduri, *An expedited forwarding PHB*, IETF Internet-Draft, February 1999

[122] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, *Assured forwarding PHB group*, IETF Internet-Draft, January 1999

[123] K. Ramakrishnan, Sally Floyd, and D. Black, *The Addition of Explicit Congestion Notification (ECN) to IP*. IETF Internet-Draft, January 2001

[124] P. Fiorini and R. Oboe, "Internet-based telerobotics: Problems and approaches," http://citeseer.nj.nec.com/148450.html, 1997

[125] A. Lasso and T. Urbancek, "Communication architectures for web-based telerobotic systems," in *IEEE Mediterranean Conference on Control and Automation, Dubrovnik*, Croatia, June 27-29, 2001

[126] L. Kleinrock, *Queuing Systems. Volume 2: Computer applications*, Wiley-Interscience, New York, 1976

[127] K. Thompson, G.J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," in *IEEE/ACM Transactions on Networking*, pp.10--23, November 1997

[128] M. Noyes and T. B. Sheridan, " A Novel Predictor for Telemanipulation through a Time Delay," in *Proceedings of International Conference on Systems, Man and Cybernetics*; 1984

[129] L. Conway, R. Volz And M. Walker, "Tele-Autonomous Systems: Methods and Architectures for intermingling Autonomous and Telerobotic Technology," in *Proceedings of International Conference on Robotics and Automation*, Volume 3; 1121-1130; 1987

[130] A. Bejczy, S. Venema, and W. Kim; "Role of Computer Graphics in Space Telerobotics: Preview and Predictive Displays," *in SPIE Volume 1387 Cooperative Intelligent Robotics in Space*, 365-377; 1990

[131] G. Hirzinger, J. Heindl, and K. Landzettel, "Predictive and Knowledge-Based Telerobotic Control Concepts," in *IEEE International Conference on Robotics and Automation*, May 14-19, Scottsdale AZ; 1768-1777; 1989

[132] http://www.activrobots.com/

[133] E. L. Schwartz, "Computational anatomy and functional architecture of striate cortex: A spatial mapping approach to perceptual coding," in *Vision Research*, vol.20, pp. 645-669, 1980

[134] B. Overall, *Foveated Imaging*, http://www-ise.stanford.edu/class/psych221/99/, Mar 1999

[135] W.S. Geisler and J. S. Perry, "A real-time foveated multi-resolution system for low-bandwidth video communication," in B. Rogowitz and T. Pappas (Eds.), *Human Vision and Electronic Imaging, SPIE Proceedings*, 3299, 294-305, 1998

[136] E. Chang and C. K. Yap, "A wavelet approach to foveating images," in *ACM Symposium on Computational Geometry*, vol. 13, pp. 397-399, 1997

[137] G. Jenkins and D. Watts, *Spectral Analysis and Its Applications*, Holden-Day, 1968

[138] P. Bloomfield, *Fourier Analysis of Time Series*, John Wiley and Sons, 1976

[139] C. Croarkin and Paul Tobias, "NIST/SEMATECH Engineering Statistics Internet Handbook," http://www.itl.nist.gov/div898/handbook/

[140] H. Stanley, L. Amaral, D. Canning, P. Gopikrishnan, Y. Lee and Y. Liu, "Econophysics: Can physicists contribute to the science of economics?" in *Physica A* 269 (1999) 156-169

[141] F. Busby, "Remotely Operated Vehicles," in *U. S. Dept. Commerce Rep.* , 03-78-603-0136

[142] R. S. Mosher, and B. Wendel, "Force reflecting electro-hydraulic servo-manipulator," in *Electro-Technology,* pp66-138

[143] W. R. Ferrell, and T. B. Sheridan, "Supervisory control of remote manipulation," in *IEEE Spectrum 4*, no. 10, October, 1967, pp81-88

[144] W. R. Corliss, and E. G. Johnsen, "Teleoperator Controls," in *NASA SP-5070*, Washington, DC: NASA Office of Technology Utilization, 1986

[145] E. G. Johnsen and W. R. Corliss, "*Teleoperators and Human Augmentation*," in *NASA SP-5047*, Washington, DC: NASA Office of Technology Utilization, 1967

[146] E. G. Johnsen, and C. B. Magee, "*Advancements in Teleoperator Systems*," in *NASA SP-5081*, Washington, DC: NASA Office of Technology Utilization, 1970

[147] E. (ed.) Heer, *Remotely Manned Systems,* California Institute of Technology, 1973

[148] J. R. Vauds, "International status and utilization of undersea vehicles," in *Proceedings of Inter-Ocean 76 conference*, June 1967

[149] V. S. Yastrebov, and G. A. Stepanov, "Underwater robot/manipulator development," in *Marine Technology Society Journal* 12, no 1,1978

[150] J. Vertut and P. Coiffet, *Robot Technology, Vol 3A: Teleoperation and Robotics: Evolution and Development*, Prentice-Hall, 1986

[151] J. Vertut, and P. Coiffet, *Robot Technology, Vol 3B: Teleoperation and Robotics: Evolution and Development*, Prentice-Hall, 1986

