Real-Time Data Flow Models and Congestion Management for Wire and Wireless IP Networks

By

Guy-Alain Lusilao Zodi

Supervisors:

Mqhele E. Dlodlo

Gerhard de Jager

Co-supervisor:

Keith L. Ferguson

This thesis is submitted in partial fulfillment of the academic requirements

for the degree of

Doctor of Philosophy in Electrical Engineering

in the Faculty of Engineering and The Built Environment

University of Cape Town

August 2011

As the candidate's supervisor, I have approved this dissertation for submission.

Name: Prof. Mqhele Dlodlo, UCT Associate Professor

Signed: _____

Date: _____

Name: Prof. Gerhard de Jager, UCT Emeritus Professor

Signed: _____

Date: _____

Declaration

I declare that this thesis is my own work. Where collaboration with other people has taken place, or material generated by other researchers is included, the parties and/or materials are indicated in the acknowledgements or are explicitly stated with references as appropriate.

This work is being submitted for the Doctor of Philosophy in Electrical Engineering at the University of Cape Town. It has not been submitted to any other university for any other degree or examination.

Guy-Alain Lusilao Zodi

Name _____ Date _____

Dedication

To Natasha Lusilao Kimakesa, my beautiful wife, who always keeps me from falling apart during the endless years of the PhD programme. Thanks for being so caring and for being there during my most difficult moment.

To Joel Lusilao, my son, who is his father's pride and who always teaches me new things in life.

To Joyce Lusilao my coming daughter for all the excitement you are already bringing in the house.

To my parents: Felix Willy Lusilao-die-Lumba wish you were still alive to witness my progress and Angelique Mayiza, for their many sacrifices and encouragement.

Finally, to my savior and lord Jesus-Christ without whom, nothing would have been possible,

I dedicated this work.

Abstract

Since their introduction in the early 1990s, video streaming applications have experienced an explosive growth and transformation, from novel applications into one of the main manner in which people experience the Internet today. The Internet however is a best effort transmission platform with no quality of service (QoS) guarantees, which creates several challenges for video streaming applications. The major one being the fluctuations in the network throughput, which do not often match the video traffic pattern and negatively affect its service-related requirements, characterized by stringent delay and delay variation (jitter) as well as high throughput.

Still, video streaming over the Internet remains possible provided that the network is sufficiently provisioned to accommodate the bit rate requirement of the video. Problems arise when new requests for network resources from other web, file transfer protocol, telnet or other video applications cause an aggregate traffic load larger than the network's capacity. When such a situation occurs, the network experiences long queues at routers which lead to excessive delay and high delay jitter for the video packets. The situation is further exacerbated when the queue length at the router exceeds the maximum number of packets allowed at the router's buffer resulting in loss of video packets. This problem, known as network congestion, not only compromises the video throughput performance and impairs its perceptual quality but may also interrupt the display of the transmitted video. To help reduce packet loss, delay and delay jitter and ensure continuous display of the video clip at the receiver end, a mechanism such as congestion control has been proposed.

For streaming applications, congestion control may take the form of rate adjustment. Rate adjustment mechanisms attempt to minimize the probability of congestion by adjusting the rate of the streaming video to match the available capacity of the network. This can be achieved either by adapting the quantization parameter of the video encoder or by varying the rate through a scalable video technique. This thesis proposes a congestion control protocol for streaming video where an interaction between the video source and the receiver is essential to monitor the network state. The protocol consists of adjusting the video transmission rate at the encoder every time that a change in the network conditions is observed and reported back to the sender by the receiver. The proposed protocol runs on top of the real-time transport protocol, and takes advantage of its sister protocol real-time transport control protocol for quality of service control. Transmission rate is adjusted at the sender using the square increase and multiplicative decrease

rate-based congestion mechanism. Since packet loss is not a reliable indicator of congestion, the proposed protocol includes an additional congestion control criterion based on the delay factor and the average cumulative jitter. The average cumulative jitter and delay factor scheme are used to detect incipient congestion prior to a loss of a packet. In this way, rate adjustment is done not only after a loss of a packet as it is the case in most control congestion protocols but also in a way that can anticipate packet loss.

Furthermore, to improve the aggressiveness and responsiveness of the proposed protocol to changes in the available network throughput, the protocol is enhanced with a proposed composite filter that better captures the dynamic process of the round-trip time than the classical round-trip time filter currently used in the implementation of the standard transport control protocol TCP. The performance evaluation of the composite filter is conducted using live experiments with the ping protocol and the open source media streaming pilot live555. Results are compared with those of the classical TCP filter, and the composite filter shows a much better capability to recover from sudden changes in the round-trip time process than the classical TCP filter, providing the proposed protocol fast adaptation to changes in network conditions.

For the proposed congestion control protocol, the performance evaluation is conducted using network simulations with trace files built from real video clips, and the results using both network-related metrics and video quality measurement shows that the proposed protocol exhibits better performance with respect to lost frames ratio and jitter, and hence an improved quality display than the standard TCP-friendly rate control currently used. An experimental pilot of the proposed protocol is also currently running over the Internet when accessing the South-African YFM/YTV (Youth FM and Youth TV) website using the following URL address: http://www.yfm.co.za/ytv/.

Moreover, for wireless networks, where the bit rate error is high and often due to channel impairments, forward error control (FEC) packets are transmitted along with original video packets to recover packets loss at the destination end without them being retransmitted by the sender. However, this technique comes with a cost of increase overhead in the network; which may aggravate the network condition. To overcome the hurdle, this thesis proposes an adaptive forward error control mechanism that dynamically adjusts the number of error control packets based on the traffic load at the network and the state of the wireless channel. The proposed protocol is an extension of an existing mechanism that respectively computes the number *x* of FEC packets as a function of the traffic load and the number y of FEC as a function of the wireless status and linearly combines the two values to obtain the number of FEC

packets to add to video data. The linear combination parameter is set such as the contribution *x* of the traffic load is weighted higher than the contribution *y* of the wireless channel, since the main concerned was to improve delivery of video content with sudden scene changes.

However, weighing the contribution *x* of the traffic load higher than the contribution *y* of the wireless channel for the whole video session may degrade the perceived quality of the video content, especially when the network is unloaded and the wireless channel has a high bit rate error. The approach proposed here addresses the issue by dynamically switching the value of the weight parameter, that is, by allocating a higher weight to the contribution *y* of the wireless channel if the wireless channel status is bad and the network not congested; conversely by giving a higher weight to *x,* the contribution of the traffic load when the network is congested and the wireless channel in good state.

Simulation results using video trace files show an improvement in term of user-perceived quality for a streaming session of a video clip with sudden scene changes, when the network is experiencing a higher packet error rate compare to two existing solutions that are implemented at the base station and use the same sensing mechanisms.

.

Acknowledgements

I have been gifted with wonderful supervisors, to whom I am indebted for their unstinting support, guidance, advice and encouragement over the course of my PhD program. Prof. Mqhele E. Dlodlo, Em. Prof. Gerhard de Jager and Dr. Keith L. Ferguson, this work owes a lot to your counsel, for which I would like to express my sincere gratitude.

To a host of friends and colleagues, thank you very much for your warm welcome and support. My special thanks to Josephine Nakato Kakande and Gloria Mohasi for the help in editing this thesis. Thanks to Dulce-Maria Boneke Nokonoko, Veronica Setongo, Naga Rohini Koduri, Mthulusi Velempini, Sabelo Dlamini, Smart Charles Lubobya, Roaldje Nadjiasngar, Abel Ajibesin, Sikhumbuzo Mthombeni and the entire Mqhele E. Dlodlo research group. Thanks a lot to Dr. Morrison for all the help in filtering and tracking techniques. Gergios Kiomourtzis' replies to my queries are also greatly appreciated.

To my wife Natasha Lusilao Kimakesa, you always have been able to bolster my confidence when the going got tough, and cheered me on when I flew ahead with the wind in my sails. My thanks for your presence and encouragement that have meant more to me than you know.

To my pastor Crispin Kayembe Kakonde, for all the support, encouragement and advice you never cease to provide.

Finally, without the support of my family, it would have been difficult to go through the PhD programme – I am grateful for the mainstay that you have been and continue to be. Mum Angelique, Yvette, Mickel, Delhi, Eugene, Julien Lusilao – and many others I could not name here, thank you and may God continue to bless you richly.

This research was funded by the DST Innovation Fund on project no. T70026: Bandwidth Adaptive Real-Time Video Broadcasting over the Internet.

Table of Contents

List of Figures

Glossary

Autonomous System (AS): A collection of networks under a common administration that share a common routing strategy. An AS is subdivided into areas and is sometimes called a domain.

Acknowledgement: A feedback packet sent by the receiver to the sender to acknowledge a receipt of a packet.

CIF: Common Intermediate Format is a video format with each frames containing 288 lines and 352 pixels.

DVD: Digital Video Disk is a high density format for playing full motion video. It provides a high amount of data storage.

H.264: Latest video codec standard developed by the ITU-T video coding Experts Group (VCEG together) with the MPEG to compress video efficiently when retaining the high definition television quality.

Jitter: The delay variations within a single transmitted stream that alter the timing relationship between the multimedia samples which arrive at the receiver.

JPEG: The Joint Photographic Experts Group is a group of experts from the ITU-T and ISO who have developed advanced data compression. It is an easily compressed graphics format that displays photographic as well as graphic images.

IP: Internet protocol is the standard that allows dissimilar hosts to connect to each other through the Internet. This protocol defines the IP datagram as the basic information sent over the Internet.

IPTV: Internet protocol Television is a television service delivered via a broadband IP link using data communications wiring.

ITU: International Telecommunication Union is an organization sets up to promote worldwide standards and the development of linked networks throughout the world.

Live555: An open-source set of libraries written in C++ used to set up streaming environments.

MPEG: Moving Picture Experts Group is the name associated with a family of standards used for coding audio-visual information in a digital compressed format.

Multicast: One-to-many transmission of media content.

QoS: Quality of Service is a measure of performance for a transmission system that reflects its transmission quality and service availability.

QCIF: Quarter common intermediate format is a video format with each fames containing 144 lines and 176 pixels.

RTT: The Round-Trip Time is the elapsed time between the sending of a packet and the reception of its acknowledgement.

Telnet:A standard IP protocol used to access remote hosts**.**

Unicast: One-to-one transmission of media content.

YUV: A YUV colour model describes colour information in terms of luminance (Y) and chrominance (U, V).

Acronyms

AIMD	Additive Increase and Multiplicative Decrease
ARQ	Automatic Repeat reQuest
CBR	Constant Bit Rate
CUSUM	Cumulative SUM
DiffServ	Differentiated Services
DCCP	Datagram Congestion Control Protocol
DS-REAFEC	Dynamic-Switch Research on Enhance Adaptive Forward Error Control
EAFEC	Enhance Adaptive Forward Error Control
EWMA	Exponential Weight Moving Average
FEC	Forward Error Correction
FTP	File Transfer Protocol
GAIMD	General Additive Increase and Multiplicative Decrease
GoP	Group of Picture
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IIAD	Inverse Increase Additive Decrease
IntServ	Integrated Services

LAN	Local Area Network
LDP	Loss-Delay based Adjustment protocol
TFRC	TCP-Friendly Rate Control
TCP	Transport Control Protocol
RAP	Rate Adaptation Protocol
RB-SIMD	Rate Based Square Increase Multiplicative Decrease
REAFEC	Research on Enhance Adaptive Forward Error Control
RRB-SIMD	RTP Rate-Based SIMD protocol
RGB	Red Green Blue
RSVP	Resource Reservation Protocol
SIMD	Square Increase Multiplicative Decrease
SMG	Statistical Multiplexing Gain
TV	TeleVision
UDP	User Datagram Protocol
VCD	Video Compact Disk
VoD	Video on Demand
VoIP	Voice over IP
VBR	Variable Bit Rate

Chapter 1. Introduction

The Internet was introduced in the early 70's by researchers that used the network to facilitate the exchange of information between remote computers [1] [2]. At that time the supported applications were restricted to protocol such as the remote terminal (telnet), *electronic* mail (e-mail) and *File Transfer Protocol* (FTP). Only a few research laboratories and major universities in the *United States of America* (USA) were connected to and experiencing the benefit of this exciting technology [3]. The rapid growth thereafter of the network to encompass the public market in the early 90's caught many by surprise and was largely due to the birth and success of the Web [4].

Web browsing is an Internet application that relies on the usage of the *Transmission Control Protocol* (TCP) for reliable delivery of packets (segments of data) between a web server and client. To guarantee the reliable transmission of packets, TCP establishes a connection-oriented session between the server and the client. The aim of this connection is to exchange important information on the state of the two end-hosts, rather than to reserve bandwidth from the network along the path from the server to the client. Upon connection establishment, packets flow from the server to the client. Due to the complex structure of the Internet, which constitutes of an inter-connection of several autonomous systems with different characteristics, TCP relies on the data link layer to ensure correct delivery of packets from one network to the next network, that is, one router to the next router. The service provided for at routers is the simplest one and consists of the assurance that each router will try its best to deliver a packet to the correct next router until the packet reaches its final destination. If a packet enroute to the client encounters a full router buffer, the packet will simply be dropped. For this reason, the Internet is often referred to as a "best-effort" network. In case of a packet drop, the server uses a feedback message from the client, also called an "acknowledgement", to detect the missing data packet and retransmit it.

The retransmission mechanism of TCP has worked well for web browsing and applications with similar requirements such as e-mail, Telnet and FTP. These applications, termed "elastic", are tolerant of the delays caused by the retransmission mechanism of TCP and are the dominant applications on the Internet.

However, with recent advances in compression technology, high-bandwidth storage devices and high-speed networks [5] a new class of Internet applications called "real-time

multimedia" has emerged and is rapidly growing in popularity. Real-time multimedia is expected in the near future to comprise the largest portion of the overall data traffic traversing the Internet (see Figure 1.1). The problem is that real-time applications generally have requirements that differ from those of elastic applications and that are difficult to meet given the best-effort service provided by the Internet.

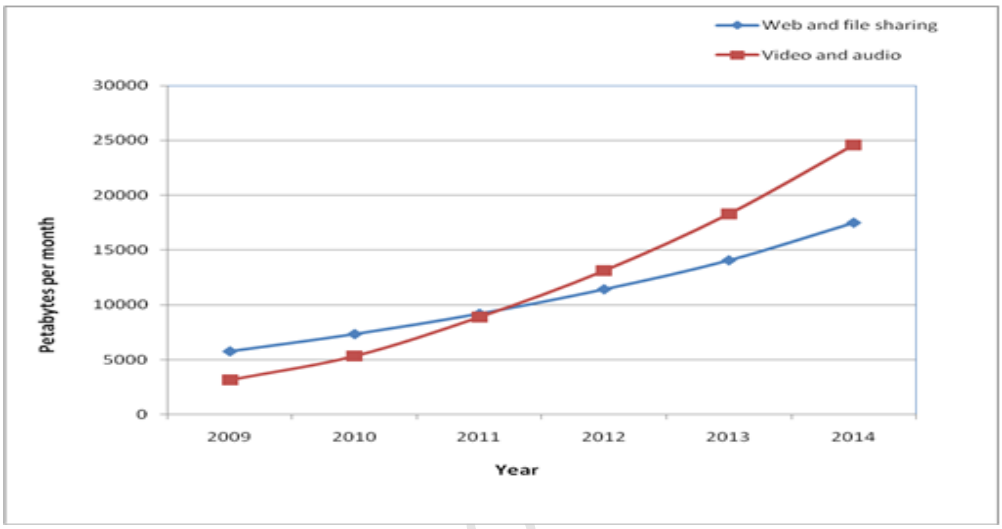

Figure 1.1: Global consumer Internet traffic trends 2009-2014 [6]

1.1 Background and Motivation

Real-time multimedia applications, as the name suggests, include in general audio and video content services with time constraints. The term "time constraints" means that the audio and video data need to arrive at the destination within a predefined time limit. For example, video needs to be played-out continuously. If the video packets are late, the playback process may be interrupted, which is annoying for the human observer.

Real-time transport of both live and stored video is the predominant component of real-time multimedia [5]. In this thesis, streaming stored video, which refers to real-time transmission of stored video, is of primary concern. Stored video is composed of a set of compressed video files in various formats, such as MPEG-4 [7] [8] or H.264 [7] etc., stored at a terminal or server. From the terminal the video is then transferred to the receiver using two common techniques, the download-and-play method or the streaming method. In the download-and-play mode, play-out is delayed until the download of the entire video file is completed, incurring an unacceptably long transfer time if the video file is large and the network bandwidth low. The download mode

also requires end devices with enough memory space to store the video file. This is in some cases not possible, generally when the video file is too big and the end device has a low memory capacity, as is the case with most mobile handsets. To avoid the unnecessary delay and high-storage devices needed by the download-and-play mode, multimedia applications such as distance learning, video-conferencing and entertainment rely on the streaming technique for video. Here, the play-out of the video starts when part of its content has been received and decoded, with an initial delay of only few seconds and thus almost in real-time.

Despite this advance, streaming video over the Internet still poses several challenges. These challenges are essentially due to the real-time nature of the streaming video. Video streaming typically has stringent throughput, delay, jitter (inter-packet delay variation) and loss requirements for correct and satisfactory operation. In traditional circuit-switched networks such as the telephony system a connection (call) is prior to the start of a conversation assigned a fixed bandwidth (channel bit rate) and bounded delay to guarantee a certain *Quality of Service* (QoS). The QoS in this case is measured in terms of connection setup delay, voice quality or trunk availability [9]. However, in packet-switched networks such as the Internet, information flows in packets. Packets carry header information about their source and destination hosts and may be of different sizes. All packets share common resources such as router buffers and links. The inherent nature of the Internet does not differentiate between packets belonging to different flows. As a result packets of a particular flow can reach their destination via different paths and therefore experience different delay, jitter, and loss rates along the way. Furthermore, the Internet does not impose any limitation on the number of simultaneous flows that can access the network; for this reason the bandwidth associated with a flow varies in an unpredictable manner. This poses serious problems for video streaming applications with their strict bandwidth, delay, jitter and packet loss requirements.

Streaming video over wireless links adds further complications. The wireless links are prone to error, time variance and bandwidth limit; and these constraints render the design of streaming applications very difficult. Over the past few years, channel coding techniques such as *Forward Error Control* (FEC) and *Automatic Repeat reQuest* (ARQ) [10] [11] have been proposed to improve the reliability of the wireless links. FEC and ARQ are designed to recover from packets loss, by adding redundant data packets to video stream and by retransmitting loss packets respectively. However, both techniques come with a cost of reduced bandwidth available

for transmission due to error correction overhead in the FEC and that of increased delay due to retransmission of data in the ARQ case.

Another challenge streaming video faces is related to the video content itself. Since raw video consumes a large amount of bit rate capacity, a video encoder is normally used to process the video content and produce a compressed bit rate format. In general, a video encoder compresses a raw video into *Variable Bit Rate* (VBR) traffic. The variability in the output bit rate is caused by the content of the video sequence which typically varies over time. To reduce the effect of this variability during transmission, the VBR traffic is passed on to a rate controller. The function of the rate controller is to match the output bit rate of the encoder to the link's transmission rate. Rate controllers may operate in two modes; *Constant Bit Rate* (CBR) mode or controlled VBR mode [12]. In CBR mode, rate controllers generate flows with variable quality, constant bit rate and bounded delay while in controlled VBR mode the generated flows have steady quality and are only characterized by the peak rate. The CBR flows are easy to transport as their behavior is easily predictable. CBR traffic is suitable for transport over networks with bandwidth reservation capability, such as the circuit-switched telecommunication networks [12]. In a best-effort context CBR may waste the network bandwidth due to a very low *Statistical Multiplexing Gain* (SMG). SMG is a measure which, given the link capacity and the quality per flow, describes the potential of serving a number of video flows at the same time [13]. VBR flows on the other hand use the network bandwidth more efficiently [12] [13] through their large SMG and are suitable for best-effort service [12]. However, VBR flows are difficult to transmit because of the possibility of simultaneous occurrence of peak bit rate from multiple uncorrelated flows and the uncontrollable duration of such peak bit rate [13]. This can lead to excessive delays due to queue build-up at routers and possible packet loss over significant time periods.

To address the delay and high bandwidth demand issues, the *Internet Engineering Task Force* (IETF) proposed to provide the Internet with services models and mechanisms to meet the demand of flows in terms of QoS. QoS is defined in [14] as the capability to provide a flow with resource assurance and appropriate service to meet its requirements. Among the models that were proposed are the *Integrated Services* (IntServ) and *Resource Reservation Protocol* (RSVP) [15] model and the *Differentiated Services* (DiffServ) model [16]. The two models are briefly discussed below.

IntServ is a framework designed to provide QoS to individual flows on the Internet. The idea was to develop a technology that can integrate a range of different services into a single network. IntServ introduces two service classes namely guaranteed service [17] and controlled-load service [18], in addition to the basic best-effort service, that are geared towards meeting the needs of applications. Guaranteed service was introduced for applications requiring fixed delay bound such as streaming video and controlled-load service for applications that need reliable transmission and enhanced best-effort service. The operation of the IntServ framework can be described as follows. A sender needing a specific QoS guarantee initiates a setup procedure to request resource reservation at routers, using RSVP protocol. If the request is accepted, RSVP sets up related flow state information in each router along the path from source to destination specifying the class and the need in resources for the incoming flow. The reservation remains as long as the flow is active, but expires if the state information is not used for a certain time period. However, IntServ through RSVP creates an amount of states' information at routers which increases proportionally with the number of flows. This places a huge storage and processing overhead on the routers. Moreover, the requirements for routers to have RSVP, admission control, classifiers and packet scheduling is too high.

DiffServ architecture was proposed to overcome the scalability problem of IntServ and RSVP. DiffServ is based on the idea of using limited service classes, four in this case and marking packets based on the service class to which the data flow belongs. Packets belonging to different classes are treated differently by the network routers. It is up to each network router at the path between the source and the destination to decide how to treat a specific class relative to other classes. There is a limited number of service classes that are introduced in addition to the best-effort service, among which are the "premium service" used by applications that require low delay and a low jitter and the "assured service" used by applications that require better reliability than the best-effort service. The limitation placed on the number of service classes ensured a number of states' information at the router proportional to the number of service classes rather than to the number of flows traversing the routers as is the case for the IntServ. This makes the DiffServ model more scalable than the IntServ model. However, while IntServ could provide flows with an absolute QoS guarantee through its signaling and reservation mechanism, DiffServ only provides a kind of priority based QoS.

The deployment of both IntServ and DiffServ has been hampered by a number of issues [9], such as the scalability problem for IntServ and the lack of an appropriate mechanism for accurate admission control for DiffServ. Besides, even if both protocols were successfully deployed or will be fully deployed in future, the high amount of network resources needed by the protocols will result in high costs of the associated services. This in turn will leave a large number of users (those with limited financial resources) with no choice other than to rely on the low cost best-effort service of the Internet for transmission of their video content. *For Streaming video over the best-effort platform one possible way to achieve a certain QoS is to control the packet loss ratio, delay and jitter at the hosts, i.e. end-sides of a connection [*19*]. This can be done by appropriately regulating the rate at which video packets are injected in the network. For example, a video sender can to a certain extent increase the transmission rate of video packets as long as the network is not congested, and as soon as the network becomes congested (queue build-up or packets loss), immediately react by decreasing the transmission rate. In this way queue builds-ups (delays) and packets loss at routers are maintained within a certain level. This congestion control technique often referred to as "rate adaptation" is the approach considered in this thesis.*

1.2 Problem Definition

Rate adaptation can significantly improve streaming video delivery over the Internet. It is a process under which a receiver informs the sender of the transmission rate at which it is prepared to accept packets. In the dominant Internet's transport protocol TCP, this information is fed back in the advertised window field of every acknowledgement packet and transmission rate is increased by one more packet per *Round-Trip Time* (RTT) interval if a packet loss is not reported; where RTT is the elapsed time interval between the sending of a packet and the reception of its acknowledgement. If packet loss is reported, the current rate (window) is decreased to one half [20]. This congestion control mechanism known as the *Additive Increase and Multiplicative Decrease* (AIMD) [21] is to a great extent responsible for the remarkable stability of the Internet [22]. However, the halving of the transmission rate for a single packet loss causes dramatic sending rate fluctuations. If AIMD is applied to a streaming session, the high rate fluctuations may result in a video sequence that randomly varies in quality or in the need for a large decoder buffer at the receiver to smooth its effect. The first option can cause

viewer discomfort while the second option may result in poor interactivity and delayed response. Furthermore, AIMD congestion control rule assumes that a packet loss is due to congestion. This assumption is not valid [23] in wireless environments for example, where packets loss is not only due to congestion but also to fading of channels. Because of all these reasons and also due to the additional delays caused by the reliability mechanism of TCP, TCP is not often seen as an ideal protocol for streaming applications.

Streaming video commonly relies on the *Real-time Transport Protocol* (RTP) for data transmission. RTP is commonly run over the transport protocol services provided by the *User Datagram Protocol* (UDP) rather than TCP. UDP is preferred to TCP because it only includes basic transport layer functionalities to guarantee fast delivery of data to the correct destination process. There is no retransmission of data in case of packet loss and no regulation of the transmission rate when the network is congested. In this way data packets are transmitted quickly, as determined by the application rate and pattern.

TCP, on the other hand, is based on the principles of reliability, flow control and congestion control. The reliability and congestion control have already been described in this chapter. Flow control ensures that the receiver buffer is not overloaded by the transmitted data while congestion control is implemented to allow fair utilization of the network resources amongst competing TCP flows. Deploying non-congestion controlled UDP traffic will not only impact the stability of the Internet but also result in unfairness towards competing TCP traffic. The unfairness is manifested when in response to packets loss, TCP flows sharing the same congested bottleneck with UDP traffic reduce their transmission rate leaving the scarce bandwidth to unresponsive UDP traffic. Congestion collapse will arise from a network continuing to transmit packets which will simply be dropped before reaching the destination. *Therefore UDP sources need to be enhanced with congestion control algorithms to preserve the stability of the Internet. The algorithms will not only reduce* the *transmission rate in a smooth manner in response to congestion (packet losses or queue build-up) to suit streaming video applications but also be fair to other streaming applications. Moreover, since today's Internet is still dominated by TCP traffic, the algorithms in addition, must operate in a friendly manner towards competing TCP flows, hence the concept of "TCP-friendly"[22][24].* The term TCP-friendly simply means that TCP flows should be allocated a fair share of bandwidth when competing for bottleneck bandwidth with such algorithms [24].

Several TCP-friendly algorithms have been proposed in the literature [19] [24] [25] [26] [27] [28]. The *TCP-Friendly Rate Control* (TFRC) [24] algorithm is one of the most popular and often used as the standard. The TFRC algorithm uses packet loss metrics as well as round-trip time to compute the acceptable rate at which to transmit data. It relies on the steady state analytical equation of the TCP throughput [29] that models the amount of traffic consumed by a TCP flow when subjected to varying packet losses conditions. TFRC is smooth, fair towards other TFRC flows and friendly to TCP flows. However, TFRC presents a number of disadvantages. First, TFRC wastes bandwidth in the sense that it is slow to react to changes in the available bandwidth. Second, TFRC reacts to congestion when a packet loss has already occurred [22]. For predictive video coding, packets loss do not only result in decoding errors of the current frame, but also in quality degradation for subsequent frames included in the dependency chain [30]. Finally, in the wireless context where channels (links) have a high random bit error, TFRC is inefficient. TFRC always interprets a loss of a packet as link congestion regardless of the wireless channel condition. As a result the occurrence of packet loss is over-estimated and the sending rate of the video is unnecessarily reduced in accordance with the over-estimated packet loss rate. Another issue associated with TFRC, is the difficulty in estimating packet loss ratio and RTT. These issues created the need for a new end-to-end congestion control mechanism to enable and improve stored streaming video application delivery over the Internet.

1.3 Thesis Assumptions

The proposed solutions for streaming video over the Internet presented in this work are based on the following underlying assumptions:

- The network is providing best-effort service and is dominated by TCP flows.
- The network is forwarding packets according to the unicast Internet Protocol (IP). There is no notification of congestion from the network devices. The link bandwidth, delay, buffer size or the number of flows competing for the shared link bandwidth is unknown.
- The wireless network is provided with a so-called smart base station capable of adding redundant data to video data that are sent to mobile nodes.

- The video is encoded in a scalable format with recently developed coding schemes such as MPEG. Scalable schemes usually encode the video into a base layer which can be independently decoded and an enhancement layer, which can only be decoded together with the base layer. The enhancement layer is introduced to provide more flexibility in meeting different bit rate and different delay requirements.

1.4 Research Questions

The thesis answers the following central question; can the sending rate of a video session operating under normal or congested best-effort network be regulated in real-time so as to ensure continuous high quality display of the video clip at the receiver?

With the following specific questions,

- Can this regulation be done in a TCP-friendly way?
- How can the states of the end-to-end network path be estimated?
- How can the onset of congestion be predicted?
- How can the visual quality of the video clip be protected against faulty networks?

1.5 Objective and Contributions

As discussed in the preceding sections, the main objective of this research is to control the streaming media transmission rate such that the media encoder at the source is able to adjust its bit rate to match the available bandwidth of the network. The mechanism must operate in a "friendly" way towards competing TCP flows in order to allow fair utilization of the shared link and at the same time for the video, provide acceptable frame loss ratio and low delay (< 300 ms) or delay jitter bounds. Thus, the major objectives and contributions of this thesis are summarized as follows.

- **A new rate control scheme for video streaming**. For video streaming delivery over the Internet, this thesis presents a unicast transport protocol named RTP Rate-Based SIMD protocol (RRB-SIMD). RRB-SIMD operates on top of RTP and takes advantage of the *Real-Time Transport Control Protocol* (RTCP) reports to multiplicatively decrease the transmission rate in response to congestion and

quadratically increase the transmission rate in the absence of congestion. Another particularity of RRB-SIMD is that it uses, in addition to packet loss, the cumulative jitter as a control criterion to detect incipient congestion prior to loss of packets. The cumulative jitter is reinforced with the delay factor, a measure of the buffer occupancy at the bottleneck path between the sender and the receiver, to reduce risk of unnecessary decrease of the transmission rate every time that incipient congestion is detected through the cumulative jitter scheme.

- **A new filter for RTT prediction**. Packet loss and RTT are critical in the implementation of transport protocols. This thesis presents a recursive model that uses a sample of RTT values to construct a first-step predictor using a combination of an expanding memory polynomial filter with a fading memory polynomial filter. The resulting filter is called the composite filter and it provides improved results in comparison to the standard approach currently used in TCP. The composite filter is included in the implementation of RRB-SIMD.
- **An Adaptive FEC scheme**. This thesis also presents an adaptive *Forward Error Correction* (FEC) mechanism to enhance delivery of video packets over wireless local area network environments.
- **A novel approach to improve TFRC performance**. This thesis studies and analyzes the performance improvement of TFRC enhanced with adaptive Kalman and adaptive TCP filters. The two filters use the cumulative sum (CUSUM) mechanism to detect drastic changes in the average RTT and reset the predictions to improve RTT. However, both filters rely on the negative drift and the alarm threshold which are difficult to set in real-time.

1.6 Scope of research

As stated above, in practice a decoder buffer is used at the receiver to smooth the bit rate fluctuations of the streaming video. However, starting the playback when part of the video has been downloaded and decoded may lead to decoder buffer overflows or underflows. An underflow occurs when there is not enough data in the decoder buffer at the time a frame needs to be decoded and displayed. This leads in a display problem or play-out interruption. In

contrast, an overflow occurs whenever there is too much data arriving at the decoder buffer. As a result, a part of the video content is discarded and thus lost. This in general leads to significant quality degradation. These two aspects of the streaming video are not dealt with in this research.

1.7 Publication List

Part of the above mention contributions are contained in the following documentation list by this author

- G. A. Lusilao-Zodi, M. E Dlodlo, G. de Jager, and K. L. Ferguson, "Round-Trip Time Estimation in Telecommunication Networks using Composite Expanding and Fading Memory Polynomials," *15th IEEE Mediterranean Electro-technical Conference (MELECON 2010)*, Apr. 2010, pp. 1581-1585.
- G. A. Lusilao-Zodi, J. N. Kakande, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "Towards Improved TCP-Friendly Rate Adaptation for Real-time Applications using Adaptive Filters," *Proceedings of the South-Africa Telecommunication Network and Applications Conference (SATNAC),* Spier Estate, South-Africa, Sept. 2010, pp. 293-298.
- G. A. Lusilao-Zodi, J. N. Kakande, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "Performance Evaluation of TCP-Friendly Rate Control Enhanced with Adaptive Filters," *International Journal of Advance in Computing Technology,* Vol. 3, No. 1, Feb. 2011, pp. 13-22.
- G. A. Lusilao-Zodi, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "RRB-SIMD: RTP Rate-Based SIMD Protocol for Media Streaming Applications over the Internet," *Proceedings of the 9th Annual Conference on Communication Networks and Services Research* (CNSR), Ontario, May 2011, pp. 69-76.
- G. A. Lusilao-Zodi, and N. Morrison, "A System and Method for Estimating Round-Trip Time in Telecommunication Networks," World Intellectual Property Organization, International Publication Number WO 2011/104620 A1, September 2011. Available from www.wipo.int/pctdb/en/.
- G. A. Lusilao-Zodi, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "A Switching Mechanism for Adaptive FEC in Last Hop Wireless Networks," *Proceedings of the 2nd International Conference on Computational Problem-solving* (ICCP), Chengdu, China, October 2011, in press.

1.8 Thesis Outline

This thesis is organized in six chapters with references at the very end.

Chapter two –this chapter overviews the general problems encountered in video streaming over the Internet, discusses the video compression approach taken in this project and studies that have been done in relation to congestion control for video streaming over the Internet. The chapter

overviews two techniques used to improve video delivery over wireless networks.

Chapter three – this chapter is subdivided into two parts. In the first part, use is made of the TFRC protocol and two adaptive filters with cumulative sum change detection to demonstrate the importance of more accurate filters for RTT estimation. In the second part, a scheme that overcomes the drawback of the cumulative change detection while still providing improved RTT estimation over that of the classical TCP filter is proposed so as to enhance performance of the congestion controller protocols.

Chapter four – this chapter presents the proposed RRB-SIMD rate protocol for video streaming. The protocol includes the cumulative jitter and a delay factor as criteria to detect incipient prior to loss of a packet. Simulations are also conducted to illustrate the performance improvement of the RRB-SIMD compare to TFRC.

Chapter five – in this chapter an adaptive FEC scheme that improves video delivery over wireless networks while adjusting the FEC overhead intelligently to keep the bandwidth consumption at an acceptable level is discussed and tested over a wireless network.

Chapter six – this chapter concludes this thesis.

Chapter 2. Controlling Video Streaming

Congestion is a situation which manifests itself when a router in a network with limited buffer length, processing speed and output link capacity is receiving traffic flow at a bit rate exceeding its capacity. Congestion in large networks may also result from routers or switches having different input and output bit rates. Congestion is a serious issue in communication networks as its can seriously degrade the performance of the network. Early work on congestion control has been largely dedicated to elastic applications where the focus is more on reliability than timely delivery of data. The emergence of streaming applications, with their demand for smooth rate fluctuations and stringent delay, jitter and loss requirements has created the need for a new class of control algorithms that will respond to congestion in a way that suits streaming applications. This chapter gives an overview of certain problems encountered in video streaming over the Internet and discusses the general trends of video traffic characteristics. It then discusses some of the congestion control algorithms proposed for streaming applications, with special emphasis on *TCP-Friendly Rate Control* and *Rate-Based Square Increase Multiplicative Decrease (RB-SIMD)*, the two techniques mainly used in this thesis. Finally, this chapter ends with a brief discussion of two techniques: loss classification and error control used to improve video delivery over wireless networks.

2.1 Brief overview

Video streaming is concerned with real-time transmission of delay-sensitive information. The Internet is a best-effort network originally designed for transmission of elastic applications. Elastic applications by nature contain information which does not have real-time delivery requirements. Whether the content of a web page, an FTP file or an email is delivered a few seconds or more after the user requests it does not really alter the value of the content. However, if a portion of the information is not transmitted correctly, the information is rendered partially or even totally unusable to the receiver. Thus, the initial concern when designing the Internet was to stretch out the delivery time of data packets and prioritise their reliable transmission.

Video streaming on the other hand is characterized by bandwidth-intensiveness and delay-sensitive information. The latter is caused by the video content which constitutes of a continuous flow of information varying in time. Whether a streaming application is tolerant to

the loss of packets or not results in two sub-categories: loss-tolerant and loss-intolerant. Control signals for robots are an example of real-time loss-intolerant content, whereas audio-visual content is generally tolerant, to a certain extent, of packet loss. A loss of less than 2% of video packets does not really affect the perceived quality of the video content. Streaming video applications are also classified into two categories based on their delay tolerance. The first category is unidirectional (non-conversational) streaming, such as Internet radio, Web TV broadcasting and *Video on Demand* (VoD), where there is no strict delay as there is no interactivity involved, and a delay up to 10 s is generally tolerated [13]. The second category is conversational media such as Internet telephony (*Voice over IP*, VoIP), video conferencing and gaming, that have the same delay requirements as traditional conversational applications (telephony), which should generally be less than 150 ms.

In order to meet the stringent requirements of video streaming over the Internet, the initial approach was to provide the Internet with mechanisms for bandwidth-reservation during connections start-up and admit new connections only if the bandwidth demand could be met without hampering the requirements of already established connections. Thus the IETF proposed an end-to-end flow reservation mechanism for the Internet that could support a range of different service requirements. Here, each flow should be charged based on its service requirements. The IntServ was developed to achieve this goal but failed in its deployment due to its scalability problem [15] and the lack of a billing system on the Internet [9]. DiffServ, which was later proposed to address the scalability problem of IntServ by relaxing the end-to-end reservation mechanisms to a hop-by-hop (router-by-router) priority-based service, also fails due to numerous reasons [16] amongst which is the unresolved problem of the Internet's billing system [9]. Thus the use of the best-effort service of the Internet as a service for media streaming applications was the only option left. However, the best-effort nature of the Internet is also unsuited for real-time media traffic, for reasons which are discussed below.

2.1.1 Challenges of Internet streaming

Generally, when a video is streamed over the Internet, the end-viewer expects a continuous display of the content with an acceptable quality. This calls for a steady flow and delivery of packets by the play-out deadline. Beyond this deadline the packets are considered as lost because they are meaningless to the application. However, the Internet is a public packet-

switched network shared by millions of users. Users may run different types of applications. Information from each application flows in packets and all packets are treated equally regardless of the type of application. In the presence of packets from other applications, packets from the video session can experience varying and unpredictable amounts of delay, jitter and packet loss, making the demand for a steady throughput and timely delivery of packets difficult to meet. Furthermore, the Internet is a complex system with various types of equipment including routers, connection links and terminals having various functionalities, processing speeds and capacities which add further complications to the streaming process [31].

The requirements in terms of throughput, delay, jitter and packet loss are the general challenges associated with the transmission of video streaming over the Internet [5, 31]. Throughput, delay and packet loss are common metrics used in performance evaluation of Internet protocols and as such, they are well known. The term jitter, as already pointed out, refers to the variation on per packet basis in network delay within a single transmitted stream and it destroys the temporal relationship between the periodically transmitted media units that constitute a real-time media stream [32] [33]. In the absence of jitter, the video can be played out as received. However, when jitter is present, the inter-arrival time of received video packets varies, resulting in a loss of user's perceived quality of the video. In order to reduce the effect of the jitter, the receiver terminal includes a receiver buffer that stores the incoming packets for a short time period before they are played back. The challenge here is of two types. Buffer overflows and underflows can occur and lead to the jittering [34]. Hence a suitable size of the receiver buffer should be determined. If a large buffer is used, a greater amount of jitter is compensated for without it being noticed by the end-viewer [35]. However, since the objective in video streaming is to provide continuous video display, the initial play back delay should also be minimised, and too much buffering will have an adverse effect. The requirement is for a small buffer in order to achieve an overall low end-to-end delay and at the same time limit the occurrence of buffer underflows to less than 2% of the time [13]. This requirement is difficult to meet in situations where the channel throughput varies too much, as is the case of the Internet. As a result packets arriving beyond the delay limit are dropped at the receiver, generating the same effect as packet loss and decreasing the perceptual quality of the video. During video streaming over the Internet, packets are dropped for various reasons, ranging from poor equipment connections to fading of channels.

However, one of the major causes of packet losses and delays on the Internet is congestion. Congestion arises when the aggregate traffic load from users' applications is larger than the network's capacity. Congestion not only compromises the throughput performance, delay and delay jitter but may also interrupt the display of the transmitted video. To help reduce packet loss, delay and delay jitter, and ensure continuous display of the video clip at the receiver, techniques such as congestion control have been used. Congestion control can significantly improve video delivery over the Internet and is discussed in more detail in Section 2.2 of this chapter.

Congestion is not the only issue video streaming faces over the Internet. Other problems related to video streaming are: the processing of the video content, its storage and transport. These three aspects are addressed in the following six key area of streaming video: video compression, protocol for streaming media, application-layer QoS control, continuous media distribution, streaming server and media synchronization mechanisms [5]. The development of these six key areas has played a critical role in the deployment of video streaming over the Internet [5]. However, in the context of this thesis, only video compression is discussed at length.

2.1.2 Compression

Raw video consists of a series of digital photos called frames, which in turn consist of a grid of pixels (e.g. *Common Intermediate Format* (CIF) 352x288 pixels). Each pixel is represented by three values of one byte each in RGB (*Red Green Blue*) format. Without compression, this produces a quantity of bits that is impractical to store on or transmit over normal computer components and network links. For example, one minute of raw video in RGB format will require a 435 MB storage capacity or a 435MB/minute link capacity for transmission. This requirement exceeds the transfer capacity of most computers' components and Internet access links (for example, the more common ADSL access links have 1.5 Mb/s capacity).

A more elaborated format in which to store video with three full colours is the YUV format. This format takes advantage of the human visual capacity, and represents the three colours in terms of luminance and two chrominance. The luminance which is perceived better by the human eye than the chrominance is represented with the highest resolution. However the size reduction introduced by the YUV format compared to RGB format is not enough for video

streaming [35]. A video encoder is therefore used to compress the video content before it is stored or transmitted. Video encoders are generally supplied with video data at a constant bit rate, and process these data to produce a compressed bit stream. The amount of compression depends on many parameters, such as the value Q of the quantiser [36]. The more the video is compressed (i.e. the higher the value of Q), the lower the average number of bits required to represent the images and the lower the resulting video quality (see figure 2.1) [37].

Uncompressed (262 KB) *Compressed (22 KB, 12:1)* *Compressed (6 KB, 43:1)*

Figure 2.1: Effect of compression on quality on a 300x420 Lena picture

Generally, encoders are classified into two groups: non-scalable and scalable. A non-scalable video encoder produces compressed bit-stream at a fixed target bit rate or target storage size. Examples of such schemes include JPEG [35] for image coding and MPEG-1 or MPEG-2 [36] for video compression. Non-scalable encoders exploit data redundancy (spatial and/or temporal redundancy) within frames to achieve a high compression ratio. However, they also introduced inter-dependency of data within compressed frames, leading to significant artefacts in the decoded frame if a packet (or bit) in the compressed frame is lost during transmission. Similarly, since non-scalable encoders compress raw video at a fixed target rate, they are unable to adapt to change in the available bandwidth, that is, they cannot for example take advantage of an increase in the available bandwidth or storage capacity. Non-scalable encoding works well for applications such as VCDs and DVDs, where a certain storage size is targeted or for TV broadcast where a dedicated capacity is assigned to each channel prior to data transmission. However, for packet-switched networks such as the Internet, where bandwidth varies greatly with time, non-scalable encoding is not appropriate.

Therefore, scalable video encoding is preferred to non-scalable video encoding for video transmission over packet-switched-networks. A scalable encoder also exploits spatial redundancy and temporal redundancy, i.e. redundancy between successive frames, to produce multiple compressed sub-streams to deal with QoS fluctuations in the network. One of the compressed sub-streams is the base stream which comprises of a sequence of Intra-coded frames (I-frames). I-frames are compressed relative to information contained in the current frame and not relative to any other frames in the video sequence. As such, they can be decoded independently. Other compressed sub-streams are enhanced sub-streams built either with a sequence of *Predictive frames* (P-frames) or a sequence of P-frames and *Bidirectional frames* (B-frames). P-frames are encoded using preceding I-frames or P-frames as a reference, such that the information that can be derived from the preceding frame is not coded; only the information that cannot be derived is coded and transmitted (forward prediction). B-frames are encoded in a similar way using the preceding I-frame and following P-frame as references (bi-directional prediction). The main advantage of P-frames and B-frames is that they require fewer bits than I-frames. P-frames and B-frames are correctly decoded only if the corresponding reference frames are successfully decoded. The decoding of the complete bit-stream (the base stream with all enhanced streams) provides a better visual quality compared to the visual image obtained when decoding only the base sub-stream or when decoding the base sub-stream enhanced with only P-frames. With scalable encoding, enhanced B-frames or P-frames can for example be dropped or added from time to time to reduce or increase the bit rate and cope better with the bandwidth fluctuations of the Internet or any other environment with similar behaviour. In this thesis, the scalable video encoder MPEG-4 [35] is used to compress video sequences. Figure 2.2 depicts an example of MPEG-4 video frame sequence [38].

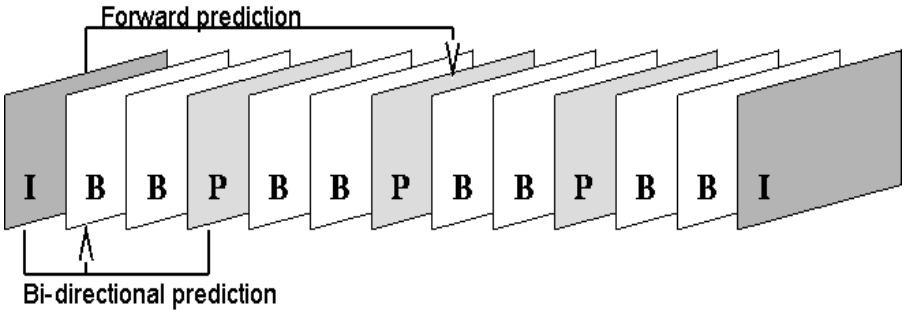

Figure 2.2: MPEG Video frame Types [38]

With MPEG-4, frames are organised in groups called *Group of Picture* (GoP) and then coded in various ways within each GoP. A GoP may consist of one I-frame, some P-frames and sometimes some B-frames between the I-frame and P-frames. In the example of Figure 2.2, the length of a GoP is 12 frames. Therefore, the number of bits would vary from frame to frame and peaks are expected to appear at the beginning of each GoP, that is, after a cycle of 12. Figure 2.4 illustrates the encoding bit-rate trends of a one-minute concatenated video sequence of the official ISO MPEG test sequences News, Foreman, Stephan and Paris. Each sub-component is at 25 fps of CIF resolution in YUV format (see figure 2.3) and encoded using MPEG-4. This concatenated video sequence is used later in the simulation part in this thesis.

Figure 2.3: First frame of "News" (top left), "Foreman" (top right), "Stefan" (bottom left), "Paris" (bottom right).

The most important thing to glean from the bit rate pattern in terms of traffic characteristics and visual perceived quality respectively is that the output is a VBR traffic stream and peaks represent I-frames, which as such are more important than P-frames and B- frames when the

objective is to minimize distortion in the transmitted video. Therefore, if an I-frame is lost during transmission due to congestion or fading of the channel, the negative impact on the perceived video quality is big.

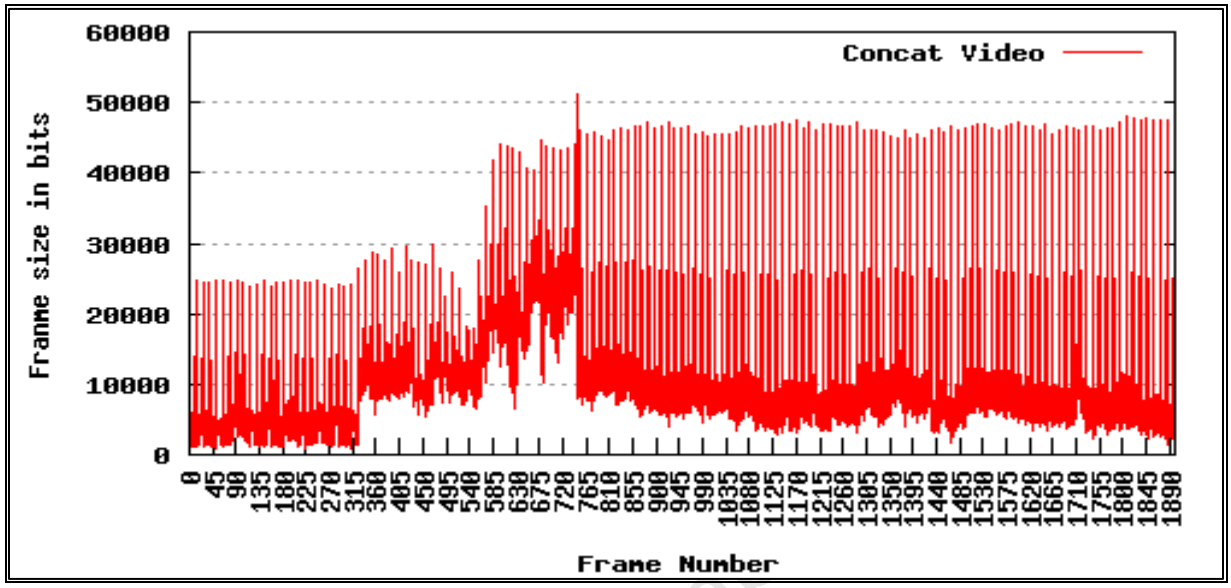

Figure 2.4: Frame size pattern of the compressed concatenated video sequence with Q=2

Before concluding this section, it is important to mention that the receiver end is also equipped with a video decoder, of which the function is to rebuild the video clip with minimum possible distortion.

2.1.3 Controlling Encoding Rate

The variability in the encoder output bit rate is a real issue for many transmission environments. For example, a constant bit rate channel such as the circuit-switched network cannot transport VBR data streams. Packet switched networks, on the other hand, can support varying throughputs, but the average throughput at a specific point in time is limited by factors such as the channel rate and congestion [35]. It is therefore necessary to adapt or control the bit rate in order to produce compressed video data at a rate that matches the available bit rate of the transmission channel. Often, the variable output bit rate of a video encoder is smoothed by buffering the compressed video data prior to transmission. Video data is then removed from the encoder buffer at a rate that corresponds to the available throughput of the transmission channel.

A similar mechanism is also used at the receiver side, where video data fills the decoder buffer at the channel rate but is removed from the buffer at a variable bit rate.

However, buffering comes with a cost of buffer storage and delay, and the larger the bit rate variation, the larger the buffer size and decoding delays [35]. Furthermore, for the buffer approach to cope with an arbitrary variation in bit rate, the buffer size and decoding delays need to be set to impractically high values. Rate controllers, on the other hand, attempt to modify the encoding parameters (such as the quantiser step size) in order to meet the bit rate of the transmission channel [35]. A rate controller, as already pointed out may operate in two modes: CBR and VBR. For a reasonexplained in Chapter 1, rate controllers in CBR modeare suitable for transport over networks with bandwidth reservation capability, such as the circuit-switched telecommunication networks [12]. In a best-effort context CBR may waste the network throughput [13]. Therefore, a rate controller in VBR mode will be used in this project.

There are many models of video encoders operating in VBR mode [35] [36] [38] [39]. The model considered here is a VBR rate controlled encoder of a leaky bucket type as described in [38] [39]. The key idea of the model is to adapt the quantization step of the video encoder every time that a change is detected in the available network throughput. In this way, the compression ratio is increased or decreased depending on the network current condition; and the compressed video produces at the best possible bit rate that can match the available throughput of the network. The model is implemented such as, if a change in the network throughput is reported when a sender is busy transmitting a group of picture, the encoding adjustment will take effect only in the group of picture following the one in transmission.

The only remaining issue now is, how does the sender estimates the rate at which the network is prepared to accept video packets? This question is discussed in depth in the following section.

2.2 Congestion Control

2.2.1 TCP-Friendly Behavior

Video streaming as already pointed out has stringent throughput, delay (jitter) and packet loss requirements. The Internet, however, only provides the best-effort service over which data packets flow and compete for network resources. Since there is no limitation on the number of

flows that can concurrently access the system, the traffic load varies unpredictably, leading sometimes to network congestion. Network congestion is a real threat for streaming applications. In the presence of congestion, video packets experience excessive delays and bursty losses. Packet losses induce frame losses and severely affect the video quality. Thus, congestion control mechanisms are necessary to help reduce packet loss and delays for streaming session. Congestion control is not a new concept on the Internet. It was first introduced in the early 1980s when the Internet experienced a congestion collapse [22] and has since then contributed to its relative stability. Congestion control algorithms on the Internet are implemented as an integral part of the transport protocol TCP. This is because the quasi-majority of the Internet's applications run over TCP. However, emerging video streaming applications do not often use TCP, for reasons that were explained in Chapter 1. UDP, which delivers packets faster than TCP, is instead preferred.

UDP, on the other hand, is not adequately fashioned for streaming applications. RTP was therefore designed with the objective of facilitating media transmission over the Internet. RTP is commonly run over UDP and not TCP due to TCP's disadvantages towards media transmission as described previously. However given the lack of congestion control mechanisms within UDP and with due consideration that the majority of Internet traffic is still TCP-based [40], it has become of importance to regulate the flow of UDP video traffic so as to achieve fair bandwidth sharing. This necessitates that UDP operations be customized in a "TCP-friendly" manner. This means that UDP flows should react to congestion signals as any long-term TCP flow operating under the same conditions would react. This is termed TCP-friendly behaviour. A possible way to provide streaming sessions with inbuilt TCP-friendly congestion control, is to implement at the hosts (sender or/and receiver) mechanisms to adjust the video transmission rate in response to change on the available network throughput. This end-to-end process is referred to as rate adaptation. Rate adaptation not only preserves the stability of the Internet but also improves video delivery at the receiver.