[152] M. R. Stein, "Painting on the World Wide Web: The PumaPaint project," in *Proceedings of SPIE The International Society for Optical Engineering,* v 3524, SPIE, Bellingham, WA, USA. pp. 201-209, 1998

[153] M. R. Stein, M. R., "One year of Puma painting: site experiences," in *Proceedings of SPIE The International Society for Optical Engineering*, vol. 3840, pp.200-209, 1999

[154] Personal observation at Aastra Aerospace Inc., Downsview, Ontario in May 1995

[155] T. B. Sheridan and W. L. Verplank, "Human and computer control of undersea teleoperators", *MIT Man-Machine System Lab. Rep.*, 1978

[156] T. B. Sheridan, "Space Teleoperation Through Time Delay: Review and Prognosis," in *IEEE Transactions on Robotics and Automation,* Vol. 9, No. 5, October 1993, pp592-606

[157] W. M. Tam, K. P. Leu, H. A. Marcelo and Y. S. Wong, "Machines accessed via Internet issues and architecture," in *Part of the SPIE Conference on Telemanipulator and Telepresence Technologies VI*, SPIE, vol. 3840, pp192-199, September 1999,

[158] D. Schulz, W. Burgard, and A. Cremers, "Robust visualization of navigation experiments with mobile robots over the Internet," in *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1999, pp942-947

[159] P. Fiorini, A. Bejczy, and P. Schenker, "Integrated interface for advanced teleoperation," in *IEEE Control Systems*, October 1993

[160] W. S. Kim, B. Hannaford and A. K. Bejczy "Force-reflection and shared compliant control in operating telemanipulators with time delay," in *IEEE Transactions on Robotics and Automation*, vol. 8, no. 2, Apr. 1992, pp176-185

[161] G. Niemeyer and J. Slotine, "Towards force-reflecting teleoperation over the Internet," in *Proceedings of IEEE International Conference on Robotics and Automation,* vol. 3, 1998, pp1909-1915

[162] R. Anderson and M. Spong, "Bilateral control of teleoperators with time delay," in *IEEE Internetional Conference on Decision and Control*, vol. 1, pp167-173, 1988

[163] L. Conway, A. Richard, and W. Michael W. Walker, "Teleautonomous systems: Projecting and coordinating intelligent action at a distance," in *IEEE Transactions on Robotics and Automation*, 6(2): 146-158, April 1990

[164] S. Lee, E. Zapata and P. S. Schenker, "Interactive and Cooperative Sensing and Control for Advanced Teleoperation," in SPIE vol. 1611 Sensor Fusion IV, 1991, pp516-530

[165] S. Hayati, and S. Venkataraman, "Design and implementation of a robot control system with traded and shared control capability," in *Proceedings of 1989 IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, May 14-19,1989, pp 1310-1315

[166] P. G. Backes and K. S. Tso, "UMI: An Interactive Supervisory and Shared Control System for Telerobotics," in *IEEE International Conference on Robotics and Automation*, 1990, Vol.2, pp1096-1101

[167] Y. F. Ho, H. Masuda, H. Oda and L. W. Stark, “Distributed control for teleoperation,” in *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 2, June 2000, pp100-109

[168] R. C. Luo and R. M. Chen, “Remote supervisory control of a sensor based mobile robot via the Internet,” in *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, 1997, pp1163-1168

[169] K. Brady and T. J. Tarn, “Internet-based remote teleoperation,” in *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998, pp65-70

[170] S. Lee, E. Zapata, and P. S. Schenker, “Interactive and Cooperative Sensing and Control for Advanced Teleoperation,” in *SPIE vol. 1611 Sensor Fusion IV*, 1991, pp516-530

[171] Y. F. Ho, H. Masuda, H. Oda and L. W. Stark, “Distributed control for teleoperation,” in *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 2, June 2000, pp100-109

[172] A. M. Fraser and H. Swinney, “Independent coordinates for strange attractors from mutual information,” in *Physical Review A*, 33:1134 – 1140, 1986

[173] B. A. Huberman and R. M. Lukose. “Social dilemmas and Internet congestion” in *Science*, 277:535– 537, 1997

[174] M. Gilge and R. Gusella, “Motion video coding for packet-switching networks—an integrated approach,” in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Boston, MA, Nov. 1991

[175] S. Biaz and N. Vaidya, "Discriminating congestion losses form wireless losses using inter-arrival timers at the receiver," in *IEEE Symposium ASSET'99*, RiChardson, TX, USA, March 1999

[176] S. Cen, P. Cosman, and G. Voelker, "Eng-to-end differentiation of congestion and wireless losses," in *Proceedings of ACM Multimedia Computing and Networking 2002*, San Jose, CA, Jan. 23-24, 2002. SPIE vol. 4673, pp. 1-15

[177] A. Rastogi, *Design of an Interface for Teleoperation in Unstructured Environments using Augmented Reality Displays*, M. Sc. Thesis, University of Toronto, 1996

[178] H. Balakrishnaa, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," in ACM SIGCOMM'96, Aug. 1996