2.2.2 TCP-Friendly Rate Adaptation

By definition, rate adaptation is a technique that consists of selecting the best possible transmission rate that matches the current transmission link conditions, while at the same time optimizing the throughput and packet delay (jitter) of the streaming session [5] [34]. Depending

on whether the sender or the receiver is responsible for regulating the rate, rate adaptation techniques are classified into three categories: sender-based, receiver-based and hybrid-based schemes. Hybrid-based schemes are a combination of both sender-based and receiver-based schemes. A receiver rate-based scheme is a technique where the receiver adds and drops different layers of a scalable video encoder to cope with fluctuation in the network throughput. This approach and hence the hybrid-rate are used more in multicast scenarios and as such are not discussed in this thesis. In a sender-based rate approach, the sender adjusts the sending rate based on feedback information from the network. For example, the sender can decide to reduce its transmission rate to maintain a low packet loss rate when the network condition is poor and to increase the transmission rate to improve the throughput of the session when the network condition improves. Generally, rate adaptation operates in two phases; in the first step the sender estimates the network condition and in the second decides on the appropriate transmission rate. However without the receiver's assistance in assessing the network condition, it is difficult for the sender to obtain accurate estimations of the available throughput. Thus, many existing sender-based schemes incorporate receiver feedback to assess the network condition. It should be noted though that this does not makes them hybrid-based models.

The need for rate adaptations to behave in a TCP-friendly manner was earlier explained in Section 2.2.1. However, this is not the only requirements for their correct operation; indices such as smoothness, aggressiveness and responsiveness are also considered in the performance evaluation. Aggressiveness indicates the ability of the protocol to probe for extra bandwidth by increasing its rate. Responsiveness indicates how fast the algorithm reacts to increased congestion by decreasing its transmission rate. Smoothness is a measure of the steady-state behaviour of the protocol. Aggressiveness and responsiveness are used to analyse the transient behaviour of a protocol, while smoothness is used to analyse the long-term behaviour of the protocol. Achieving an optimum trade-off between the three is not an easy task; for example it is demonstrated in [20] [41] that higher smoothness means less aggressiveness and responsiveness.

Rate adaptations mechanisms are also classified into congestion window-based and rate based schemes, depending on how the sender adjusts the sending rate.

2.2.2.1 Window-Based Schemes

In the window-based category, feedback (acknowledgement) from the receiver is used to adjust a transmission window at the sender, with the window size determining the transmission rate [40]. An example of a window-based scheme is the AIMD scheme [20] [21]. In AIMD, a sender probes for extra throughput by linearly increasing by one packet its congestion window for data acknowledged per RTT, and on detecting packet loss multiplicatively decreases its window size by a factor of two. Thus, the AIMD algorithm can be represented by the following relationship pair:

$$\begin{aligned} &increase: \quad w_{t+RTT} \leftarrow w_t + \alpha \qquad \alpha > 0; \\ &decrease: \quad w_{t+\Delta} \leftarrow w_t - \beta w_t \quad 0 < \beta < 1; \end{aligned} \tag{2.1}$$

where w_t is the congestion window size at time *t*, *RTT* is the round-trip time and Δ the time to detect packet loss since the last window update. AIMD is the approach used in the Internet's dominant transport protocol TCP. TCP congestion control is divided into two phases. First, at start-up of a connection and in order to quickly reach the target network throughput, a TCP sender starts with a small congestion window (*cnwd*) which grows exponentially per RTT, until a slow start threshold (*ssthreshold*) value is exceeded or congestion is detected. When $cnwd > ssthreshold$, TCP enters the congestion avoidance phase. In this phase the sender uses the AIMD algorithm, that is, he increases his window size by one, for all acknowledged data packets per RTT until congestion is detected. TCP uses two indicators to detect congestion: timeout and three duplicated acknowledgement. Timeout is interpreted as severe congestion to which TCP responds by re-entering again the slow start phase with *cnwd* reinitialised and *ssthreshold* cut to half. Three-duplicate on the other hand is viewed as indicating moderate congestion, to which TCP responds by cutting the window size to half.

On the positive side, the following can be said about AIMD, its properties in terms of fairness, stability and oscillations are well understood [41], and make its behaviour easy to predict. The stability of the Internet is due in large part to AIMD [38]. However, while AIMD appears to be effective for elastic applications such as FTP, many streaming applications find the abrupt halving of the window size in response to a single packet loss too severe, as it creates high rate fluctuations; which noticeably reduce the user-perceived quality [40]. In responsive to this limitation, a more general approach of AIMD, the *General Additive Increase and Multiplicative*

Decrease (GAIMD) protocol was proposed. GAIMD is a linear class of window based control, initially introduced in [27] and further investigated in [42] and [41]. The increased and decreased rules are as follows: the sender increases his window size by α if there is not packet loss, and decreases the window size to β of the current value in case of loss indication. GAIMD offers some interesting properties regarding video transmission. In [43], based on experimentation it is demonstrated that a value of ($\beta = 7/8$) smaller than 0.5 is shown to achieve smoother window size adjustments, with α ($\alpha = 0.31$) selected according to Equation 2.2 to ensure fairness towards TCP traffic.

$$\alpha = \frac{4(1-\beta^2)}{3} \tag{2.2}$$

GAIMD is reported to generate a fair share of network throughput even in the presence of a first-in first-out queue and to be safe to employ in both stationary and varying network conditions. However, decreasing the value of β also diminishes the ability of GAIMD to react quickly to change on the available bandwidth (responsiveness). Thus, the selection of parameters (α, β) plays a crucial role in the performance of the GAIMD protocol. For example a GAIMD flow with a large α and small β is very sensitive to network throughput fluctuations and consequently has a highly variable instantaneous throughput [43]. This behaviour, if not appropriately dealt with, may cause frequent interruptions in packet delivery which will have a negative effect on the play back quality.

Binomial techniques [28] are a non-linear approach to congestion control proposed to provide TCP-friendly adjustments to streaming applications over the Internet. Their increase/decrease rules obey Equation 2.3:

$$\begin{aligned} &increase: \quad w_{t+RTT} \leftarrow w_t + \frac{\alpha}{w_t^k} \qquad \alpha > 0; \\ &decrease: \quad w_{t+\Delta} \leftarrow w_t - \beta w_t^l \quad 0 < \beta < 1; \end{aligned} \tag{2.3}$$

Binomial congestion controls satisfy the TCP-friendliness if and only if $k + l = 1$ and $l \leq 1$ with any arbitrary values of α and β. TCP AIMD is a particular case of binomial family with $k = 0$ and $l = 1$ and is proven to be the most aggressive in probing for the available network throughput [28]. Binomial controls with values of $l < 1$ on the other hand have smoother

decreases than AIMD, and present desirable properties for streaming applications. Two particular algorithms, *Inverse Increase Additive Decrease* (IIAD) $(k = 1, l = 0)$ which increases its window size inversely proportional to the current window, and SQRT $(k = 1/2, l = 1/2)$ which has an increase in window size inversely proportional to the square-root of the current window size and a decrease of the square-root of the current window were found to interact well with TCP across a wide number of network conditions. One of the disadvantages though of GAIMD and binomial control techniques is that they provide smoothness at the expense of aggressiveness and responsiveness.

In [25], it is questioned whether or not it is possible to achieve high smoothness in steady state while still maintaining high aggressiveness and responsiveness to cope with sudden changes in network conditions. A protocol called *Square Increase Multiplicative Decrease* (SIMD) is therefore proposed. SIMD uses the history of the window size after detecting packet loss and the current window size to improve the transient behaviour of the AIMD family. Thus, the congestion window size in SIMD increases quadratically, in proportion to the square of the time elapsed since the detection of the last packet loss. In this way, SIMD achieves high aggressiveness and fast convergence to fairness. Equation 2.4 describes the increase and decrease rules of SIMD.

$$\begin{aligned} &increase: && w_{t+RTT} \leftarrow w_t + \alpha(t - t_0)^2 && \alpha > 0; \\ &decrease: && w_{t+\Delta} \leftarrow w_t - \beta w_t^l && 0 < \beta < 1; \end{aligned} \tag{2.4}$$

where, w_0 is the window size after the last window decrease action, and the increase in window depends on w_{max}, the window size just prior to its decrease in response to congestion (i.e. $w_0 = w_{max}(1 - \beta)$).

For SIMD to be fair to TCP flows, α and β must satisfy Equation 2.5.

$$\alpha = 3\sqrt{\beta}/\sqrt{2w_{max}}(1 - 2\beta/3) \tag{2.5}$$

2.2.2.2 Rate-Based Schemes

Rate-based protocols, on the other hand, operate by varying the rate at which the senders transmit based on received information about congestion levels experienced along the communication path [44]. Thus, it is important to accurately estimate the network throughput. Hence the estimation of network parameters, round-trip time and packet loss ratio, plays a significant role in protocol performance. TFRC is an example of an end-to-end rate-based protocol suitable for best-effort protocols like UDP and RTP; and which emulate TCP flow steady state behaviour. Its function consists of carrying streaming video applications, avoiding the abrupt halving of the sending rate on congestion detection while still being fair to concurrent TCP flows. TFRC does not incorporate flow control at the receiver but includes the slow start mechanism of TCP which ends on detection of the first packet loss; after which the protocol enters the congestion avoidance.

During congestion avoidance, TFRC relies on packet loss metrics as well as the RTT to compute the acceptable rate at which to transmit video. Packet loss is counted at the receiver during a predefined period time, usually a RTT, grouping all the losses into the same loss event and send the current loss event rate back to the sender using a feedback report. A packet is considered as loss when three or more packets with a higher sequence number than the one expected are received at the receiver end. The receiver then computes the loss interval as the total number of packets drop between two consecutive loss events. RTT is computed at the sender side, using the time stamp field of the feedback packet and, keeping memory of past RTT values with the *Exponential Weight Moving Average* (EWMA) technique. The sender then inserts both parameters, i.e., the loss event rate and RTT in an equation that models the amount of traffic consumed by a TCP flow when subjected to varying packet loss conditions. The general equation is such that available bandwidth r in bytes/sec is given by Equation 2.6,

$$r = \frac{S}{RTT\sqrt{2/3p} + t_{RTO}\left(3\sqrt{3p/8}\right)p(1+32p^2)} \tag{2.6}$$

where S denotes the packet size, RTT the round trip time estimate, p corresponds to the packet loss event rate and t_{RTO} the TCP retransmit timeout value.

If for some reasons, the sender has not received any feedback after several RTTs, he interprets this as a congestion notification and reduces its transmission rate. TFRC presents the advantage of having a relatively steady sending rate while still being responsive to congestion. Its sending rate is not reduced to half in response to a single packet loss event, but in response to several successive packet loss events. However, the protocol is quite sensitive to packet loss events and round-trip time estimates. In practice these two parameters are very difficult to measure effectively and predict accurately [45] [46], and an entire chapter is dedicated to the estimation of RTT. The TCP long term throughput equation does not capture the transient and short-lived behaviour of TCP, hence TFRC is less responsive to short-term network and session dynamics [41].

Rate-Based Square Increase Multiplicative Decrease (RB-SIMD) [26] is a rate based-protocol proposed to emulate the properties of SIMD. In RB-SIMD the conventional TCP feedback mechanism is replaced with some periodic feedback from the destination which takes two forms. One form is the congestion feedback, where the sender is requested to reduce its transmission rate and the other, the normal feedback where the transmission rate may increase. The way the source sends his packets as a continuous batch, is also an innovation, with each batch identified with a control action number. Packets within a batch are sent at a constant rate, which is determined by the last control action. And the destination returns a single feedback for each batch. If the feedback is a congestion notification, then the sender uses Equation 2.7 to decrease his sending rate

$$\begin{aligned} r_{max} &= r_t \\ r_{t+\Delta} &= r_t \times (1-\beta) \\ r_0 &= r_{t+\Delta} \end{aligned} \tag{2.7}$$

where $r_{t+\Delta}$ is the new transmission rate, r_t the current rate, and r_{max} and r_0 the rate equivalents of w_{max} and w_0. Otherwise the returning feedback is a normal feedback, in which case the sender may increase, maintain or even decrease his current transmission rate [26]. If the rate can be increased, then the sender uses Equation 2.8

$$r_{t+1} = r_t + K \times 2 \times \sqrt{\alpha(r_t - r_t)} \tag{2.8}$$

where r_{t+1} is the new transmission rate and $0 < K < 2$ a parameter depending on RTT. The choice of K is also discussed in [26]. For RB-SIMD to be TCP-friendly, α and β must satisfy the relationship in Equation 2.5 where w_{max} is replaced by r_{max}.

Datagram Congestion Control Protocol (DCCP) [47] is a protocol proposed to provide congestion control to delay-sensitive applications with relaxed packet loss requirements. DCCP protocol uses the transport protocol UDP enhanced with basic TCP functions to make it connection-oriented like TCP. In doing this, DCCP benefits in two ways. Firstly the protocol is guaranteed traversal of network firewalls which blocked UDP packets and secondly it is provided with the possibility to negotiate some parameters during session initiation, such as the congestion control algorithms. DCCP provides applications with two congestion control possibility: TCP-like and TFRC, with TFRC being the most suitable algorithm for video traffic.

Other adaptive rate-based models proposed in the literature include the *Rate Adaptation Protocol* (RAP) [48] and the *Loss-Delay based Adjustment protocol* (LDA) [49]. Both protocols employ the AIMD algorithm and adapt the sending rate by reducing the inter packet gap additively or multiplicatively depending on whether the network is congested or not. The basic difference is that RAP runs on top of a lightweight UDP while LDA runs on top of RTP. RAP includes all the basic functionalities of TCP except those related to reliability. Its feedback mechanism is also less frequent compared to that of TCP and redundant information is added to each single acknowledgement to facilitate detection of packet loss. LDA, on the other hand relies on the RTP and its sister protocol RTCP for feedback reports. However, both inherit the halving rate of AIMD which is critical for streaming applications.

2.3 Loss Differentiation and Error Control

Most of these protocols were proposed to provide rate adaptation to real-time media applications over wire line networks. The major disadvantage of these models is that the wire link is assumed to have negligible bit errors; hence packet loss is seen as a reliable indicator of congestion. However, some links such as wireless and satellite are characterised by high bit error rate due to effects such as fading and multipath; and packet loss due to link errors can occur with probabilities as high as 10^{-2} [50]. In this context, if the sender decreases its transmission rate when a packet loss is due to link error rather than network congestion, the performance of the network may suffer a degradation of up to 20% [50]. Thus, it is important for these adaptive rate

techniques to integrate mechanisms to combat wireless channels impairments. These mechanisms may take two forms, loss classification and error control [51] [52].

2.3.1 Loss Differentiation

In loss differentiation schemes, the receivers uses the session statistics to differentiate between losses due to wireless error and losses due to congestion without making use of any intermediate node, and communicates congestion losses information to the sender who makes used of it by adjusting appropriately the transmission rate. Loss differentiation techniques have been widely used to improve performance of TFRC over wireless networks. In [53] for example, a loss differentiation approach based on the assumption that the variance of the RTT is high when congestion occurs and low otherwise is proposed to facilitate streaming over wireless. A combined packet inter-arrival times and *Relative One-Trip Time* (ROTT) approach is presented in [54] to differentiate packet loss caused by congestion and that due to wireless channel errors. The observation behind this approach is that relative one-trip time delay increases monotonically if there is congestion. Additionally inter-arrival time is expected to increase if there is packet loss caused by wireless channel. Thus the two schemes can be used to differentiate packets losses. [55] is another approach that uses the ROTT and plotted it against time at the receiver. The observation is that ROTT is characterised by spikes that indicates congestion; thus losses during spikes are classified as due to congestion. The problem is that a packet may traverse many congested routers characterised by different level of congestion, resulting in varying spikes and making it difficult to catch spikes of all sizes. The Zig-zag scheme [56] is another scheme which makes use of the ROTT. The disadvantage of all of ROTT models is that congestion schemes based on statistics are not sufficiently accurate and sometimes require cross layer information or modifications of the transport stack. In [57] a TFRC cross-layer scheme that uses link-layer information to differentiate packets losses and adjusts the transmission rate using a TFRC-like equation is proposed to facilitate video streaming over wireless networks.

2.3.2 Error Control

Error control is a technique used to improve the reliability of the wireless channels. Error control mechanisms fall into one of the two classes, *Forward Error Correction* (FEC) and *Automatic Repeat reQuest* (ARQ). FEC relies on an dditional number of data sent along with the compressed video bit stream so that some of the video packets losses are recovered using the

redundant data. Practically, the technique consists of breaking a video stream into segments, with each segment organised into k packets; then for each segment a block code is applied to the k packets to generate $n - packet\ block$, where $n > k$ [5]. The receiver only needs to receive any k packets in the $n - packet\ block$ to perfectly recover the segment. The used of FEC is encouraged because of its small transmission delay. However, FEC comes with two disadvantages, first the cost of reduced network throughput available for transmission due to FEC overhead and second, the impossibility of recovering video data in case of bursty loss exceeding the recovering capability of the FEC code.

ARQ mechanisms, on the other hand, are based on retransmission of video packets that were not received at the destination. ARQ schemes increase the delay due to retransmission of the data, and as such are often dismissed as a mean for transporting real-time video. However if the one way delay is short, ARQ can be a viable solution.

Over the past few years they have been some attempts to develop mechanisms to dynamically adjust the number of FEC packets in order to reduce FEC overhead when a network is congested. Such FEC mechanisms have the advantage of being flexible to varying states of the wireless network channel and are referred to as adaptive FEC schemes. Adaptive FEC schemes have been a topic of interest over the past few years [58] [59] [60] [61]. In these studies, both congestion losses and wireless losses were considered when adapting FEC redundancy. For example, the scheme in [59], decreases FEC redundancy when congestion losses occur to alleviate congestion. Alternatively, when wireless packet losses happen, FEC redundancy is increased to recover more video segments.

However, using only packets losses rate to adjust FEC redundancy is ineffective in recovering from bursty loss [58]. In [61] a burst-aware FEC, where the length of continuous losses is counted, is proposed to overcome burst packet loss. Another intelligent FEC adjustment mechanism is proposed in [59] and later modified in [60]. The approach in [59] presents the advantage of taking into account the congestion level and the error level. However, the limit of this mechanism is that when the queue length is too large only transmitted data without FEC packets affect the quality of the video.

2.4 Summary

In this chapter, some challenges in streaming video over the Internet were briefly discussed, followed by a large discussion on the issue of how to control congestion for streaming applications with emphasis in the rate adjustment technique, and how it is done at sender end to meet the challenges described in Sections 2.1 and 2.2. The wireless limitations have also been considered and two approaches: congestion classification and error control, used to overcome the impairments of wireless channels are highlighted. It is these approaches that are expanded on in later chapters.

Chapter 3. Adaptive Filters for Round-Trip Time Estimation

Chapter 2 contained a description of protocols used to control congestion for video streaming over the Internet. Most of these protocols rely on packet loss and round-trip time estimations for their correct operation. However the estimation of the round-trip time is difficult since it is a non-linear process. The exponential weight moving average presently used is often criticized because of its inability to adjust quickly to sudden changes in the RTT process, making it difficult for control protocols to quickly react to change in network conditions. In this chapter TFRC is used to illustrate the importance of having more accurate estimators for RTT prediction. The exponential moving average used in the estimation of the RTT is replaced by two filters namely, the adaptive TCP filter and adaptive Kalman filter. Based on jitter, delay and throughput results, it is illustrated that the adaptive filters respond better to drastic changes in network RTT, providing better aggressiveness and responsiveness in rate adaptation at the multimedia server. Since, the adaptive Kalman and adaptive TCP filters include parameters which are very difficult to determine on run up, a switching mechanism between the family of expanding memory and fading memory polynomials is proposed at the end of the chapter to capture in a simple way sudden changes in the network RTT and hence enhance performance of congestion controllers for video flows over the Internet.

3.1 Introduction

Most of the rate adjustment protocols presented in Chapter 2 handle congestion by adjusting the video transmission rate based on the packet lost event (or ratio) and the round-trip time experienced by the video stream during its transmission. While window-based protocols react to a single packet event within an RTT by reducing the sending rate, rate protocols based on model of the throughput equation such as TFRC estimate the fair sending rate using packet loss rate, RTT and the model of the throughput equation. Hence, measurement of packet loss events and RTT are highly critical in the implementation of congestion control algorithms. Usually, in order to minimize the effect that a spurious loss event or RTT can have on the overall calculated sending rate, the measured values of packet loss rate and RTT are smoothed using

filtering techniques. While filter techniques such as EWMA is widely adopted for packet loss rate estimation, RTT estimation is still a very open area of research. The EWMA currently used in TCP and TFRC implementations is often criticized for its inability to quickly adapt to changes in RTT. This chapter focus on the estimation of RTT to enhance video delivery over the Internet. The TFRC protocol and two adaptive filters with CUSUM change detection are first used to illustrate the importance of more accurate RTT estimators than the EWMA. Then, a composite filter that overcomes the drawback of filters with CUSUM change detection is proposed to estimate RTT.

RTT as already pointed out, is the time interval that elapses between the sending of a packet and the reception of its acknowledgement. RTTs are measured by a protocol that sends packets from the source to the destination and tracks their delivery. Although various existing programming frameworks [62] make the design of such a protocol very easy, the accuracy of such measurements is not very good due to noise sources such as the difference in synchronization between the sender and the receiver clocks [45] [63]. Rather than using the measured value directly, most traffic controllers use a prediction of the future value; as the predicted value may give better results than the measured value [64]. Hence, modifications are made to the classical estimation of RTT, the accuracy of which is a key factor in achieving timely change of transmission rate during a TFRC real-time streaming session.

TFRC operation wastes bandwidth because it is slow to probe for extra bandwidth by increasing its rate (aggressiveness) and does not react fast enough to increased congestion by reducing its sending rate (responsiveness). Through simulations it is observed that the slow aggressiveness and responsiveness of the TFRC is a result of the low reaction speed of the classical TCP filter (EWMA). To address the problem, it is proposed that the classical TCP filter is replaced with more accurate filters. Two filters were used; an existing adaptive Kalman filter with a change detection scheme [65] [66] and a novel adaptive TCP filter derived from Gustafsson's work [67] that provides a more accurate estimation during drastic changes of a streaming session than the normal TCP filter.

While some studies have been carried out using adaptive filters to improve RTT estimation for TCP traffic [65], this chapter emphasizes the aspect of improving rate adaptation when multimedia is being streamed. Through comparison of the jitter, delay and throughput

performance when using the traditional TCP estimator, it is demonstrated that adaptive filters enhance the aggressiveness and responsiveness of TCP-friendly rate control when network conditions change drastically, with the adaptive Kalman filter realizing the best results [68] [69].

3.2 TFRC Rate Adaptation

TFRC uses packet loss metrics as well as RTT to compute the acceptable rate at which to transmit video. It is based on an equation that models the amount of traffic consumed by a TCP flow when subjected to varying packet loss conditions. The general equation is such that available bandwidth T in bytes/sec is given by Equation 2.6.

An exponential weighted moving average of the packet loss rate is used to minimize the effect of a spurious loss event on the overall calculated value. The RTT is the other key factor in the computation of the sending rate. The most common implementation of RTT estimation is given in Equation 3.1,

$$\hat{x}_k = \alpha \hat{x}_{k-1} + (1-\alpha) RTT_{eff,k} \tag{3.1}$$

where $\hat{x}_{k-1}$ is the previous estimate of the RTT, $\hat{x}_k$ is the new computed value, $RTT_{eff,k}$ is the k^{th} sample of the effective RTT and α is a constant typically chosen between 0.84 and 0.9. This chapter utilizes RTT estimation filters as a means of enhancing rate adaptation for improved QoS of a streaming session.

3.3 Round-Trip Time Estimation

To implement rate adaptation, traffic-moderating mechanisms have been designed that operate by predicting the future value of the RTT. This approach reduces the possibility of having the estimated value being unduly skewed by noise in the system such as that caused by non-synchronous clocking at the source and receiver [45, 70] or by spikes in the current RTT value. The classical TCP filter, the proposed Kalman filter and the adaptive TCP filter are described briefly in this section.

3.3.1 TCP Filter

TCP (Reno) uses Equation 3.1 with α = 0.845 to estimate the RTT. Round-trip time estimates produced using the TCP filter present a number of issues, many of which are examined

in [45, 66]. In this chapter, the focus is on the inadequacy of the filter to quickly adapt to drastic changes in the network round-trip time, leading to bad estimates when such a phenomenon occurs.

3.3.2 Adaptive Kalman Filter

Several studies have been undertaken that utilize the Kalman filter to estimate bandwidth availability as well as round-trip time [65, 66]. An investigative study was carried out of six techniques for estimating RTT delay variation in a network [71]. It was found that the TCP-filter (referred to in the study as the Exponential Averaging Estimation Algorithm) and the Kalman filter provided the best performance. Of all the six algorithms, the Kalman filter was the only one that made provision for the effects of noise on the path or link. The Kalman algorithm displayed improved performance relative to statistical-based algorithms by having smaller variations between the successive estimation values, which results in smoother transitions. The Kalman filter uses a feedback mechanism to operate in both predictor and corrector modes. In the predictor mode, the emphasis is on predictor equations that are time-based, while the correction component is measurement-based. Through estimating the state of a process at a given time, noise measurements are used to achieve feedback control, hence the affirmation that this algorithm takes into account noise effects.

If the RTT is regarded as being composed of a smoothed desired RTT signal together with an additive noise component, then the equations that model the desired RTT as a noise observation of a constant exposed to step change in the mean are summarized in Equation 3.2 and Equation 3.3 below [65]:

$$x_k = x_{k-1} + \delta_k v_k \quad \delta_k \in \{0,1\} \tag{3.2}$$

$$RTT_{eff,k} = x_k + e_k \tag{3.3}$$

where the noisy characteristics of the RTT are captured by the measurement noise e_k of variance R_e. The step changes in the desired RTT x_k are modelled by process noise v_k with variance R_v and the discrete variable δ_k. If a change occurs at time k, then $\delta_k = 1$; otherwise $\delta_k = 0$. To estimate the sequence δ_N of instances of changes, a cumulative sum [67] is used.

The algorithm for the computation of the estimates $\hat{x}$ of the desired RTT can be summarized as follows. For the predictor stage, the process current state $\hat{x}_k$ and covariance error P_k predictions are respectively

$$\hat{x}_k = \hat{x}_{k-1} + K_k\left(RTT_{eff,k} - \hat{x}_{k-1}\right) \tag{3.4}$$

$$P_k = (1 - K_k)P_{k-1} + \delta_{k-1}R_v \tag{3.5}$$

where K_k is known as the Kalman constant. The equation for the corrector stage is as follows:

$$K_k = \frac{P_{k-1}}{P_{k-1} - R_e} \tag{3.6}$$

as well as there being an equation to track changes in the RTT:

$$g_k = max\left(g_{k-1} + \left(RTT_{eff,k} - \hat{x}_k\right) - \xi, 0\right) \tag{3.7}$$

If $(g_k > h)$, where h is an alarm threshold value, then the alarm is activated and δ_k set to one and g_k to zero. Otherwise δ_k is set to zero.

The filter has two design parameters, the negative drift ξ and the alarm threshold h. These parameters play a critical role in determining the precision and accuracy with which the adaptive Kalman filter operates.

3.3.3 Adaptive TCP Filter

Despite the simplicity of the adaptive Kalman filter algorithm, its implementation poses several challenging issues [67]. If the initial values of the variance of the measurement process and that of the Kalman constant are not set properly, Equation 3.6 may lead to floating point exceptions. To avoid the latter problem and provide the TCP filter with the capability of detecting sudden changes in the RTT, the TCP filter is enhanced with a change detection scheme similar to that of Kalman.

The key ideas of the scheme are derived from the CUSUM recursive least mean (RLS) algorithm of Gustafsson [67] and the resulting filter is named ‘adaptive TCP Filter’ (ATCP). The algorithm of the filter is as follows:

$$\hat{x}_k = \alpha \hat{x}_{k-1} + (1-\alpha) RTT_{eff,k} \tag{3.8}$$

$$\varepsilon_k = RTT_{eff,k} - RTT_{k-1} \tag{3.9}$$

$$S_k^{(1)} = \varepsilon_k \tag{3.10}$$

$$S_k^{(2)} = -\varepsilon_k \tag{3.11}$$

$$g_k^{(1)} = max\left(g_{k-1}^{(1)} + S_k^{(1)} - \xi, 0\right) \tag{3.12}$$

$$g_k^{(2)} = max\left(g_{k-1}^{(2)} + S_k^2 - \xi, 0\right) \tag{3.13}$$

There is an alarm if $g_k^{(1)} > h$ or $g_k^{(1)} > h$. After an alarm reset, $g_k^{(1)} = 0$, $g_k^{(2)} = 0$ and $\hat{x}_k = RTT_{eff,k}$. The point of the reset $\hat{x}_k = RTT_{eff,k}$ is that the algorithm forgets all previous information instantaneously, while at the same time avoiding bias and a transient. However, as was the case for the Kalman filter, the ATCP also presents the difficulty of the online determination of the drift and alarm threshold values.

3.4 Simulation Setup and Results

3.4.1 Simulation Setup

The simulation was conducted using the network simulator NS2 [72]. The network topology used is shown in Figure 3.1. The network nodes include one multimedia server feeding two multicast receivers over a bottleneck link of 0.5 Mbps with a propagation delay of 10 ms. The bottleneck link is also shared by four TCP flows. All the access links are set to 10 Mbps with a propagation delay of 10 ms. Links Router0→Router3, Router3→Router2 and Router2→Router1 have the same bandwidth and delay characteristics as the bottleneck link Router0→Router1. It should be noted that the adaptive Kalman filter and adaptive TCP filter in this simulation are applied only to the video flow and not the background TCP flows.

From time 0 s the video traffic flows via the link Router0→Router1 to the different destinations. At time 250 s the link goes down, and the traffic is automatically routed along the path Router0→Router3→Router2→Router1. The link Router0→Router1 is restored at time 350 s, at which point the traffic flow reverts to this initial link since NS2 routes traffic through the shortest link. A sudden, significant change in end-to-end delay is thus achieved.

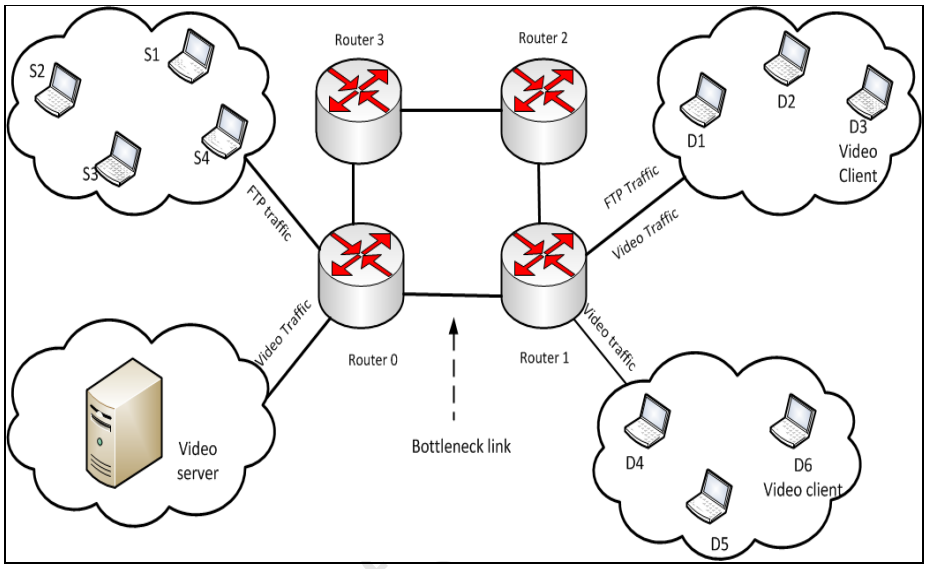

Figure 3.1: Network Topology

The implementation of NS2 used in the simulation is enhanced with RTP/RTCP as described in [73]. This implementation not only replicates more closely the RTP/RTCP actual behaviour and reporting mechanisms, but also facilitates TCP-friendly transmission of video streams. The streaming server used in the simulation is equipped with TCP-friendly rate adaptation capabilities, adjusting its transmission rate based on information gleaned from the RTCP receiver reports. The RTP traffic is set as 50 kbps at the outset, although this later adjusts dynamically, depending on the packet losses and the round-trip time along the path from server to receivers. The RTCP reports are generated every five seconds, providing the parameters used to compute the appropriate transmission rate at the source. Values of jitter, throughput and delay for the RTP traffic are obtained and used to assess the performance of the adaptive Kalman and adaptive TCP filters for rate adaptation relative to the TCP filter. For the Kalman filter, the parameters are set as in [65], with a drift value of 0.005 and a threshold of 0.05. The same values are used for the adaptive TCP filter.

3.4.2 RTT Results

The simulation is run for 500 seconds and four sets of RTT values are collected. One set is the actual measured RTT for the video traffic, derived from the RTCP reports. The RTT values when the TCP filter is implemented, the adaptive TCP filter values plus the corresponding values when deploying the Kalman filter are illustrated in Figures 3.2, 3.3 and 3.4.

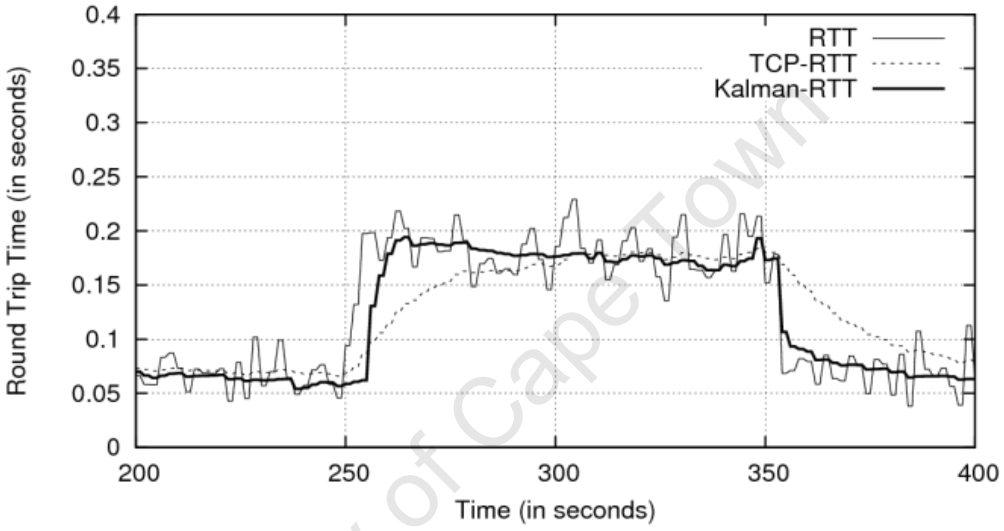

Figure 3.2: Measured, TCP-filter and Kalman-filter RTT values

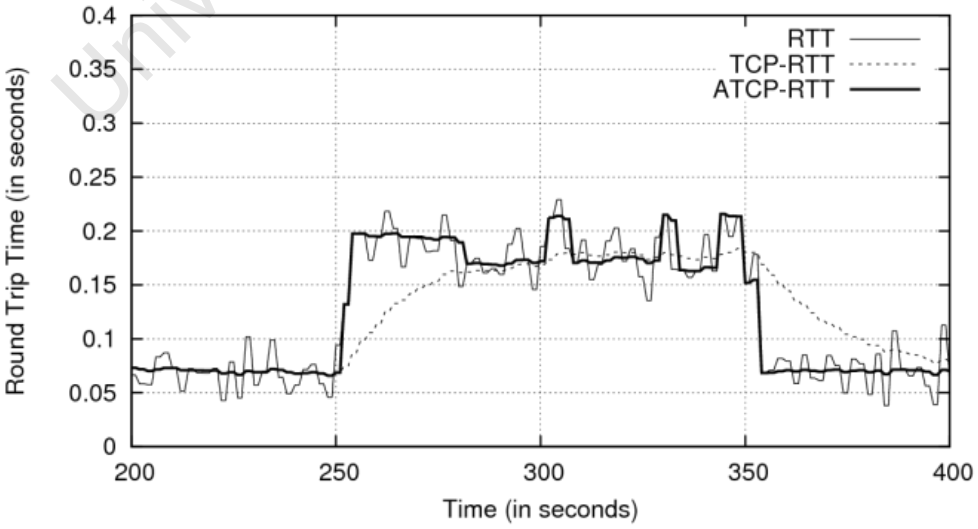

Figure 3.3: Measured, TCP-filter and Adaptive TCP-filter RTT values

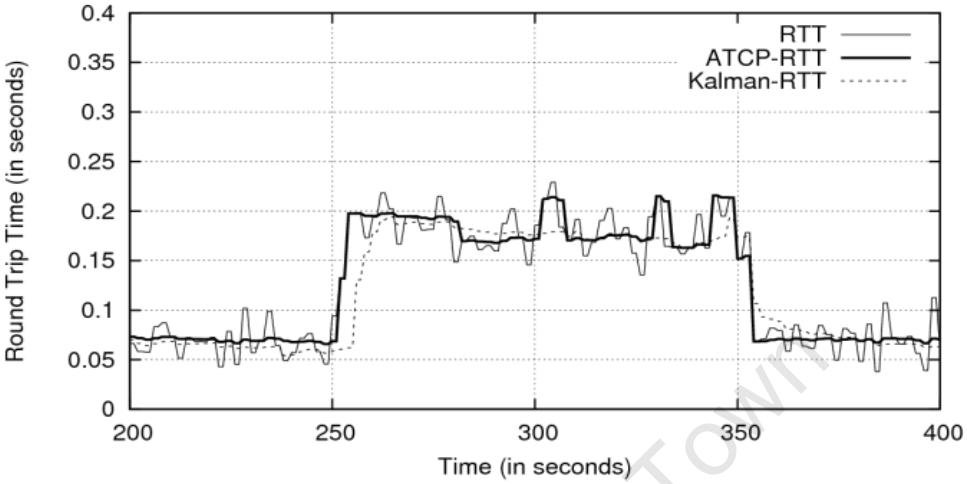

Figure 3.4: Measured, Adaptive TCP-filter and Kalman-filter RTT values

The plots illustrate the rapidly fluctuating values of the measured RTT that are typical of the end-to-end delays encountered on a real network. The estimated RTT values produced when the TCP filter is deployed display smoother value transitions as illustrated in Figures 3.2 and 3.3, reflecting the weighting effect of the underlying algorithm that assigns lower significance to more recently measured values. The Kalman filter is found to exhibit the smoothest characteristics as shown in Figure 3.2, which can be attributed to its error correction capabilities. This however occurs at the expense of precision of RTT estimation, reflecting the trade-off between smoothness and precision.

Furthermore, the adaptive Kalman and adaptive TCP filters are found to react more rapidly than the TCP filter to sudden changes of the RTT value, reverting faster to tracking closely the measured RTT value, as occurs at simulation time 250 and 350 seconds in Figures 3.2 and 3.3. A comparison of the two adaptive filters, as illustrated in Figure 3.4, shows that the adaptive TCP filter reacts much faster than the adaptive Kalman to sudden changes in the RTT but also introduces more fluctuations in its RTT estimates.

Increasing the drift for the adaptive Kalman filter produces smoother output and results in higher throughput of the RTP traffic. The higher throughput can be attributed to the

associated less conservative RTT estimates, which are interpreted by the rate adapting scheme as being indicative of low congestion and therefore permission for higher bit rates at the server.

3.4.3 Jitter Results

The Kalman filter displays slightly better performance with respect to jitter and delay than the TCP filter. The average delay experienced by the video flow packets is 1.494 s, the corresponding figure in the case of the Kalman filter and the adaptive TCP filter being respectively 1.432 s and 1.475 s respectively. Of even greater importance for real-time applications are the jitter values in the two scenarios. As mentioned in [74], applications being streamed in real-time are more affected by jitter than RTT. The average jitter for the Kalman prediction filter was 2.678 ms, better than the 2.859 ms and 3.815 ms average experienced by the RTP traffic deploying the adaptive TCP filter and the TCP filter, respectively. The jitter is computed as shown in Equation 3.14,

$$jitter = \frac{\left|\left(R_{j+1} - R_j\right) - \left(R_j - R_{j-1}\right)\right|}{16} \tag{3.14}$$

where R_x denotes the arrival time of packet x at the receiver, jitter thus being the difference in end-to-end delay for successive packets arriving at the client.

3.4.4 Throughput Results

In the simulation, at time 250 s the link Router0→Router1 goes down and traffic is then routed via the path Router0→Router3→Router2→Router1 which has a higher RTT. An accurate adaptive TFRC model should react fast to that change by decreasing its throughput as can be determined from Equation 3.6. When route Router0→Router3→Router2→Router1 goes down and traffic is restored to link Router0→Router1 as occurs at simulation time 350 s, the RTT decreases and TFRC should react fast by increasing the throughput.

Figures 3.5 and 3.6 show the throughput for the RTP stream and one of the background TCP flows for the adaptive TCP filter and Kalman filter, respectively. The average throughput for the Kalman case was 730.297 kbps, higher than the 708.108 kbps and 698.613 kbps value obtained when using the adaptive TCP filter and the classical TCP filter, respectively.

The 31 kbps difference in throughput for the Kalman filter relative to the classical TCP filter represents a 5% gain in throughput, while the adaptive TCP filter shows a 1% throughput gain relative to the classical TCP filter. It can be seen from the throughput flows that the video flow using adaptive filters conserves bandwidth, varying in much the same fashion as the TCP flow, but fluctuating more smoothly than the latter.

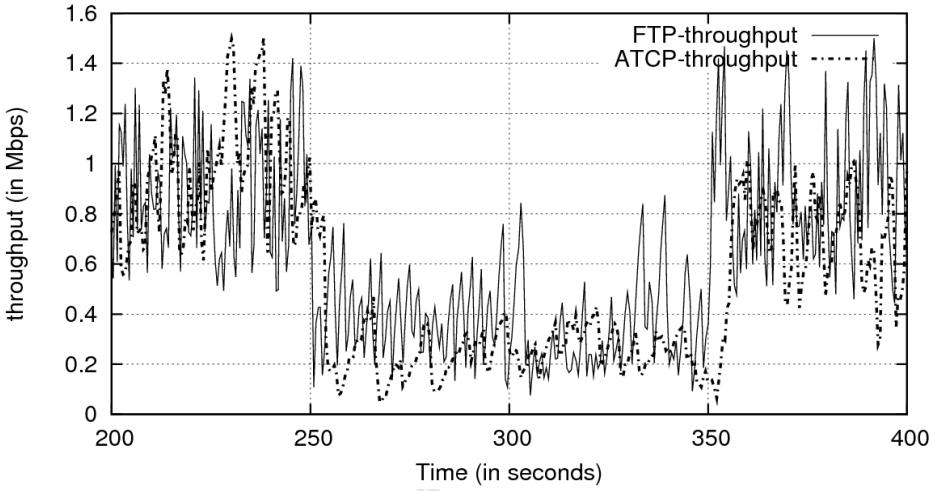

Figure 3.5: Video throughput of the ATCP filter

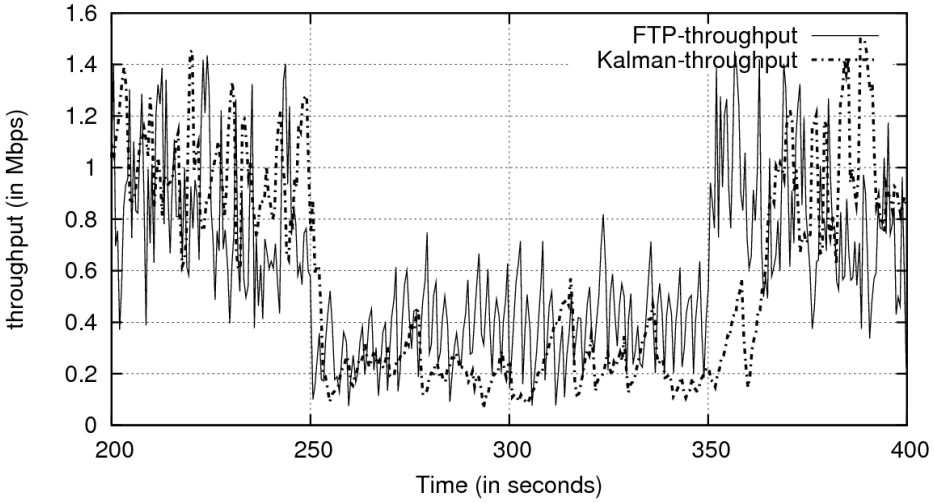

Figure 3.6: Video throughput of the Kalman filter

In Figures 3.7, 3.8 and 3.9 the video traffic instantaneous throughput for the Kalman, adaptive TCP and classical TCP filters are compared against each other.

The results in Figures 3.7 and 3.8 show that the TFRC scheme with the adaptive filters is faster in adapting its bit rate than TFRC with the classical TCP filter. This is a direct result of the change detection scheme that allows the adaptive filters to detect any changes in the network conditions much faster than the TCP filter. At time 250 s for instance, as the bottleneck link goes down the TFRC algorithms that use adaptive filters substantially reduce their bit rate which results in lower throughput values observed almost immediately after simulation time 250s.

As soon as the network capacity improves, as is the case at time 350 s, the TFRC flows enhanced with adaptive filters once again react much faster than the TCP-filter in increasing their bit rate. As a result a much higher throughput is observed almost immediately after simulation time 350 s. The classical TFRC, i.e. with TCP-filter, is slow in detecting this change and therefore adapts its rate slowly, hampered by the lower reaction speed of the TCP-filter.

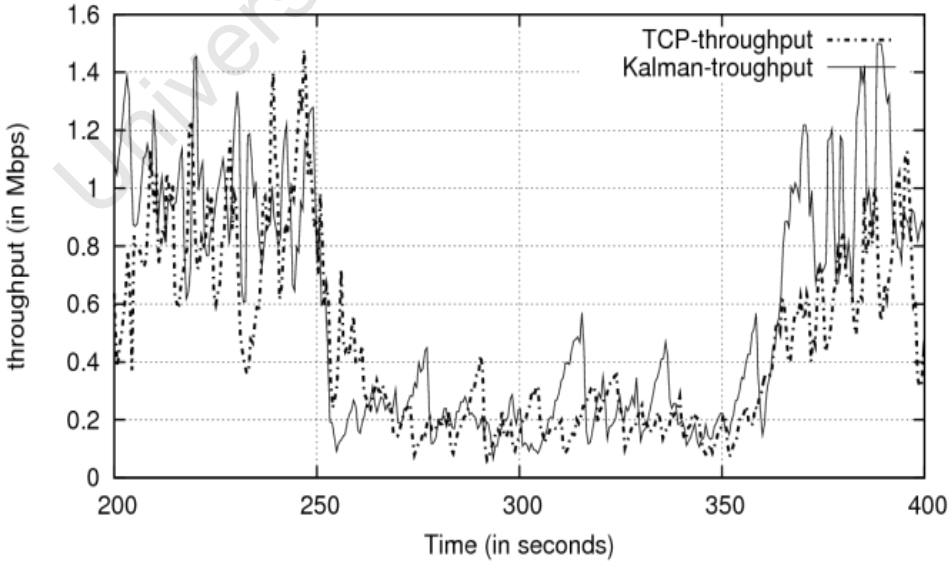

Figure 3.7: Video throughput for TCP and Kalman filters

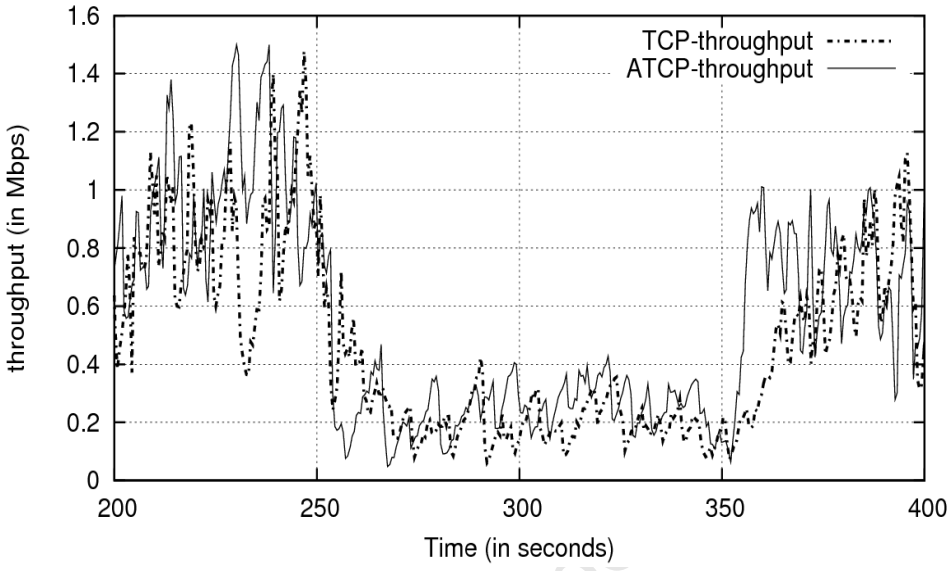

Figure 3.8: Video throughput for TCP and ATCP filters

Figure 3.9 shows that the TFRC with the adaptive TCP filter is the fastest in detecting the improvement in the network capacity, but thereafter the Kalman filter performs better, giving higher throughput values. The implication is that this is due to the Kalman filter having the lowest average RTT values of all the three filters, resulting in higher instantaneous throughput values achieved.

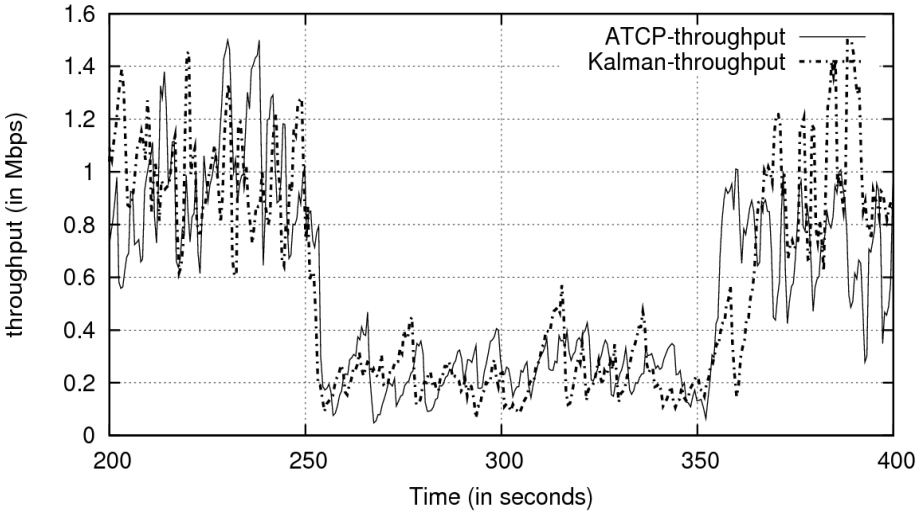

Figure 3.9: Video throughput for ATCP and Kalman filters

3.5 Composite Filters

Despite their simplicity and performance improvement, adaptive filters with CUSUM changes detection present the difficult of the online determination of the negative drift and alarm threshold. These parameters are critical however in their performance achievement and, if not set properly may render their performance ineffective. In what follows, a simple mechanism that combines two filters *Expanding Memory Polynomial* (EMP) and *Fading Memory Polynomial* (FMP) [75] to predict a future value of the round-trip time from previously recorded values is proposed as an alternative to adaptive filters with CUSUM changes detection.

3.5.1 Theory

EMP and FMP filters are based on the assumption that the input to the filter is a sequence of scalar samples obtained using a constant sampling interval, the error in these values does not show a high variability and smoothing is done by fitting a least squares polynomial. In what follows the following symbols are used: n to indicate the most recent or now, t the time and t_n the most recent time instant, y_n the scalar input, τ the constant sampling interval and m the degree of the fitting polynomial. A polynomial filter model of degree m is used, and estimation is based on least squares.

3.5.2 Model

The round-trip time $x(t)$ is observed and measured from the packet acknowledgement (ACK's) or real-time control protocol (RTCP) receiver report resulting in a sequence of y_n of raw RTT. y_n includes delays caused by transient effects in the network (attributed to short live cross-traffic). The short lived duration of these flows means that their contribution to the RTT can be considered as noise from the point of view of congestion control. It is thus necessary to filter them out [63] [64]. Thus, the model

$$y_n = x_n + v_n \tag{3.15}$$

in which ν_n is the measurement noise is obtained. The model, that is, y_n is then input to the 1-step polynomial filter, resulting in the following transposed vector

$$\hat{Z}_{n+1,n} = \left(\hat{x}, \tau\dot{\hat{x}}, \tau^2/2!\,\ddot{\hat{x}}, \cdots, \tau^m/m!\,D^m\hat{x}\right)^T_{n+1,n} \tag{3.16}$$

where, the two subscripts appearing in $\hat{Z}_{n+1,n}$ have the following meaning. The first subscript signifies that the validity instant of this state vector is t_{n+1} and the second subscript signifies that $\hat{Z}$ is an output that was based on observations that were made up to, and including, time t_n. The vector $\hat{Z}$ having this structure is called the normalized vector and must be de-normalized in order to obtain the desired estimate $\hat{x}(t)$ of the process $x(t)$. If

$$\hat{X}_{n+1,n} = \left(\hat{x}_{n+1,n}, \dot{\hat{x}}_{n+1,n}, \ddot{\hat{x}}_{n+1,n} \cdots, D^m\hat{x}_{n+1,n}\right) \tag{3.17}$$

is the state vector of the estimate, then the general de-normalization equation for the 1-step predictor is given by

$$\hat{X}_{n+1,n} = D(\tau)\hat{Z}_{n+1,n} \tag{3.18}$$

in which $D(\tau)$ is a diagonal matrix whose elements are given by

$$D(\tau)_{i,i} = i!/\tau i \tag{3.19}$$

3.5.3 Basic Structure of the Algorithms

The basic structure of the 1-step-predictor EMP is illustrated using a third-degree expanding memory polynomial.

$$\begin{aligned}
e_n &= y_n - \hat{z}_{0_{n,n-1}} \\
\hat{z}_{3_{n+1,n}} &= \hat{z}_{3_{n,n-1}} + \delta e_n \\
\hat{z}_{2_{n+1,n}} &= \hat{z}_{2_{n,n-1}} + 3\hat{z}_{3_{n+1,n}} + \gamma e_n \\
\hat{z}_{1_{n+1,n}} &= \hat{z}_{1_{n,n-1}} + 2\hat{z}_{2_{n+1,n}} - 3\hat{z}_{3_{n+1,n}} + \beta e_n \\
\hat{z}_{0_{n+1,n}} &= \hat{z}_{0_{n,n-1}} + \hat{z}_{1_{n+1,n}} - \hat{z}_{2_{n+1,n}} + \hat{z}_{3_{n+1,n}} + \alpha e_n
\end{aligned} \tag{3.20}$$

where

$$\begin{aligned} \delta &= 140(n+4)(n+3)(n+2)(n+1) \\ \gamma &= 120(2n+1)(n+4)(n+3)(n+2)(n+1) \\ \beta &= 20(6n^2+6n+5)(n+4)(n+3)(n+2)(n+1) \\ \alpha &= 8(2n^3+3n^2+7n+3)(n+4)(n+3)(n+2)(n+1) \end{aligned} \tag{3.21}$$

For maximum speed and to save on the required RAM, the algorithm will actually be implemented in computer code as follows.

$$\begin{aligned} e &= y - \hat{z}_0 \\ \hat{z}_3 &= \hat{z}_3 + \delta e \\ \hat{z}_2 &= \hat{z}_2 + 3\hat{z}_3 + \gamma e \\ \hat{z}_1 &= \hat{z}_1 + 2\hat{z}_2 - 3\hat{z}_3 + \beta e \\ \hat{z}_0 &= \hat{z}_0 + \hat{z}_1 - \hat{z}_2 + \hat{z}_3 + \alpha e \end{aligned} \tag{3.22}$$

The 1-step-predictor FMP has precisely the same structure and there is no need to repeat its equations. The only difference between the EMP and FMP algorithms is the expressions that are used for the filter weights δ, γ, β, α; which for an FMP of degree-3 are set as follows.

$$\begin{aligned} \delta &= \frac{1}{6}(1-\theta)^4 \\ \gamma &= (1-\theta)^3(1+\theta) \\ \beta &= \frac{1}{6}(1-\theta)^2(11+14\theta+11\theta^2) \\ \alpha &= 1-\theta^4 \end{aligned} \tag{3.23}$$

Thus, it is possible to switch seamlessly from EMP to FMP by simply changing the weights, to create the composite EMP/FMP filters.

3.5.4 General Observation

EMP algorithms are based on fitting a polynomial by least squares. All observations incorporated into the filter play an equal role in the determination of the estimate. However the least squares procedure in the case of FMP algorithms uses a fading memory, which means the most recent observation is weighted by one and the information preceding it by successively smaller, exponentially-decaying values. The variable *n* does not appear in the FMP filter weights *δ, γ, β, α* as it does in the EMP filters. Instead the FMP filter contains what is called the fading memory parameter *θ*. This is a real number selected by the user that lies in the range $0 < \theta < 1$

and controls the properties of the filter by controlling the exponential rate of decay of its memory. If an m^{th}-degree EMP filter is smoothing data that comes from a process that is a polynomial of degree *m* or less, then the filter model will be perfectly matched to the process. However if the process is not a polynomial of degree *m* or less, then the EMP will initially track the process with acceptable errors but later will develop unacceptable large errors. On the other hand, FMP filter can track any arbitrary process. The EMP filters are completely self-initializing, something which is not true for the FMP filter [75] [76]. This property makes the EMP filters valuable in a number of ways. For example, FMP filters must be properly initialized if start-up transients that make their outputs completely useless have to be avoided; EMP filters can therefore be used to initialize the FMP filter.

3.5.5 The composite EMP/FMP Filters

The properties of the two filters EMP and FMP make them an ideal combination, and together they form what we call the composite EMP/FMP filter shown in Figure 3.10. The operation of the composite filter is as follows:

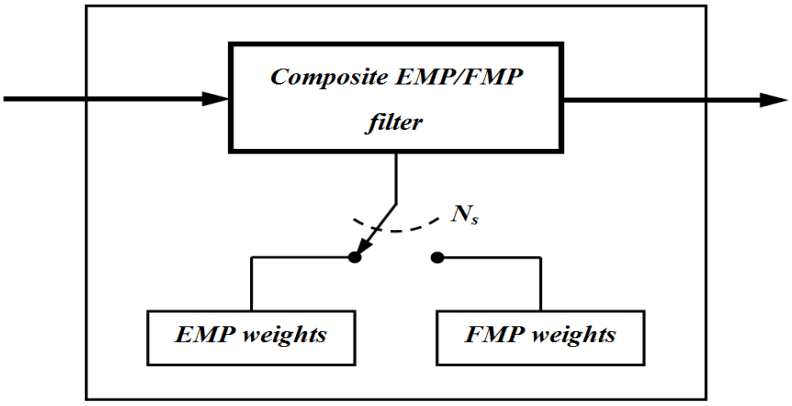

Figure 3.10: Composite EMP/FMP Filter

- At start-up the switch is set to use the EMP weights $(\alpha, \beta, \gamma, \cdots)$, and its properties are those of self-initializing EMP.

- At some point the switch is moved to the FMP weights, and its properties then become those of the FMP filter. In this way the composite EMP/FMP filter gets the best of both worlds, from the
- EMP filters that are self-initializing but cannot track an arbitrary process indefinitely and from the
- FMP filters that are not self-initializing but can track an arbitrary process indefinitely.

If the EMP switch number N_s is properly selected then the move from EMP to FMP takes place seamlessly and without transients. The question that arises then is how the switch number N_s should be selected.

Assume that the EMP filter has been running from $n = 0$, and also a hypothetical transient free FMP filter has been running in the background. At a certain point in time, a comparison is made between the foreground EMP and background FMP filters. N_s is selected such that the outputs of the two filters contain similar errors. Table 3.1 gives a summary of N_s for polynomial filters up to degree 3; details on how this table is derived are behind the scope of this discussion, interested readers should refer to [76].

Table 3.1 Polynomial Degree vs Switching Number N_s

Degree	1	2	3
N_s	$2/(1-\theta)$	$3.2/(1-\theta)$	$5.51/(1-\theta)$

3.5.6 Performance Evaluation

The performance evaluation of the composite filter is conducted using the ping protocol and the open source Live555 [77]. Results of the composite filter are compared with the classical TCP filter and the adaptive Kalman, as the latter realised a much better performance with respect to rate adaptation than the adaptive-TCP filter.

It is important to mention that during the experimentation, it is observed that an EMP filter of degree m develops smaller errors in the presence of spikes in the RTT process than its

corresponding FMP filter, i.e., FMP of the same degree. Thus, the results illustrated below are obtained from an enhanced implementation of the composite filter, with a mechanism for the filter to switch not only from the EMP to FMP but also from the FMP to EMP by choosing the filter that has a minimum error in the estimates.

3.5.6.1 Ping Results

In the first experiment, the ping protocol is used to evaluate the performance of the composite filter. Ping protocol is often used for testing remote hosts and for generating and collecting RTTs. Ping protocol approach for collecting RTTs consists of probing destination hosts with 64 bytes ICMP (*Internet Control Message Protocol*) Echo. On reception of ICMP packets, the destination responds with an ICMP Echo Reply to the ICMP sender, and from the sender clock using the ICMP Echo Reply timestamps, the RTT is obtained. To satisfy the composite filter's hypothesis, the ping protocol in this experiment is configured to send the probing packet at a constant time interval. The benchmark for the performance evaluation is a *Local Area Network* (LAN) consisting of a server located at the department of Electrical Engineering of the University of Cape Town in South-Africa and a client located at the department of Mathematics of the same university. The server is pinged during working hours and busy hours to emulate conditions of a busy network. Figures 3.11 and 3.12 present the ping results for the Kalman and the composite filters of degree-0. The green line represents the measured RTT, the blue line the TCP filter, and the red line in Figure 3.11 the Kalman filter and in Figure 3.12 the composite filter of degree-0.

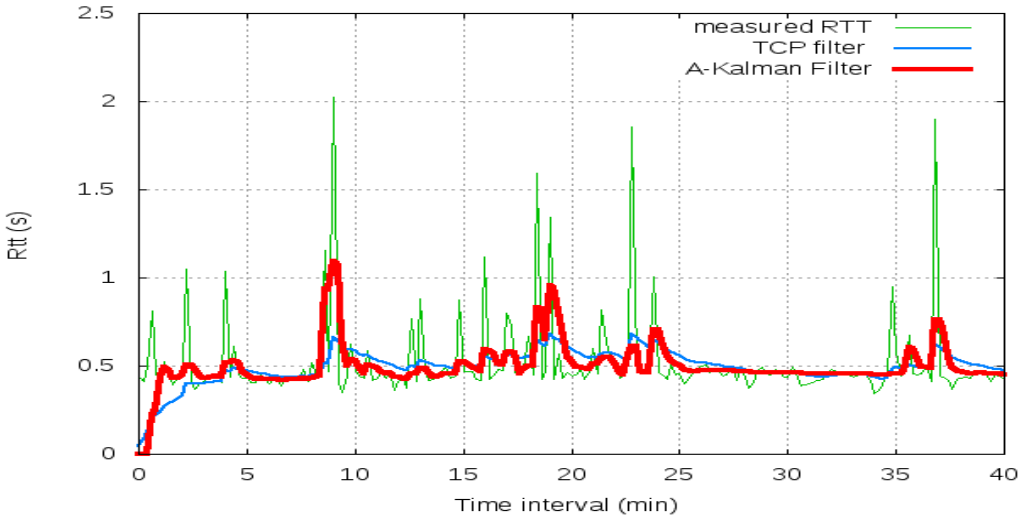

Figure 3.11: Measured RTT vs prediction using both TCP and Kalman filters

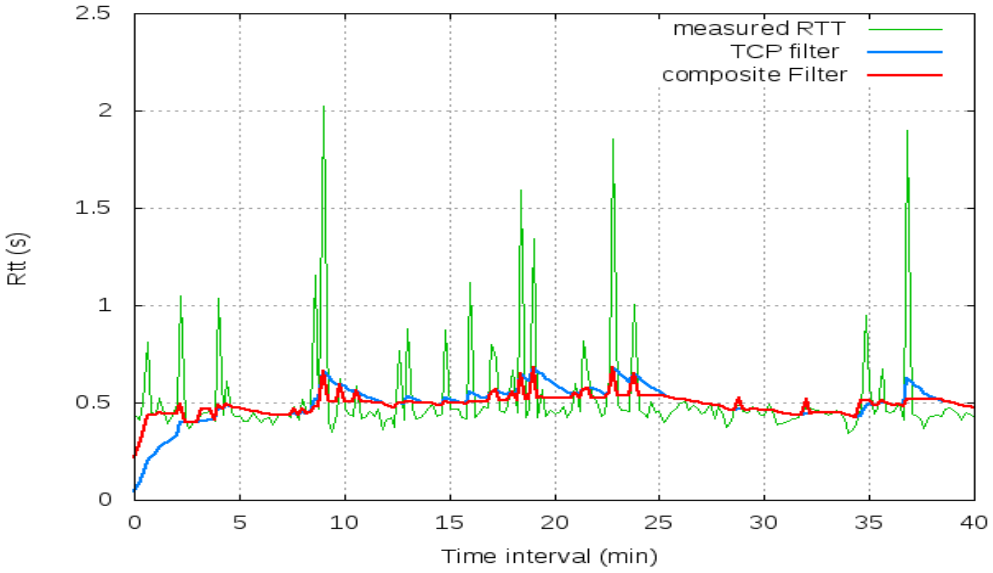

Figure 3.12: Measured RTT vs prediction using both TCP and proposed composite filter

From the plots it can be observed that both filters outperform the TCP filter in terms of initial estimates and fast recovery from spikes. The Kalman filter missing detection problem can be observed in Figure 3.12 when the filter completely misses the spike at point 35. The missing problem is also observed in the composite filter with the spikes at point 37.5 completely missed. The advantage of the composite filter is that the detection mechanism does not depend on any filter parameters making its implementation much easier. For the adaptive Kalman filter, the parameters need to be set appropriately otherwise false alarms and missing detection will aggravate and result in inaccurate RTT estimations.

3.5.6.2 Live555 Results

In the second experiment the focus is more on the robustness of the composite filter to normal video packet size. The experiment is conducted using the open source Live555 which is configured to segment video data into packets of 1500 bytes of length. The benchmark for the performance evaluation is a LAN consisting of a multimedia server and a multimedia client as

illustrated in Figure 3.13. The video is stored at the video server and streamed from there to the receiver, located at terminal node. Traffic from the video server to the client is routed through a single router. The router may route other types of traffic during busy hours.

To predict round-trip time, Live555 is extended to support the three following filters: TCP, composite filter of degree-0 and composite filter of degree 3. The value of θ is set to 0.9 for the TCP filter and the FMP part of the composite filter of degree-0; while $\theta = 1/(n+1)$ for the EMP component of the composite filter of degree-0. The composite filter of degree-3 is implemented as described in section 3.5.3 with $\theta = 0.9$. To facilitate extraction of the receiver report contents, Live555 is set to the debugging mode. Each time that a receiver has to generate a report, it first computes the RTT and other network metrics as described in RFC3550. The computed value of the RTT is then input into the above three filters, to predict the next possible value of the RTT. The objective of the TCP and the composite EMP/FMP filter of degree-0 is to predict the future value of the average RTT while the composite EMP/FMP filter of degree-3 is used to predict the instant value of the RTT. The results obtained are presented in the following section.

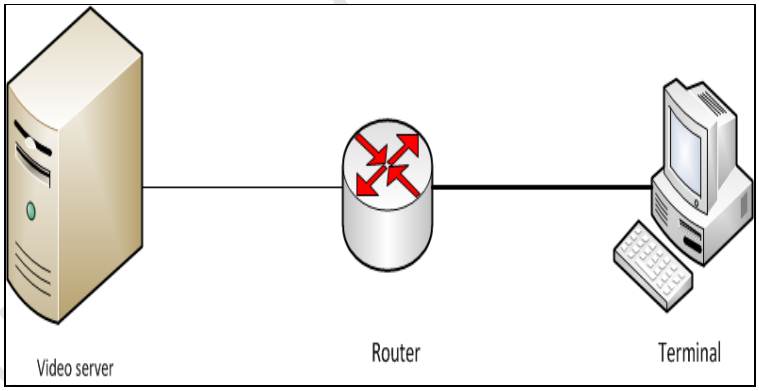

Figure 3.13: Topology of the experimental setup

Figure 3.14 and Table 3.2 highlight the start-up behaviour of the TCP filter (estimator). The dotted line represents the TCP filter, the continuous line the measured RTT and the continuous bold line the composite filter of degree-0. As expected the TCP filter takes around 35s to predict acceptable value of the average RTT. The estimates from t = 0 s to t < 35 s are completely wrong; after t = 35 s TCP filter shows an improvement on its predicted value. On the

other hand, the composite EMP/FMP filter shows much better estimation at its start-up. At t = 35s the composite EMP/FMP filter switches seamlessly to the FMP (TCP) but after t = 35 s, the initialization effect disappears and both filters show identical values. As already pointed out, the implementation of the composite filter not only switches from the EMP to FMP but also from the FMP to EMP. It is observed that this switching mechanism provides the filter the capability of quickly recovering from a sudden change in the average RTT. This is illustrated in Figure 3.14 around time t = 200 s where the composite filter adapts much faster than TCP filter in change in average RTT.

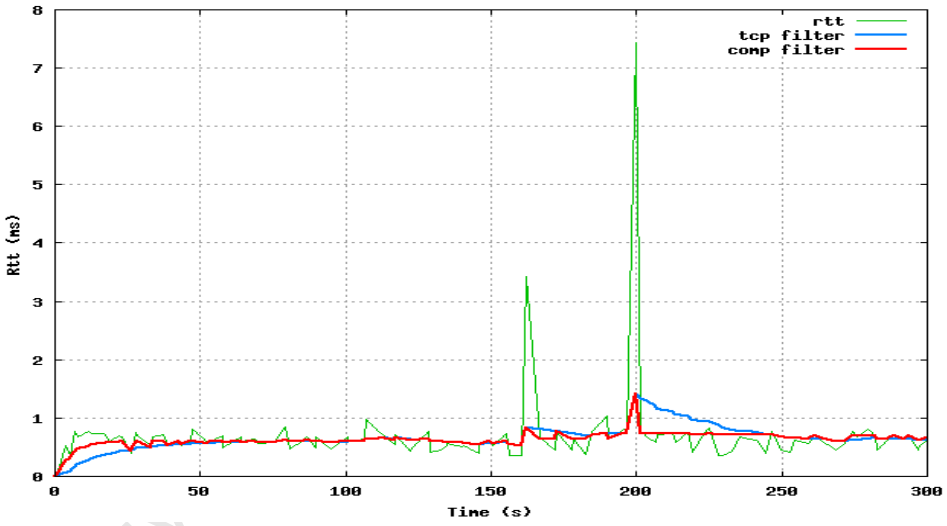

Figure 3.14: Measured RTT vs prediction using both TCP and proposed composite filter

Table 3.2 Initial Value of the Composite EMP/FMP and TCP Filter when Estimating RTT

k	x_k	$x^*_{k,TCP}$	$x^*_{k,EMP/FMP}$	$error_{TCP}$	$error_{EMP/FMP}$
1	0.00000	0.00000	0.00000	0	0
2	0.00052	0.0000519	0.0002594	0.0004671	7.1e-05

3	0.00037	0.0000833	0.0002950	0.0002827	0.000260
4	0.00076	0.00001513	0.0004120	0.0006117	0.0003510
5	0.00067	0.00002033	0.0004639	0.0004677	0.0002071
6	0.00078	0.00002608	0.0005163	0.0005172	0.0002617
7	0.00073	0.00003079	0.0005471	0.0004241	0.0001849
8	0.00073	0.00003504	0.0005703	0.0003816	0.000162
9	0.00061	0.00003764	0.0005748	0.0002336	3.525e-05

In Figure 3.15, the performance of the composite filter of degree-3 is illustrated. The dotted line represents the predicted values and the continuous line the measured RTT. From the plot, it can be observed that the composite filter of degree-3 follows nicely the behaviour of the round-trip time in absence of spikes. When there are appearances of spikes, the filter develops large errors, as shown in Figure 3.16, but will finally recover and track the measured RTT with acceptable errors.

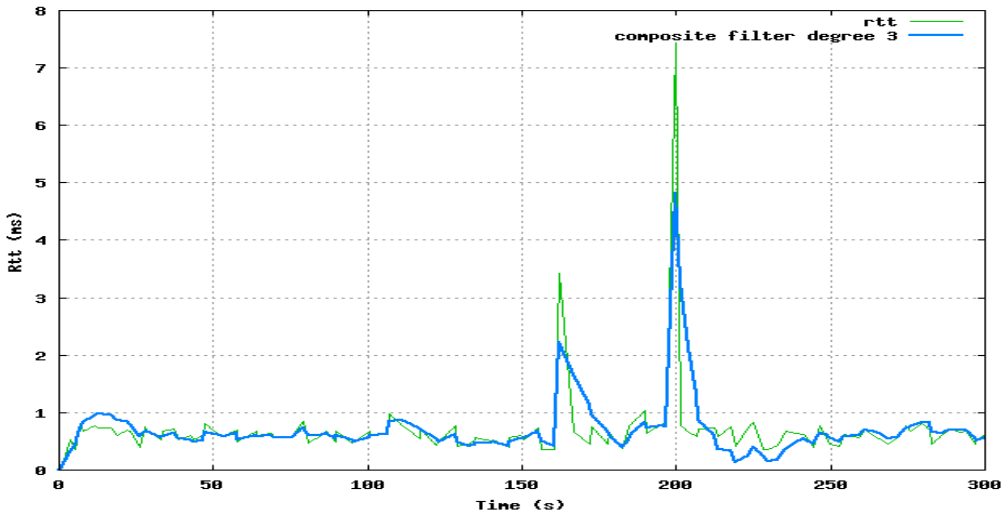

Figure 3.15: Measured RTT vs composite filter of degree-3

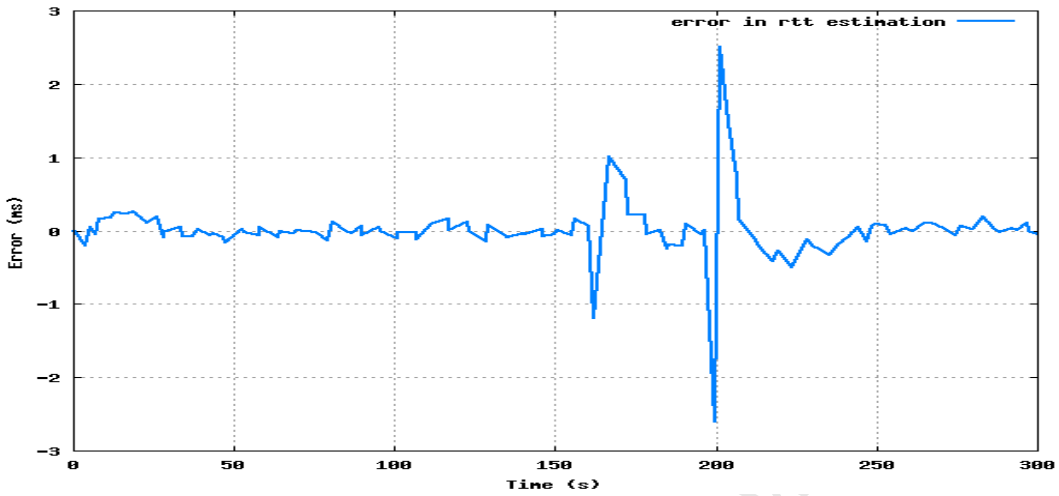

Figure 3.16: Error when replacing RTT by the estimated values

3.6 Summary

In this chapter, an investigation of the capability of the adaptive TCP and adaptive Kalman filters to provide more responsive and aggressive rate adaptation for real-time flows as network conditions suddenly change. The video traffic, despite flowing over UDP, is adapted to display TCP-friendly bandwidth utilization characteristics. Thereafter the Kalman filter and an adaptive TCP filter are incorporated into the rate adaptation model for the video flow, with the goal of achieving more precise estimation of RTT and hence variation of the sending rate at the server. It is illustrated that when there is a sudden change in network traffic levels, such as due to the collapses of a section of the streaming path and therefore traffic diversion to a new route, the adaptive filters respond more quickly and adeptly to the change, hence creating better responses by the source. The adaptive Kalman filter generally realizes much better results than the adaptive TCP filter.

However, adaptive filters rely on some parameters that are difficult to setup given the dynamic nature of the Internet. Thus the composite EMP/FMP filters are discussed in the second part of the chapter. These filters capture the RTT model dynamics and nonlinearities, and represented it in a recursive way that provides fast execution and less memory space, and also fast recovery to changes in RTT.

Chapter 4. RTP Rate-Based SIMD Protocol

This chapter describes the proposed unicast *RTP Rate-Based SIMD* protocol for video streaming over the Internet. RRB-SIMD runs on top of RTP and relies on the RTCP reports for control feedbacks. RRB-SIMD uses the Increase/Decrease rules of RB-SIMD to alter its transmission rate. The composite filter presented in Chapter 3, is used in the RRB-SIMD implementation to better capture the dynamic nature and non-linearity process of the RTT. The protocol operates in a TCP-friendly way towards TCP flows and includes in addition to packet loss ratio a control criterion that combines the cumulative jitter and the delay factor for incipient congestion detection prior to loss of a packet. With this, the transmission rate is adjusted in a way that anticipates loss of packets. Network parameters, such as packet loss ratio and RTT as well as the TCP-friendly share throughput, are computed at the receiver and the results sent back to the sender using the RTCP reports. Upon reception of the feedback information, the sender adjusts the sending rate. The performance evaluation results using both network related metrics and video quality measurement are also provided and obtained using the network simulator NS2.

4.1 Introduction

Video on the Internet can be streamed using either the transport protocol TCP or UDP. However, TCP, as stated earlier, has attributes such as retransmissions that have a negative effect on the video packet delays. Its congestion control mechanisms which are instigated by packet losses and a halving of the window size in response to packet loss, also negatively impacts the quality of the video. The transport protocol UDP which contains features such as low delay is also not entirely adequately fashioned for real-time applications. RTP was devised with the objective of facilitating audio and video transmission over the Internet. The RTP header contains fields such as sequence numbers that facilitate detection of packet loss and reordering along the end-to-end path between source and receiver. Although originally designed to work well for multicast groups on very large scales, its scope is not limited to that. More applications today use RTP for small multicast groups such as video conferences or even unicast such as *Video on Demand* (VoD) applications. RTP is further enhanced by its sister protocol RTCP, which provides quality of service functionality through its source and receiver reports. RTP is

commonly run over UDP and not TCP due to TCP's disadvantages towards multimedia transmission as described above.

However, given the lack of inbuilt congestion and flow control mechanisms within UDP and with due consideration that the majority of Internet traffic is still TCP-based, UDP video traffic are regulated in TCP-friendly manner so as to achieve fair bandwidth sharing towards TCP traffic. This is done at the application layer of the network stack using techniques such as rate adjustment. Rate adjustment as already pointed out, may handle congestion by adjusting the video transmission rate at the sender based on feedback reports from the receiver. TCP-friendly protocols such as TFRC, RB-SIMD and many others that were discussed in Chapter 2 or proposed in the literature are used to achieve this goal. Amongst these protocols, TFRC is by far the most popular and often used as the standard.

The TFRC protocol uses packet loss metrics as well as RTT to compute the acceptable rate at which to transmit video. It is based on the TCP response function (see Equation 2.6) that models the amount of traffic consumed by a TCP flow when subjected to varying packet loss conditions. TFRC is designed for applications that would prefer to maintain slowly-changing sending rate, while being responsive to network congestion over long time periods. TFRC has some known disadvantages, such as a very low speed reaction to change on the available network throughput or late reaction to congestion when a packet loss has already occurred [20] [69]. Standard TFRC is also not suitable for a wireless environment as it cannot distinguish loss due to congestion from that due to wireless channel impairment. These disadvantages have opened the door for new control protocols which can be suitable for video streaming over IP Networks. In this chapter, a unicast protocol that runs on top of RTP and called *RTP Rate-Based SIMD* (RRB-SIMD) is proposed to enhance video delivery over the Internet [78].

RRB-SIMD takes advantage of the RTCP reports to adjust the transmission rate. No modification is made to the RTP/RTCP protocol. The innovation in RRB-SIMD resides in the way the sender detects congestion and adjusts the transmission rate using the square increase multiplicative decrease rule of RB-SIMD. Since packet loss rate is not a reliable indicator of congestion [23], RRB-SIMD relies on the average of the cumulative jitter [79] [80] to detect incipient congestion prior to loss of a packet. Upon incipient congestion detection, the sender may decrease or increase the transmission. The action depends on the occupancy of the

bottleneck buffer on the path between the sender and the receiver which is assessed using the delay factor. A value of the delay factor greater than a specified high-threshold value is perceived by the sender as a severe increase in the network workload. Thus, the sender decreases the transmission rate smoothly to reduce the risk of buffers overflow at routers. Otherwise, transmission rate increases quadratically if the delay factor is lower than a specified low-threshold value or is maintained for a value of the delay factor between the two thresholds. The protocol is discussed in more detail in the following sections.

4.2 Protocol Design

RRB-SIMD is a unicast sender-based rate control protocol implemented on top of RTP and which takes advantage of RTCP reports to assess network congestion and adjusts the transmission rate. The protocol operates in a TCP-friendly way with TCP flows and adjusts the transmission rate in a smooth manner to preserve the user-perceived quality. The computation of the TCP-friendly shared transmission rate is performed at the receiver side. The receiver uses packet loss ratio as a control criterion for congestion detection. However, since packet loss is not a reliable indicator of congestion, RRB-SIMD includes a combined cumulative jitter and delay factor scheme to detect incipient congestion prior to the loss of a packet.

Since RTP was originally designed for large multicast sessions, any application that uses RTP for a unicast session, will suffer due to the timing rules of RTCP reports which restrict the reports to a minimum interval of five seconds between two RTCP feedbacks. As a result, the sender will not react in a timely manner to changes in the reception quality. RRB-SIMD increases its responsiveness through the quadratic increase of the transmission rate and the use of a cumulative jitter and delay factor scheme for congestion detection prior to loss of a packet.

The protocol is described in three stages which consist of the control variables used in the evaluation of the congestion level and computation of the share throughput, the decision and adaptation mechanism used during congestion avoidance and the connection start-up.

4.2.1 Control Variables

This section discusses the computation of control variables used in the evaluation of the network conditions and computation of the TCP-friendly transmission rate.

Considering a case where a packet index i is transmitted successfully, let S_i denote the RTP timestamp of packet i, R_i the arriving time in RTP timestamp units of packet i and $D_i = R_i - S_i$ the one way delay experienced by the packet.

A. Average Cumulative Jitter

The cumulative jitter is the amount of playback delay that must be provided for in order to avoid discarding delayed video frames at the client side [79] [81]. The cumulative jitter is correlated with the level of congestion at the bottleneck link of the path and is obtained as follows. The receiver computes the jitter using the algorithm defined in RFC 3550 [82], that is, for two sequentially packets indexed i and $i + 1$, the delay jitter J_i is expressed as follows.

$$J_i = D_{i+1} - D_i \tag{4.1}$$

The cumulative jitter CJ_k for a sequence of k jitter values, is then computed as,

$$CJ_k = J_1 + \cdots + J_k \tag{4.2}$$

The average of cumulative jitter A_k is then derived from a sequence of k cumulative jitter values as follows.

$$A_k = \frac{CJ_1 + \cdots + CJ_k}{k} \tag{4.3}$$

Finally, the receiver calculates two values of the average cumulative jitter:

- The average since the start of the session, referred to as long running cumulative average ($longAvg$) and
- The average measured between two successive RTCP reports referred to as a short running average ($shortAvg$).

B. Delay Factor

The delay factor d_f is a measure of the buffer occupancy at the bottleneck link of the path from the sender to the receiver. It is computed periodically on a round basis (a round being the time interval between two RTCP packets) and is obtained as follows.

For each received packet i, the receiver computes the delay D_i and updates the minimum delay D_{min} and the maximum delay D_{max}. From D_{min} and D_{max} the receiver derives the maximum queuing delay QD_{max} as follows.

$$QD_{max} = D_{max} - D_{min} \tag{4.4}$$

The queuing delay QD_i is a measure of the path congestion and is obtained as,

$$QD_i = D_i - D_{min} \tag{4.5}$$

The average value of queuing delay of all received packets within a round is then computed,

$$avgQD = \frac{\sum_{i=1}^{n} QD_i}{n} \tag{4.6}$$

where n is the number of packet received within a round. Finally, the receiver obtains the delay factor d_f as the ratio of the average queuing delay to the maximum queuing delay.

$$d_f = \frac{avgQD}{QD_{max}} \tag{4.7}$$

Values of the delay factor d_f range between [0, 1], with zero indicating an empty buffer and one a full buffer at the bottleneck link. Thus, it is possible to predefine threshold values that can be used to assess the network condition. In RRB-SIMD two threshold values are considered the low-threshold value T_1 and the high-threshold value T_2, both chosen in the range [0,1]. The two threshold values serve to classify the shared available bottleneck throughput into three regions, the non-congested region $[0, T_1]$, the loaded region $]T_1, T_2[$ and the congested region $]T_2, 1]$.

C. Packet Loss Measurement

It is important to notice that packet loss is unavoidable due to best-effort nature of the Internet along with other reasons, such as the difference between the time when a receiver report is generated and the time it is received at the sender side. This difference delays the rate adjustment and may cause excessive packet loss during network congestion. In RRB-SIMD, packet loss event rate is measured at the receiver using the RTP sequence number field. The receiver then minimises the effect that single spurious loss may have on the packet loss estimation, by smoothing the values of packet loss using Equation 4.8

$$\hat{l} = \frac{\sum_{i=0}^{n} w_i l_i}{\sum_{i=0}^{n} w_i} \tag{4.8}$$

where $\hat{l}$ is the smooth value of packet loss rate, l_i the measured packet loss rate and w_i the weight. Values of the weight are chosen such as a newer packet loss event has a high weight while the weights gradually decay to zero for older packet loss events. The following series $(1,1,1,1,0.8,0.6,0.4,0.2)$ of weight values are used in the implementation of RRB-SIMD, for $n = 8$. More details about this smoothing technique are provided in [24].

D. Round-Trip Time Estimation

The parameter K used in Equation 2.8 to compute the transmission rate r_t is inversely proportional to RTT [26]. Thus, for the receiver to compute the transmission rate when the network is not congested, he needs first to compute the RTT. RTT is computed as in [81], that is, using the RTP timestamp field and the arrival time of the packet at the receiver. Assuming that the arriving packet has the index i, first the receiver calculated the one way delay D_i to the sender and, computed the RTT using the one way delay as shown in Equation 4.9

$$RTT = (1 + \varphi)D_i \tag{4.9}$$

where the value of φ is unknown. To estimate φ, the receiver uses RTT_{eff} the effective estimation of RTT provided for in the RTCP reports. RTCP protocol requires the receiver to store in its reports to the sender the timestamp of the most recent report from the sender in the t_{LSR} field, and the delay between the reception of the last sender report and the transmission of

the receiver report in the t_{DLSR} field. With this information and the time A when the sender receives the receiver RTCP report, the sender computes the effective RTT_{eff} as follows.

$$RTT_{eff} = A - t_{LSR} - t_{DLSR} \tag{4.10}$$

The sender then computes an appropriate value for the parameter φ using Equation 4.11

$$\varphi = \frac{RTT_{eff}}{D_i} - 1 \tag{4.11}$$

To avoid the effect of instant RTT high value in the computation of the transmission rate, the receiver uses the composite filter of degree-0 (see Chapter 3) to smooth the RTT process, and thus, the oscillations in the computed transmission rate.

4.2.2 Decision and Adaptation Mechanism

During congestion avoidance, the receiver assesses the congestion level using a mechanism that combines the delay factor, short and long running cumulative average jitter and packet loss. The receiver first examines the packet loss rate event; if packet loss rate is greater than zero then the receiver concludes that the network is congested. Otherwise, the receiver undertakes further investigation using the short and long running average cumulative jitters, and the delay factor.

When the short running average is greater than the long running average, the network workload is interpreted as increasing and the receiver concludes that there is upcoming congestion (incipient congestion). RRB-SIMD reinforces the cumulative jitter scheme with the delay factor d_f. This is done to reduce the risk of unnecessary decrease of the transmission rate each time that incipient congestion is detected through the cumulative jitter scheme. When, a measure of d_f is substantially larger than the specified high-threshold value T_2, the receiver safely concludes that the network is congested and uses Equation 2.7 to compute the TCP-friendly share bandwidth. Otherwise, if the delay factor is lower than a specified low-threshold value T_1 ($T_1 < T_2$) the transmission rate is increased using the relationship in Equation 2.8 or maintained for a value of d_f between the two threshold values, that is, $T_1 < d_f < T_2$.

The receiver then feeds this information into the *Receiver Report* (RR) and uses the extension mechanism of RTP/RTCP [73] to communicate the transmission rate to the sender. The sender, upon reception of the receiver's RTCP reports, adapts the transmission rate. The algorithm is summarised in Figure 4.1.

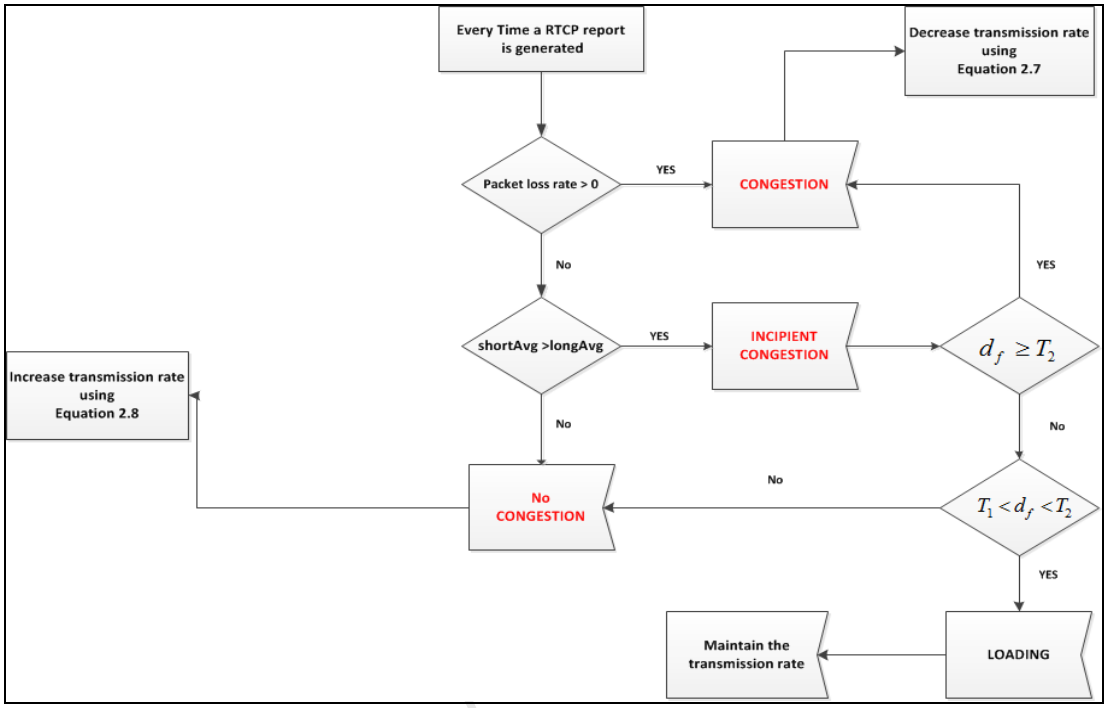

Figure 4.1: An Algorithm to implement rate adaptation

4.2.3 Connection Start-up

At the beginning of a connection, the sender starts with an initial rate r_i which is set to a very low value. The receiver, which has not experienced any packet loss, increases the rate by no more than one packet per RTT. For this reason, it calculates the new rate r_t

$$r_t = r_i + S/RTT \tag{4.12}$$

where S is the size of the packet. This process of increasing the rate by no more than one packet per round reduces the risk of bandwidth overshoot at the bottleneck link. Once a loss is registered the receiver uses Equation 2.7 to compute the transmission rate. RRB-SIMD then enters immediately into the congestion avoidance mode. During connection start-up, only packet

loss is considered as an indication of congestion and that the initial values of r_{max} and r_0 used during congestion avoidance are obtained at the end of this process.

4.3 Parameters Determination

The choice of (α, β) has a direct impact on protocol responsiveness to network bandwidth availability. For example, a flow with a large β is very sensitive to bandwidth variation while a small value of α will reduce the aggressiveness of the flow, that is, its ability to probe for the available bandwidth. In [26] a value of $\beta = 0.0625$ is suggested, this value is used in the implementation of RRB-SIMD.

4.4 Simulation Environment

This section describes the Evalvid-RA [38] simulation environment used to perform the evaluation of RRB-SIMD. Evalvid-RA is an extension of Evalvid [83] that supports rate adaptive as well as TFRC implementation. The main components of Evalvid-RA are the pre-processing and post-processing tools used to compress, decompress and evaluate the video content, and the simulated network built using NS2. The basic idea of Evalvid-RA, as shown in Figure 4.2 [38], consists of generating video trace files from a raw video clip usually in YUV format and feeding the trace files into the simulated environment to be used as a traffic generator.

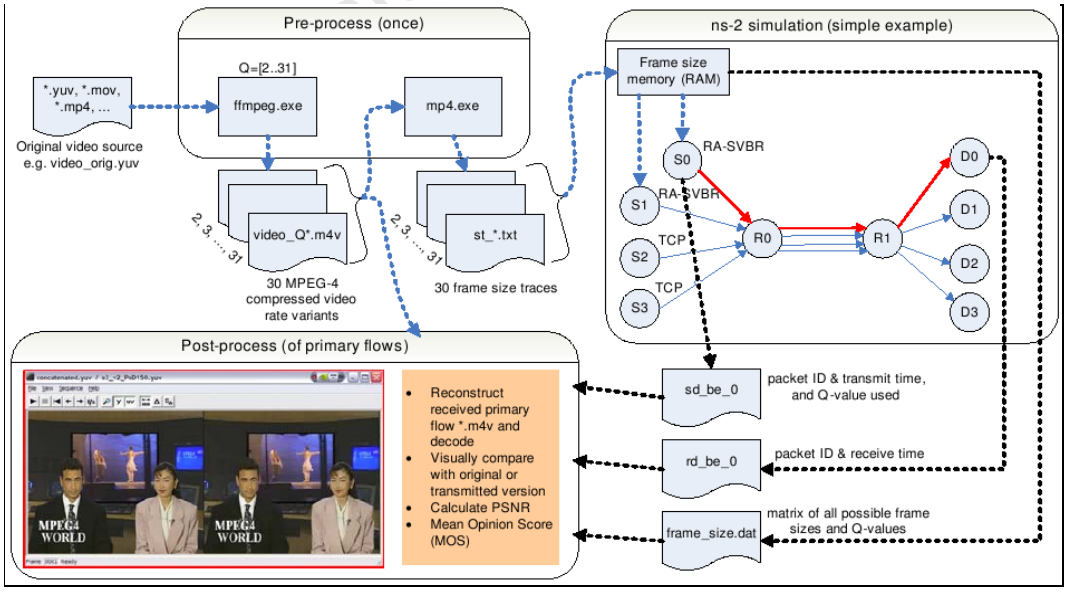

Figure 4.2: Evalvid Tool-set [38]

In this study Ffmpeg-MPEG-4 [84] is used to compress the high-motion video file highway.yuv at 25fps and generate 30 different encoded video clips with quantiser step in the range of 2 to 31. The frame sizes of all clips are in CIF mode. Ffmpeg is configured to generate I, P and B frames in a GoP of 12 as illustrated in Chapter 2. Each encoded video file is then traced using the mp4 programme of Evalvid-RA to produce 30 video trace files. Additionally, Evalvid-RA generates 30 m4v files, each with its associated frame size file. The frame size files are stored in RAM, to decrease the processing time and reduce the risk for the simulation to access external file.

The network topology is simulated using NS2. The trace files are stored into a NS2 server node from where they are streamed to the destination node. The variable bit rate controller of Evalvid-RA considers a new quantiser scale for the next GoP based on the information provided by the congestion control protocol about the path condition. During simulation, Evalvid-RA stores the trace files sd_be_0 at sender and rd_be_0 at the receiver. The two files are used at the post-processing step to facilitate reconstruction (decompression) of the video file as well as computation of network metric and video evaluation metrics. Network metrics that are computed during post-processing include the throughput of the video flow, the end-to-end delay and the jitter. For the quality evaluation of the transmitted video Evalvid-RA uses the *Peak Signal to Noise Ratio* (PSNR) [36] and the *Mean Opinion Score* (MOS) [84] values. The PSNR is obtained using the full reference method [86], where, each frame of the video at the sender is compared with the corresponding frame at the receiver. PSNR values can be converted to MOS values to get the subjective quality measurement. The mapping table from PSNR to MOS is shown in Table 4.1 [83].

It should be noted that the MOS derived from the PSNR represents an objective quality measure, and as such should not be confused with the subjective MOS, i.e., the opinion a number of human observers may have of the video clip.

Table 4.1: ITU-Quality and Impaired Scale, and PSNR to MOS Mapping

PSNR (db)	MOS	Perceived Quality	Impairment
>37	5	Excellent	Imperceptible
31-37	4	Good	Perceptible but not annoying
25-30	3	Fair	Slightly annoying
20-24	2	Poor	Annoying
< 20	1	Bad	Very annoying

4.5 Simulation and Results

4.5.1 Single Bottleneck link

In the first simulation, the Highway video sequence is used. This highway sequence is very challenging as it provides high-motion and low PSNR values. The video is stored in a server and then transferred from the server to the receiver using the single bottleneck topology in Figure 4.3. A first-in first-out scheduling queue with a length set to double the bandwidth-delay product is considered at the ingress router of the simulated network. This is done to avoid packet loss due to a short buffer. In addition a TCP background traffic flow is run concurrently with the video flow to investigate the TCP-friendliness of the protocol as well as the performance improvement for video. The video is first streamed using TFRC protocol and then the same experiment is repeated with the video now flowing over RRB-SIMD.

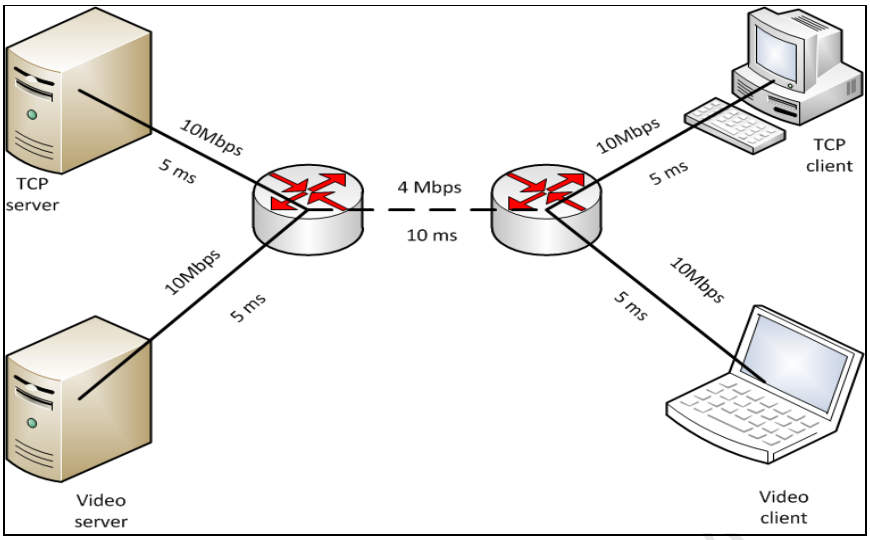

Figure 4.3: Single One-hop Link Topology

For all the simulations the high-threshold T_2 is set to 0.7 and the low-threshold T_1 to 0.5. The two values are set manually after a number of experiments conducted using the topology in Figure 4.3 and selecting the couple (T_1, T_2) that realises the best trade-off between achieved throughput and lowest frame loss rate.

4.5.1.1 Throughput

Throughput is measured during simulation time. Figures 4.4 and 4.5 present the results of RRB-SIMD and TFRC simulations. It is observed that both protocols are TCP-friendly as the shared bandwidth is almost equally distributed between the video and the TCP flow. RRB-SIMD achieved an average throughput of 1.72 Mbps and has lower frame rate loss than TFRC which had 1.73 Mbps average throughput. Over the total 2000 frames transmitted, RRB-SIMD presents a frame loss ratio of 0.10% (2 lost frames out of 2000 total) while TFRC presents frame loss ratio of 3.30% (66 lost frames out of 2000 total). However, TFRC presents a Jain's fairness index [27] equal to 0.83 inferior to the 0.89 Jain's index of RRB-SIMD meaning that RRB-SIMD is friendlier to TCP than TFRC.

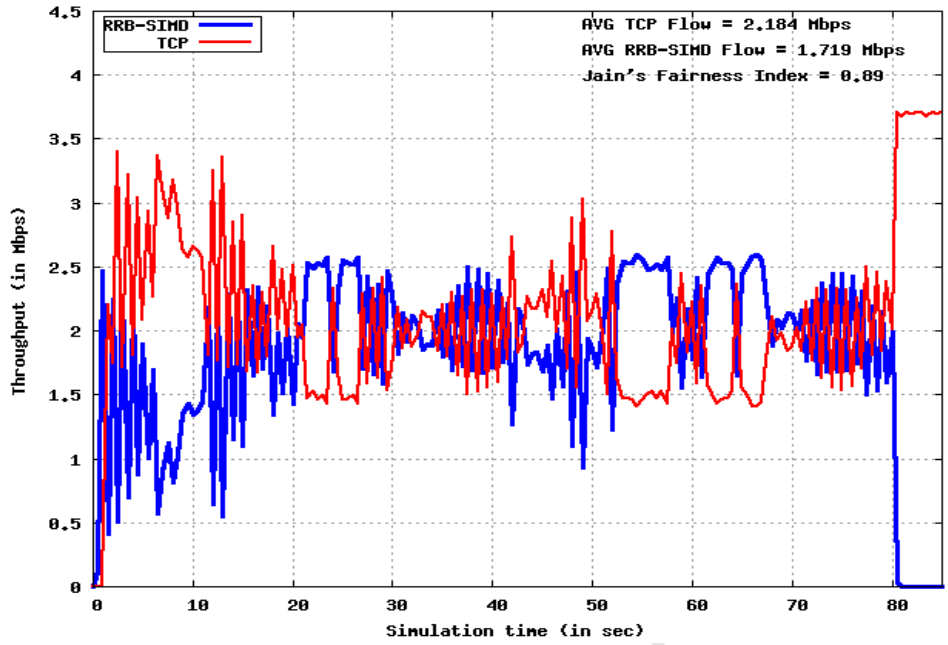

Figure 4.4: Throughput of RRB-SIMD and TCP flows

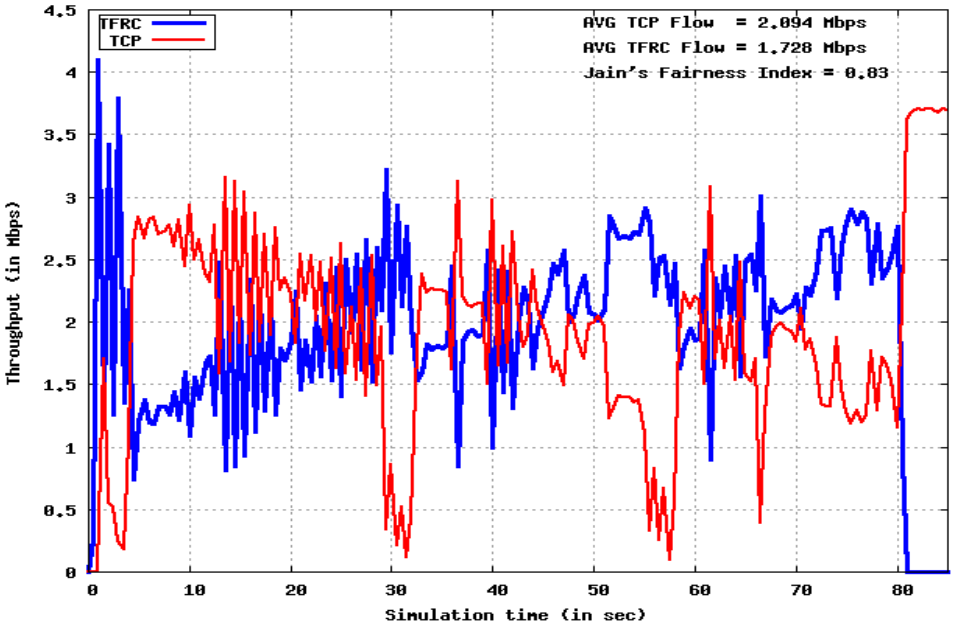

Figure 4.5: Throughput of TFRC and TCP flows

4.5.1.2 End-to-end Delay

Figures 4.6 and 4.7 show that TFRC presents an higher end-to-end delay 269.3 ms than RRB-SIMD 147.7 ms. The delay for RRB-SIMD does not exceed 150 ms, which is the delay limit of conversational media applications. However, for TFRC for the 1934 received frames 33 frames (1.7%) have arrived late (exceeded 150 ms) and could have been considered as lost which should have aggravated the perceived-quality of the media at the end observer. The delay improvement in RRB-SIMD is due to the control criterion using the delay factor that forces the sender to reduce its sending rate every time that the delay threshold is exceeded, reducing the risk of overloading the network with packets that might have arrived late.

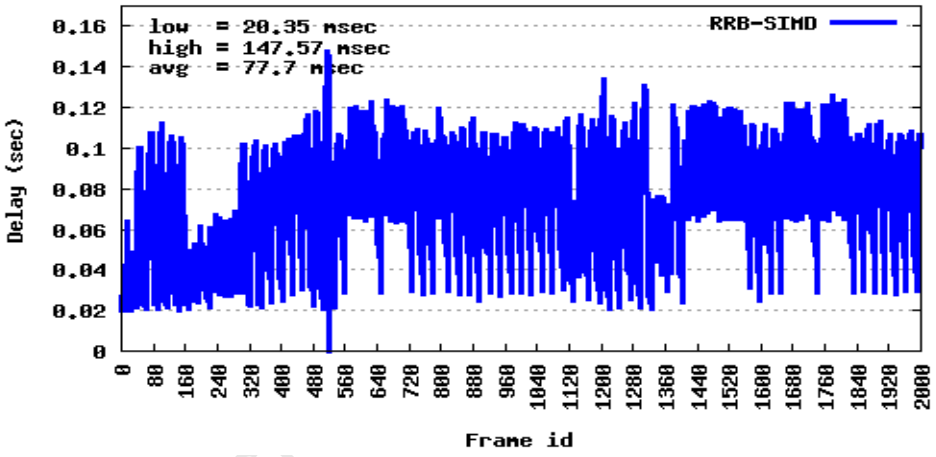

Figure 4.6: End-to-end delay of RRB-SIMD flow

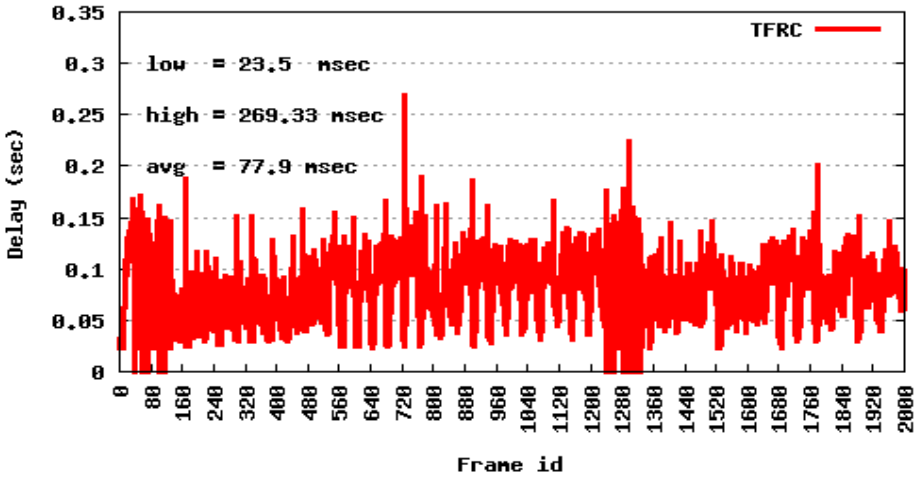

Figure 4.7: End-to-end delay of TFRC flow

4.5.1.3 Cumulative Jitter

Figure 4.8 presents the cumulative jitter for RRB-SIMD and TFRC. From the plot it is observed that the cumulative jitter of TFRC is higher than the cumulative jitter of RRB-SIMD. RRB-SIMD presents a good performance as the values of cumulative jitter were below 150 ms for over the whole simulation time.

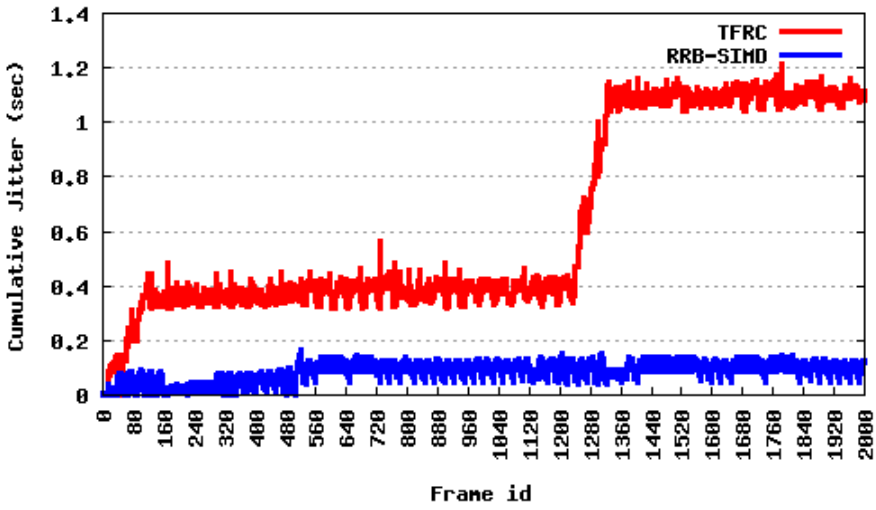

Figure 4.8: Cumulative jitter of TFRC and RRB-SIMD flows

4.5.1.4 PSNR and MOS

Results for PSNR and MOS are illustrated in Figures 4.9 and 4.10, respectively. The average MOS for RRB-SIMD is 3.51 and 2.14 for TFRC. From the MOS values 88.99% of the video frames in RRB-SIMD is graded from fair (MOS=3) to excellent (MOS=5), whereas only 27.68% of the video frames in TFRC realized the same performance. The 61.31% improvement in the video quality is due to the low lost frame ratio realized in RRB-SIMD as a result of the smooth rate adjustment and congestion control using the combined cumulative jitter and delay factor scheme.

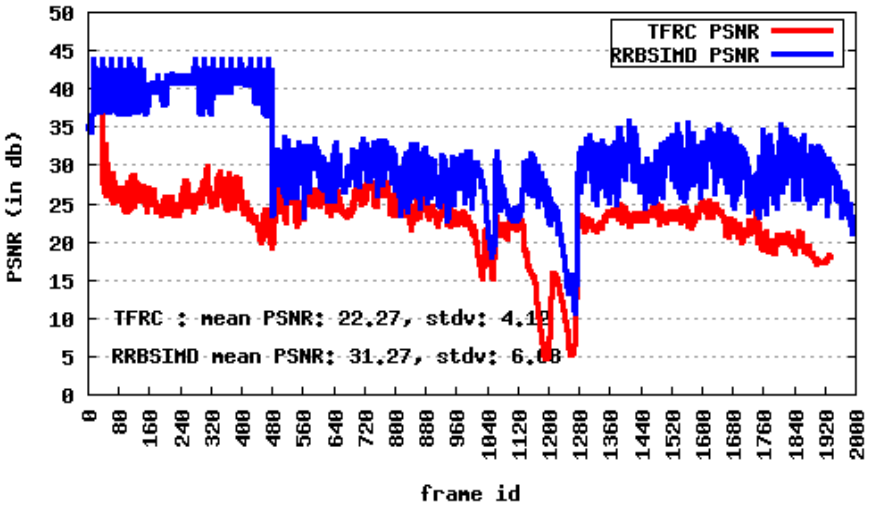

Figure 4.9: PSNR values of TFRC and RRB-SIMD

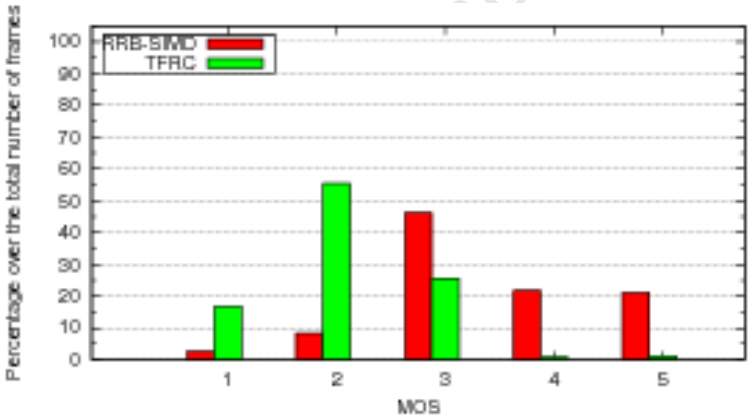

Figure 4.10: MOS values of TFRC and RRB-SIMD

4.5.2 Responsiveness to Dynamics Competing UDP Flows

In this simulation, a more complex scenario where the RRB-SIMD flow shares the same bottleneck with 15 TCP flows and one ON/OFF constant bit rate UDP traffic is considered. The bottleneck is set to 32 Mbps (see Figure 4.11) and the queue size to the bandwidth delay product. The concat.yuv video sequence introduced in Chapter 2 is used as it combines various video sequences of different complexity which can better simulate video transmission as its temporal resolution changes over time. The initial rate for the media is set to 1000 kbps with a packet size

of 1012 bytes. For TCP flows, the new Reno is used with a rate of 2 Mbps. To evaluate the responsiveness of RRB-SIMD and TFRC protocols, the available bandwidth is varied by injecting a Constant Bit Rate (CBR) source at 2 Mbps during ON periods. The experiment is then repeated with the same conditions for the media flow over TFRC. Figure 9 presents the throughput results of RRB-SIMD, TFRC and the ON/OFF traffic. TFRC presents a lower average throughput (1.41 Mbps) than RRB-SIMD (1.62 Mbps) but reacts much faster to networks changes due to injection of UDP traffic. Therefore, TFRC seems to be a more practical solution in situation where bandwidth usage is a primary concern.

In this experiment, TFRC uses a slow start mechanism similar to that of TCP where the rate is increased exponentially until a packet loss event is reported. This explains the aggressiveness of the protocol at the beginning of the simulation where the transmission rate exceeds 2 Mbps. On the other side, RRB-SIMD starts very slowly due to its start-up mechanism (1 packet per RTT). After a loss, its aggressiveness increases as RRB-SIMD uses the square increase rule. It takes longer for RRB-SIMD to reach its highest transmission rate but this is done to ensure a smooth transmission rate in order to avoid packet loss. However, the square increase rule of the protocol may sometimes be dangerous as RRB-SIMD tends to overshoot the bottleneck bandwidth. For example, around 50 to 60 s of the simulation where RRB-SIMD transmits at a higher rate than UDP traffic. This situation could have been prevented if the delay threshold parameters of the protocol, that is, T_1 and T_2, were selected appropriately.

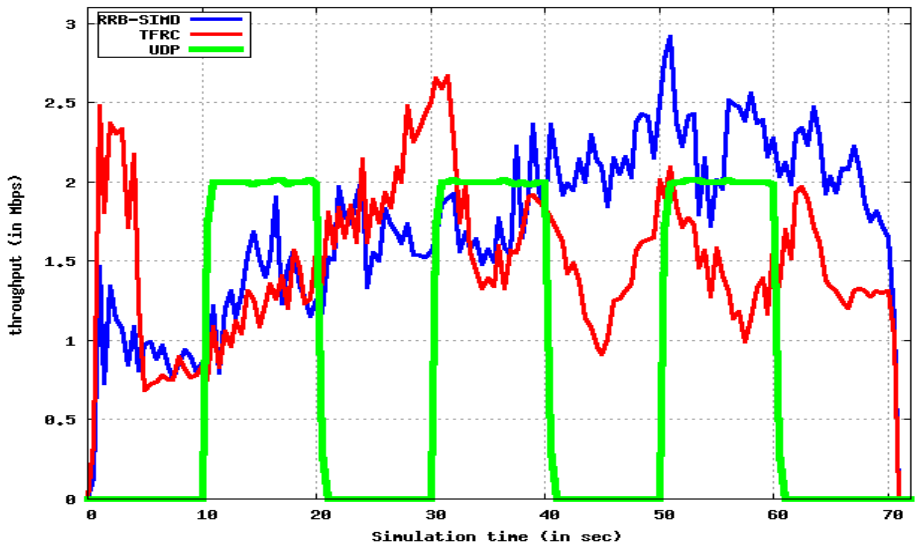

Figure 4.11: Throughput of TFRC and RRB-SIMD flows

For the 1754 frames that forms the media, TFRC realized a frame loss rate of 2.11% (37 frames lost) higher than the 0.22% loss frame ratio (4 frames lost) of RRB-SIMD. The average PSNR for TFRC is 21.69 db with a deviation of 3.86 lower than 40.59 db average PSNR value of RRB-SIMD which has a deviation of 3.56. MOS values of the received video are derived from the PSNR (see Figure 4.12) and presented in Figure 4.13. It is observed from the plot that RRB-SIMD clearly performs better than TFRC as approximately 96.00% of the received video is graded from good to excellent.

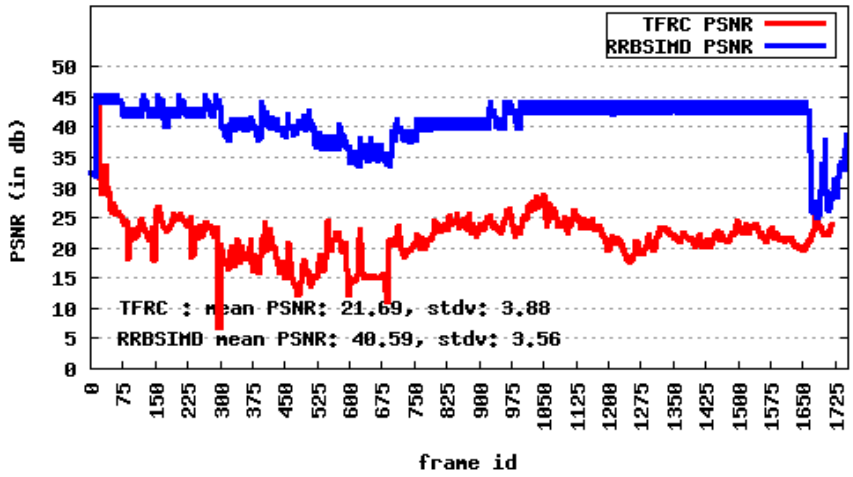

Figure 4.12: PSNR values of TFRC and RRB-SIMD

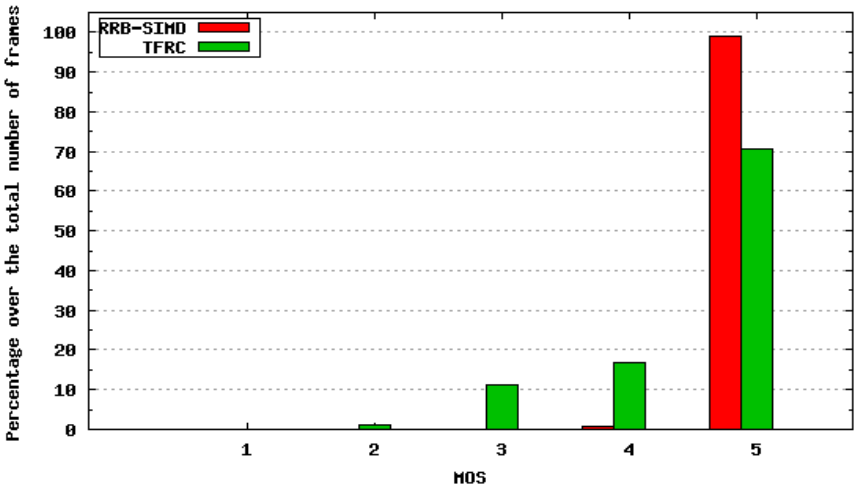

Figure 4.13: MOS values of TFRC and RRB-SIMD

The performance of both protocols in terms of delay (Figure 4.14) and cumulative jitter (Figure 4.15) is also investigated. The delay of TFRC is higher (34.2 ms) than the delay of RRB-SIMD (11.8 ms). The average delay for RRB-SIMD is 73 ms lower than the 88 ms of TFRC. RRB-SIMD also has a much better performance in terms of the cumulative jitter, as the value of the cumulative jitter rarely exceeds 100 ms far below the highest value of the cumulative jitter of TFRC which exceeds 300 ms. The improved performance of RRB-SIMD in terms of delay and cumulative jitter is a result of the cumulative jitter and the delay factor as control criteria.

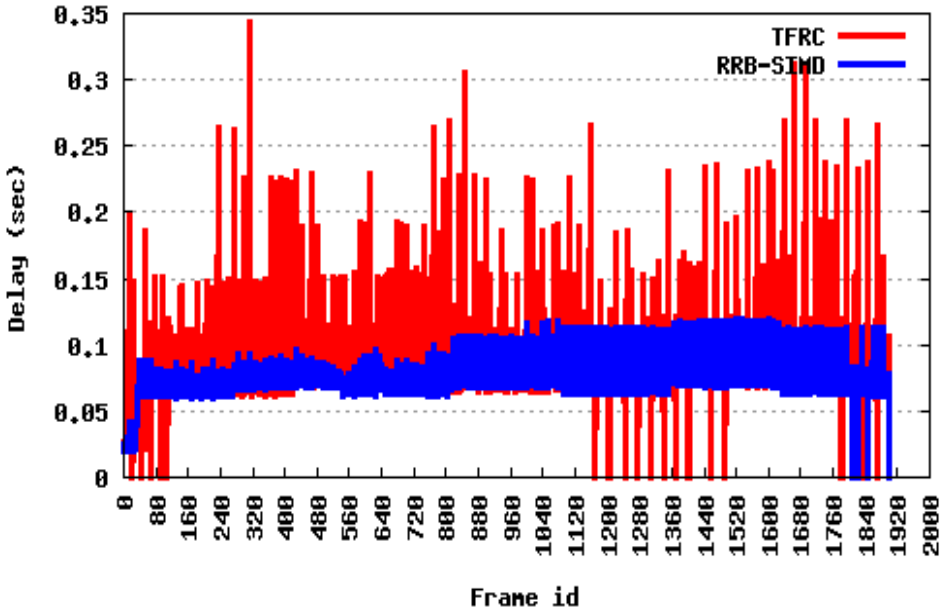

Figure 4.14: End-to-end delay of TFRC and RRB-SIMD flows

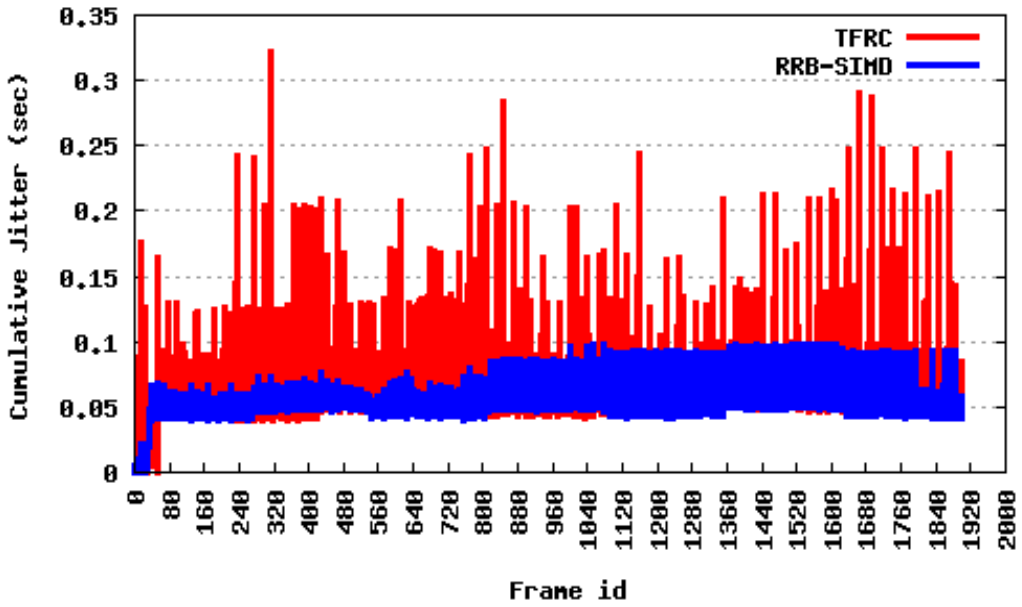

Figure 4.15: Cumulative jitter of TFRC and RRB-SIMD flows

4.5.3 Complex Network Model

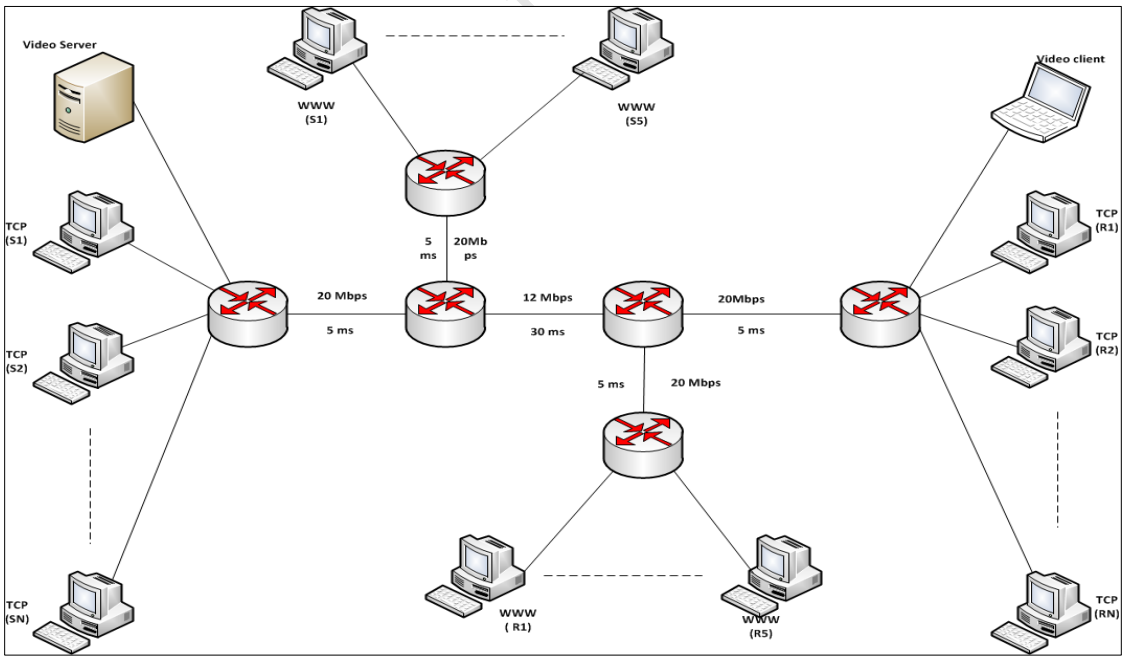

Figure 4.16: Complex network topology

This section elaborates on the behavior of RRB-SIMD under various network conditions. Similar to [44], the topology in Figure 4.16 introduces several Pareto distributed ON/OFF sources that emulate the behavior of competing short TCP connections. This behavior is often attributed to WWW traffics [87] [88]. The mean ON time is set to 1 s and the mean OFF time to 2 s. The shape of the Pareto distribution is set to 1.5. The number of competing short TCP flows is set to 5, with each WWW sender (SN) transmitting its packets at 200kbps to a WWW receiver RN (N=1,2,…,5). The network consists of several access links, all having a bandwidth set to 100Mbps and the buffer size at the bottleneck to 130 packets. The packet size for all TCP flows is 1000 bytes. The delay for all access links are set to 5 ms except for the bottleneck link which is set to 30 ms. The bottleneck link is configured to have a bandwidth of 12 Mbps. A number *N* of long live TCP background flows are also concurrently run to realize congestion at the bottleneck link. The TCP flows are all based on TCP/Reno model, with a maximum congestion window of 1000 packets. Packets from a long life TCP application flow from sender (SN) to receiver (RN), with N being the sender and the receiver identification number. All, the plots presented in this section are obtained by varying the number of long life TCP flows from 1 to 20 to realize different network congestion conditions. The concat.yuv video sample is once again used and streamed over RRB-SIMD to the destination end. The simulation is repeated with the same video sent over TFRC. Figure 4.17 illustrates the average throughput results of both RRB-SIMD and TFRC protocols.

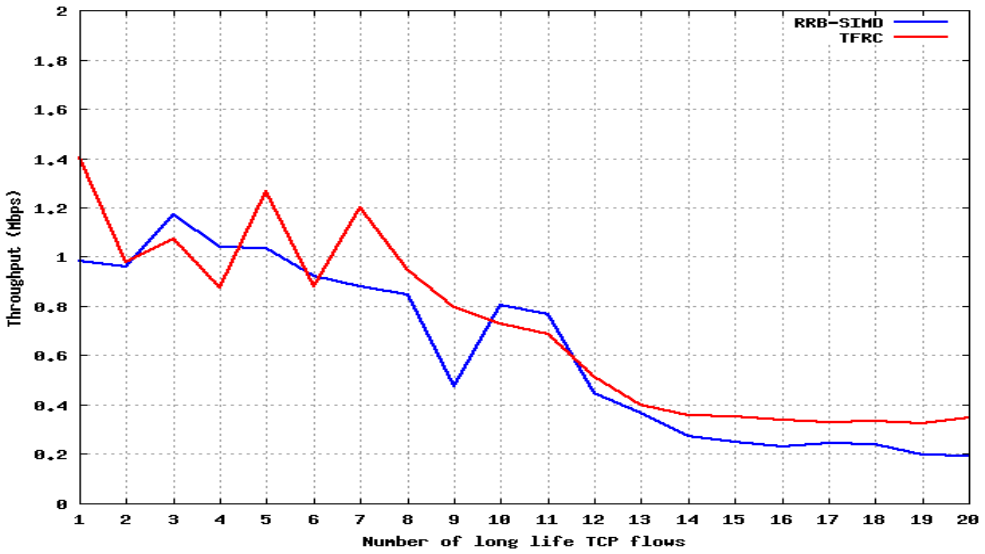

Figure 4.17: Average Throughput of RRB-SIMD and TCP flows

From the plot, after a transient behavior at start-up, the average throughput of both TFRC and RRB-SIMD protocols show a steady behavior as the number of long life flows exceeds 11. Overall, TFRC realizes the higher average throughput than RRB-SIMD. However, the throughput improvement in TFRC comes with a cost of higher frames loss. Figure 3.18 illustrates the percentage of frame loss as the number of TCP flows increases. An approximate 10.00% frame loss is reached at worst scenario for TFRC (N=17), while the average frame loss in the case of RRB-SIMD did not exceed 7.00%, for the all the simulation. Sometimes, there are unexpected spikes and dips observed in frame loss pattern as was the case with throughput. For example, a spike is observed in point 3, followed by a dip in point 4. This means that the frame loss is higher in point 3 than point 4, when one would expected the opposite since the network traffic load has increased. This situation can be explained by the traffic pattern of short time WWW flows, which may vary from one case to another.

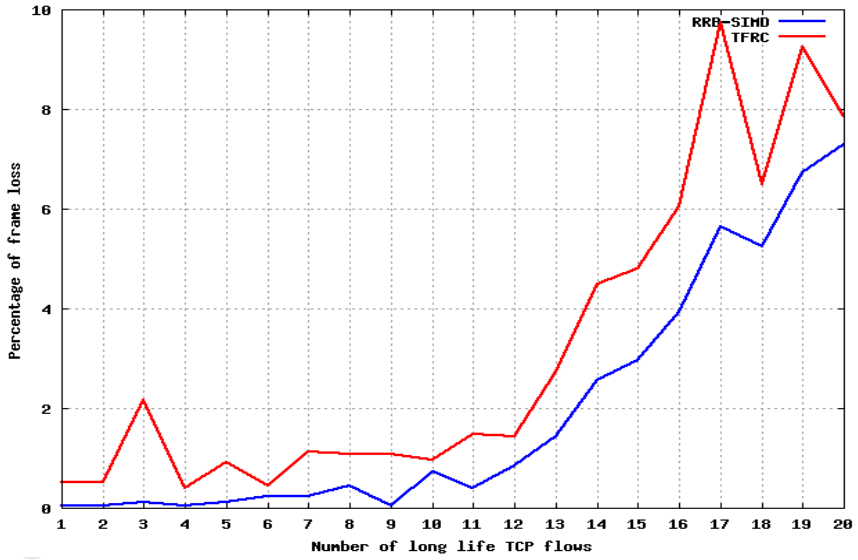

Figure 4.18: Average frame loss for RRB-SIMD and TFRC

Figure 4.19 expresses the frame loss results in term of average peak-signal to noise ratio. As expected, the lower frame loss observed when streaming with RRB-SIMD is transformed to an higher average PSNR. Higher PSNR means better visual quality. When the number of competing TCP flow is less than 7, the network condition is good, and the video is streamed with an excellent visual quality (see MOS table). As the number of competing TCP traffic increases,

the network conditions rapidly deteriorate; this leads to transmission of video content at very low rate.

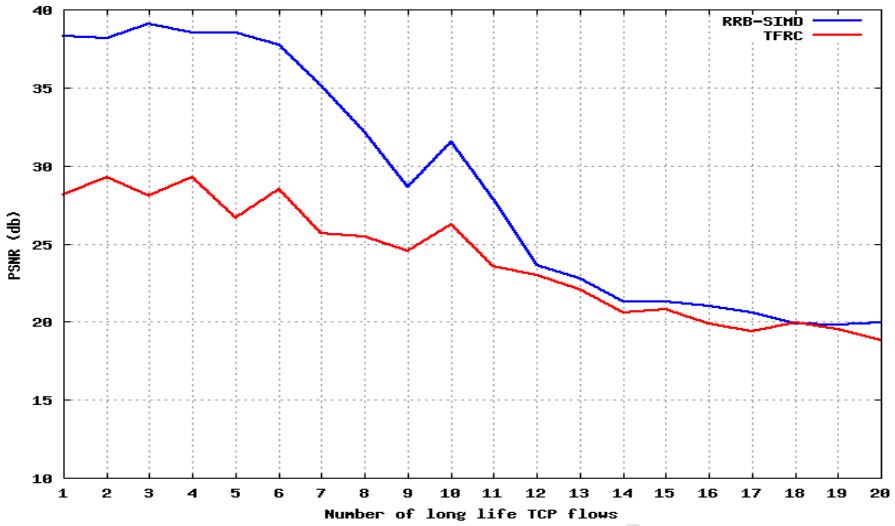

Figure 4.19: Average PSNR of RRB-SIMD and TFRC

The low rate transmission rate is also expressed with a low PSNR. RRB-SIMD realizes the better PSNR values than TFRC, when the number of flows is small, but as this number increases, the difference in PSNR between the two protocols is not significant, although RRB-SIMD still realizes better PSNR values.

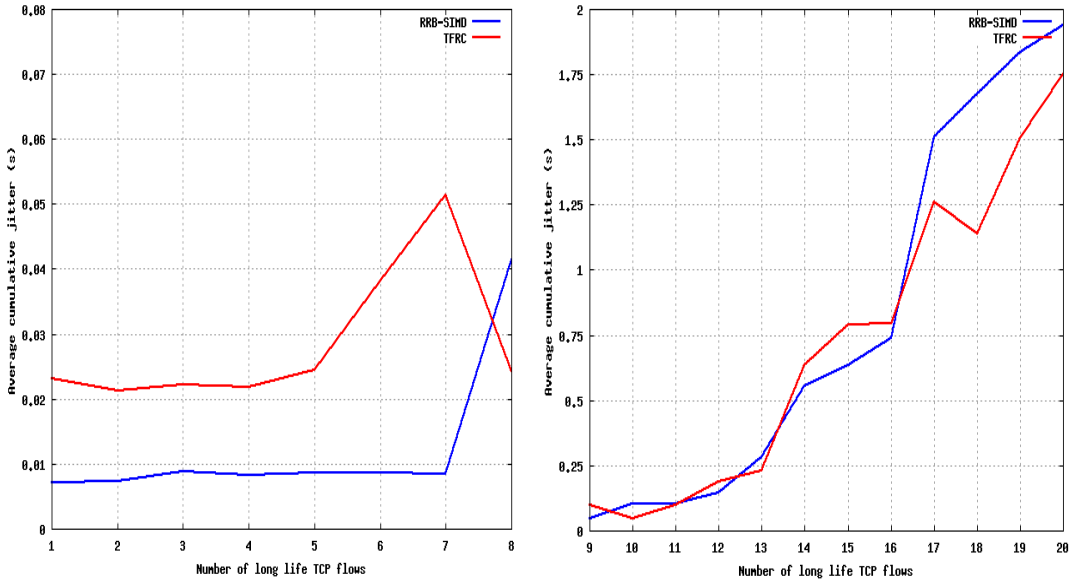

Figure 4.20: Average cumulative jitters of TFRC and RRB-SIMD

Figure 4.20 illustrates the behavior of the average cumulative jitter for both protocols. The result is split into two parts to better illustrated the changes on the average cumulative jitter. From the first plot, it can be observed that the cumulative jitter in RRB-SIMD is lower than in TFRC when the number of flows is less or equal to 7. In this case, a short buffer can be provided for at the decoder to alleviate the delay variation (jitter). When the number of TCP flows exceeds 7, the average cumulative jitter shows an unexpected behaviour, still globally RRB-SIMD has realised the lower average cumulative jitters when the number of flow is less than 16. However, TFRC proves to have the lower average cumulative jitter than RRB-SIMD when the number of competing TCP traffic exceeds 16, which reflects the situation of a network with a very restricted network throughput.

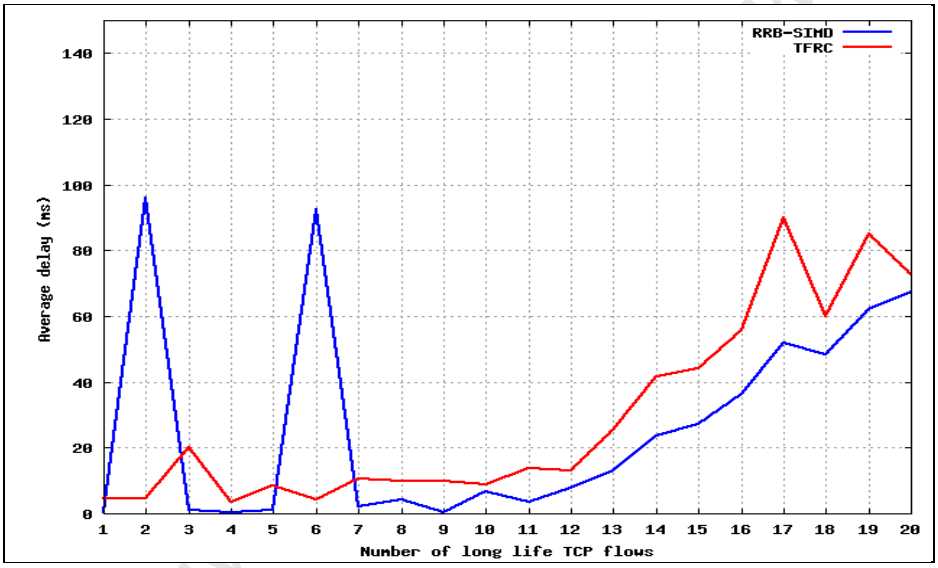

Figure 4.21: Average delay of TFRC and RRB-SIMD

Figure 4.21 illustrates the average delay behaviour of both RRB-SIMD and TFRC. From the plot, both protocols realized on average a delay within the limit of conversational applications (150ms). The average delay in both cases did not exceed 100 ms, and RRB-SIMD realizes the better average delay as the network conditions deteriorate. This deterioration is obtained by increasing the number of long life TCP flows. This performance improvement is a result of a very low conservative transmission rate and the delay factor scheme, which bound the average delay of RRB-SIMD between two threshold values.

4.6 Loss Classification

RRB-SIMD is provided with a combined cumulative jitter and delay factor scheme for congestion detection prior to loss of a packet. When the short running average cumulative jitter is greater than the long running average cumulative jitter, this is seen at the receiver as an indication of an incipient congestion at the bottleneck link. Thus, in the following, the region where the short average cumulative jitter is greater than the long running average cumulative jitter is referred to as the incipient congestion region. If a loss of a packet occurs during a streaming session and if this packet loss is due to congestion, a queue build-up should at least be observed at the buffer of the bottleneck link. If this is the case, then this packet loss is considered as due to congestion. Otherwise, the loss is interpreted as due to a factor other than congestion, and should not count in the computation of the packet loss ratio.

Since the support of RTP/RTCP extension for video streaming in NS2 is still limited to wire networks, the loss classification based on the average cumulative jitter is tested using the wire network topology in Figure 4.16. Here, the following assumption is made: the wire link is free of errors (wire link have a negligible error) and all packets losses are therefore due to network congestion. Hence, the accuracy of the scheme depends on the number of packets lost that lie within the incipient congestion region, that is, within the region where the short cumulative jitter is greater the long running cumulative jitter.

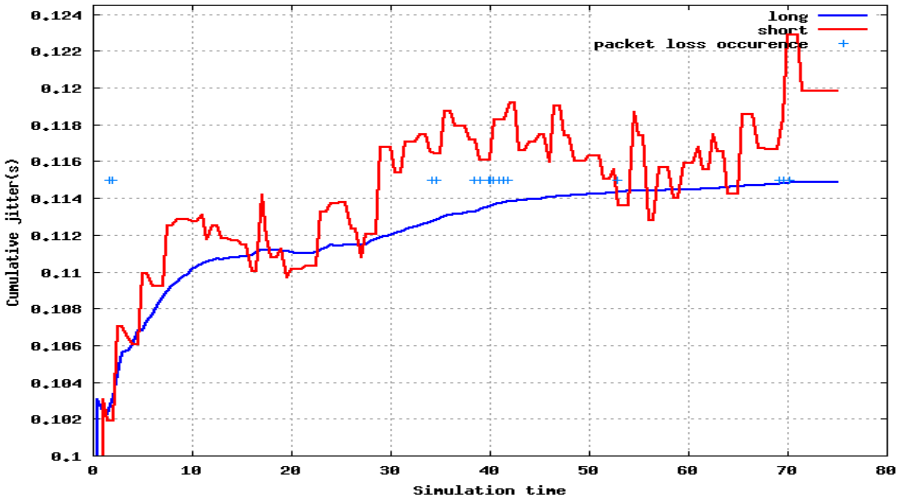

Figure 4.22: Short and long running average cumulative jitter N =10

Figure 4.22 illustrated the results when the number N of long life TCP session is equal to 10. The number is chosen such as the loss frame is less than 2.00% for the streaming session. As illustrated in the plot, all packets loss are contained within the incipient congestion region and are therefore classified as such (are classified as due to congestion). This confirms the assumption made.

The second simulation is conducted to test the robustness of the protocol to poor network conditions. Figure 4.23 illustrates the result when the video experiences a frame loss close to 4.00%. This is obtained by increasing the number of competing long TCP flow to 16.

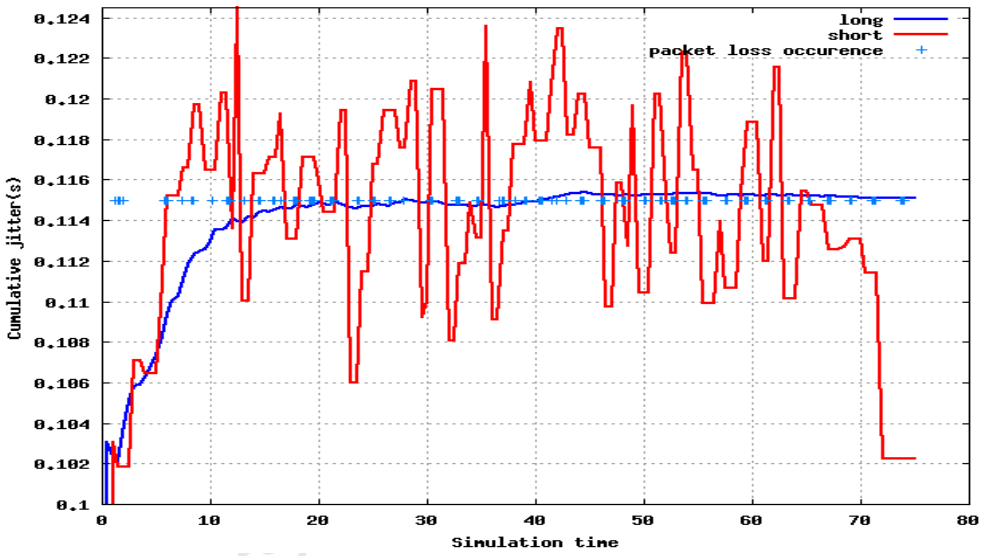

Figure 4.23: Short and long running average cumulative jitter N =16

This frame loss corresponds to a packet loss rate of 1.87%, i.e., 523 video packets loss out of 27959 transmitted. Over the 523 packets lost, 369 packets are into the incipient congestion region, while 154 packets loss are out of the region, and would have been classified as due to a factor other than congestion. This result represents an accuracy of 70.55%. It should be noted that the accuracy of the scheme depends on how often the short average cumulative jitter is sampled, that is, on the timing rule of the feedback reports. Here, the reports are sent following the timing rule of RTCP reports, hence the short average cumulative jitter is updated each five seconds. Figure 4.24, illustrates the result of the same scenario if the RTCP reports were to be send each one second. The accuracy of the scheme has improved from 70.55% to 90.00%.

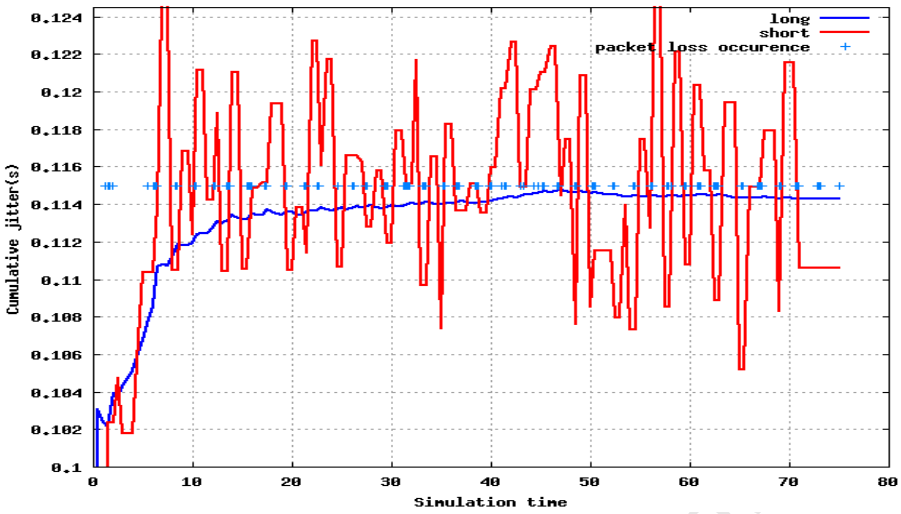

Figure 4.24: Short and long running average cumulative jitters N =16 with more reports

It is also important to mention that the accuracy of cumulative scheme in classifying packets loss also decreases as the video packets loss rate increases. However, the misclassification issue is not particular to the cumulative jitter scheme but inherent to most of the loss classification that are based on a single network metric [89].

4.7 RRB-SIMD Drawbacks

In addition to the timing constraints of feedback reports in RRB-SIMD which set to five seconds between two RTCP reports, another known problem is the difficulty to determine optimal values of the delay factor thresholds that can suit different network conditions. In the results presented above, these values were set manually after many simulation of the topology in section 4.1.

Another problem associated with RRB-SIMD is the difficulty to estimate the one way delay at the receiver end. If in the simulation part of the thesis, it is assumed that the receiver can reliable estimate the one way delay, in practical world,. the difference in synchronization between the sender and the receiver clocks may render the computation of the one way delay inaccurate. However, this problem can be addressed using for example the removal clock skew mechanism proposed in [63].

4.8 Chapter Summary

In this chapter, a transport protocol named RRB-SIMD for video streaming over the Internet that is capable of providing better QoS support than the existing TCP-friendly rate control was presented. Here, the cumulative jitter, an important QoS parameter for multimedia data, is considered in addition to packet loss ratio for congestion control. The cumulative jitter scheme is reinforced with a delay factor to reduce risk of unnecessary decrease of the transmission rate each time that incipient congestion is reported through the cumulative jitter scheme. Simulation results using both network related metrics and video quality measurements show that RRB-SIMD retains the RB-SIMD TCP-friendly property and performs better than TFRC in term of jitter, frame loss rate and, hence, better quality display.

The chapter also discussed a single metric loss classification mechanism that can be used in wireless scenario to improve the performance of the proposed RRB-SIMD protocol, and evaluated the performance accuracy of the scheme.

Chapter 5. An Adaptive FEC Scheme for Video Streaming over Wireless Networks

In the previous chapter the proposed RTP Rate-Base SIMD protocol for video streaming is presented. In this chapter, a mechanism called Dynamic Switching-Research on Enhanced Adaptive FEC (DS-REAFEC) that adaptively includes redundant packets along with original video data packets to recover packet loss is proposed to enhance performance of RRB-SIMD over wireless networks. DS-REAFEC is derived from the Research on Enhanced Adaptive FEC (REAFEC) scheme. REAFEC adaptively computes the number x of FEC packets as a function of the traffic load and the number y of FEC packets using the retransmission times at the medium access layer of the base station; and linearly combines both values to obtain the number of FEC packets to add to video data. The proposed DS-REAFEC mechanism differs to REAFEC in this, while REAFEC uses a static value of the linear combination parameter, DS-REAFEC tunes dynamically this value to cope with various network conditions. DS-REAFEC is implemented at the base station and can therefore be used in combination with any congestion control protocol streaming video data over Wireless Local Area Networks.

5.1 Introduction

Wireless networks make the design of video streaming applications very difficult due to factors such as high packet loss rate, channel fluctuation and limited network throughput [90]. Many techniques have been proposed over the past few years to address the high packet loss rate issue. These techniques include FEC and ARQ schemes [91-95]. In ARQ, missing packets are retransmitted after timeouts or explicit receiver requests, incurring sometimes unacceptable delay. The delay introduced by the retransmission is not suitable for delay-sensitive applications. For this reason, FEC is often preferred to ARQ. FEC mechanisms are designed such that redundant data are transmitted alongside with video packets to recover video packet loss from the redundant information.

In this chapter, an adaptive FEC scheme that dynamically adjusts the number of FEC packets to send along with original video data packets is proposed. The scheme relies on two network parameters: the buffer length at the base station and the number of times a packet is retransmitted from the base station to destination. The buffer length is used to assess the

congestion level at the network while the retransmission time is used to monitor the state of the wireless channel. Depending on the congestion level and the wireless link condition, the number of FEC packets is computed and sent along with the original video packets. In this way the number of added FEC packets do not only provide the destination end with enough information to recover packet losses that are due to the wireless channel impairments, but also prevent congestion that could have been caused by an overhead of a large number of FEC. The mechanism is discussed in general in the following section.

5.2 Adaptive FEC Scheme

FEC mechanisms as already pointed out, add redundant data along with original video data, such that a loss of original video data can be recovered if not totally, at least partially from the redundant data. FEC schemes are generally classified into two groups: *Static* FEC (SFEC) and *Adaptive* FEC (AFEC). In SFEC the number of redundant data added to the original video data remains constant regardless of the network conditions. For example, for every eight video packets, also here referred to as a block, an SFEC scheme will generate and add a fixed number of redundant packets. Adding a fixed number of FEC packets has some disadvantages, such as the difficulty to determine an appropriate number of FEC packets to satisfy a group of mobile devices experiencing different channel conditions. If the number chosen is low, this may result in poor protection and lost packets may not be recovered. On the contrary, if this number is set to high, the overhead caused may degrade the performance of the network [96]. Moreover, with a predefined fixed number of FEC packets a large number of redundant packets is created when the wireless channel error rate is low. For these reasons, AFEC schemes are preferred to SFEC schemes.

AFEC schemes attempt to overcome the limitations of SFEC by dynamically adjusting the number of FEC packets based on the current network condition. Several models of adaptive FEC are presented in the literature. The model proposed in this chapter is based on the approach presented in [59] and later modified in [60]. The two approaches are briefly discussed below.

5.2.1 Enhanced Adaptive FEC scheme

The *Enhanced Adaptive FEC* (EAFEC) scheme proposed in [59] is implemented at the base station (see Figure 5.1) to improve video streaming over *Wireless Local Area Network* (WLAN).

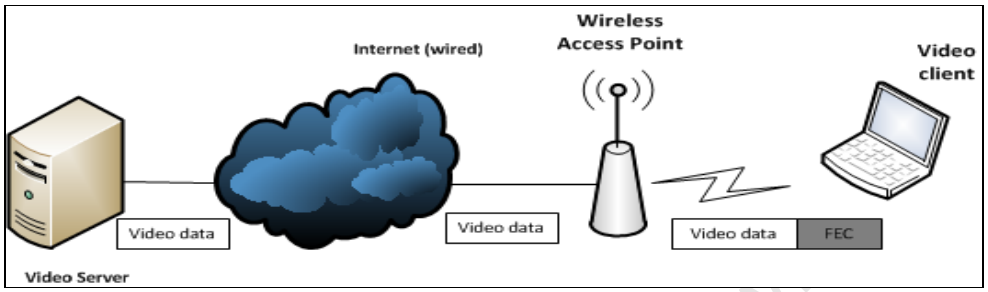

Figure 5.1: Wireless base station adding FEC data to video data

EAFEC relies on two network parameters: the packet queue length and the packet retransmission times at the base station to determine the number of FEC packets. The queue length is used to monitor the evolution of the traffic load at the network. When the queue length is long, this is interpreted at the base station as a sign of a high traffic load in the network, therefore fewer FEC packets are added to avoid overloading the network [59]. Conversely, when the queue length is short, this implies a moderated traffic load in the network, and more FEC packets can be added to improve reconstruction of lost packets at the destination end.

The packet retransmission time, on the other hand, is used to assess the state of the wireless link. When the wireless link is good, retransmission time is less and fewer FEC packets are added since packets stand a much higher chance to reach the destination end. However, when the wireless channel is bad, the retransmission time increases and more FEC packets are generated.

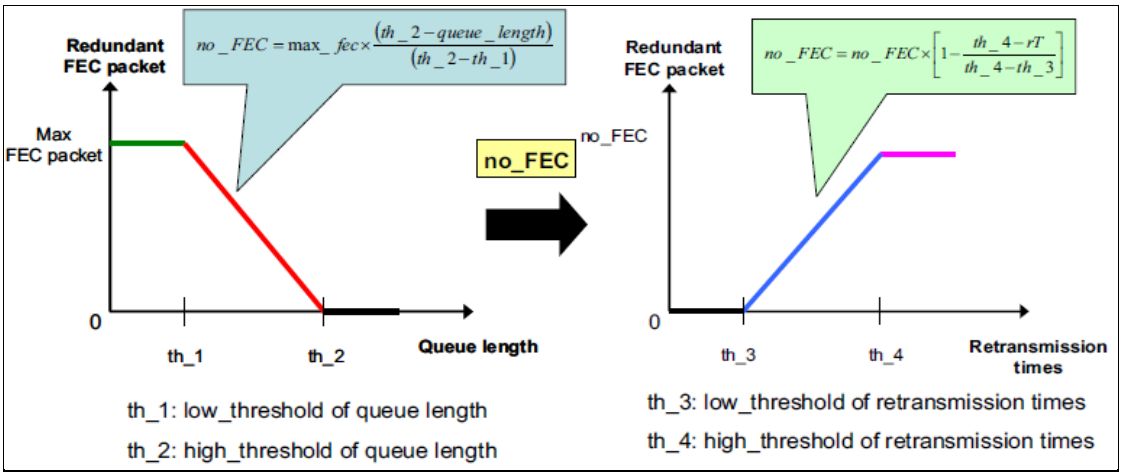

Figure 5.2: Number of FEC packet in EAFEC algorithm [59]

Figure 5.2 summarises the EAFEC running mechanism. When a block of packets is received at the base station, the queue length ($queue\_length$) is calculated and the effect of spurious individual value smoothed using the EWMA filter. The value of $queue\_length$ is then compared with two queue length threshold values: the low threshold th_1 and the high threshold th_2 to determine the number no_FEC of FEC packets. If $queue\_length$ is less than th_1, then a maximum number of FEC packets ($\max\_FEC$) is added. If it is larger than th_2, then no FEC packets are added. Otherwise, the number of FEC packets linearly decreased with $queue\_length$ as shown in Equation 5.1.

$$no\_FEC = \max\_FEC \times (th_2 - queue\_length)/(th_2 - th_1) \qquad (5.1)$$

Then, the computed number no_FEC is revaluated using the retransmission time rT. When, a value of rT is first collected, it is smoothed using the EWMA. Then, it is compared to the low threshold th_3 and the high threshold th_4. If rT is less than th_3, no FEC packets is added, that is, no_FEC is zero. If rT is greater than th_4, the number of FEC packets does not change. Otherwise, the number linearly increases with rT using Equation 5.2.

$$no\_FEC = no\_FEC \times (1 - (th_4 - rt))/(th_4 - th_3) \qquad (5.2)$$

However, the problem with EAFEC is that [60], when the queue length at the base station is too large and the wireless link status bad, no FEC packets are added. As a result, video packets

are not protected against wireless errors. In [60], a mechanism on how to address this limitation is presented. The mechanism is here referred to as REAFEC (*Research on Enhance Adaptive FEC*). REAFEC computes independently the number no_FEC_1 of FEC packets using first the queue length, and then the number no_FEC_2 of FEC packets using the retransmission time. Then, the number no_FEC of FEC packets that are forwarded along with video data is obtained as a linear combination of both numbers as shown in Equation 5.3.

$$no\_FEC = \alpha \times no\_FEC_1 + (1 - \alpha) \times no\_FEC_2 \quad (5.3)$$

where α is a constant chosen equal to 0.6 [60].

REAFEC as described above provides video content with sudden scene changes better protection than EAFEC. However, the use of a static value of the weight parameter α may limit the efficiency of the scheme in high packet rate error wireless system. When the network is congested, it is desirable to weight the contribution of the traffic load higher than the contribution of the retransmission time. However, when the network is not congested, and the wireless channel very bad, weighting the traffic load higher than the wireless state is not recommended. A better approach would be to use the available network bandwidth to protect the quality of the video by giving a higher weight to the number of FEC packets computed from the wireless channel status.

In what follows, an adaptive FEC mechanism that dynamically tunes the value of α and intelligently selects the number of FEC packets to add along video packets is proposed to cope with high packet error rate wireless networks while still providing mobile nodes with video clips with an acceptable quality level regardless the video frames pattern.

5.2.2 Proposed DS-REAFEC Scheme

The proposed adaptive FEC scheme presented in this chapter is referred to as *Dynamic Switching Research Enhance Adaptive FEC* (DS-REAFEC). DS-REAFEC is derived from EAFEC and REAFEC schemes. As EAFEC and REAFEC, DS-EAFEC is implemented at the base station and uses the queue length to monitor the traffic load at the network and the retransmission time to assess the status of the wireless line. The two parameters are then used to compute the number of FEC packets to forward along with video data. The algorithm operates as follows:

When a block of packets is received at the base station, the base station computes the number no_FEC_1 of FEC packets using the average queue length as in EAFEC. Then the base station uses the retransmission time to compute the number no_FEC_2 of FEC based on the wireless status as in REAFEC. Then a test is conducted to decide on whether to allocate a larger weight to no_FEC_1 or to no_FEC_2. If the queue length has exceeded its high threshold then the network is congested. It is natural in this case to give a larger value of the weight to no_FEC_1 to not aggravate congestion with a large number of FEC packets. Therefore, a second test is conducted to observe the status of the wireless channel using the retransmission time. If the retransmission time is less than the higher threshold (th_4), then the wireless channel has not yet reached its worst state, and a higher weight is allocated to no_FEC_1, and Equation 5.3 is used to obtain the number of FEC packets. If the retransmission time is greater than its high threshold, then both links are in their worst conditions. In this case, the minimum between no_FEC_1 and no_FEC_2 determines the number of FEC packets that should be sent.

Conversely, when the queue length is lower than its high threshold, the network is loaded but not congested, and a first priority is given to the wireless channel. If the retransmission time has a higher value than its high threshold, the retransmission time is given the higher weight to protect the video data from the high wireless bit error rate. Thus, the number of packets is obtained using equation 5.4, where the role of alpha is reversed.

$$no\_FEC = (1 - \alpha) \times no\_FEC_1 + \alpha \times no\_FEC_2 \qquad (5.4)$$

However, when the retransmission time is lower than its high threshold, then the maximum between no_FEC_1 and no_FEC_1 determines the number of FEC packets that should be added. The algorithm is summarised in Figure 5.3.

```
//compute no_FEC1 using queue length (qlen)
quelen=(1-qweight)*cur_quelen + qweight*quelen;
if(qlen<threshold1) {
      no_FEC1=max_FEC;
} else if(quelen<threshold2) {
            no_FEC1=max_FEC*(threshold2-
quelen)/(threshold2-threshold1));
 } else {
            no_FEC1=0;
}
//compute no_FEC2 using retransmission time (rT)
rT=(1-sweight) * cur_rT+ sweight * rT;
if(rT < threshold3){
      num_FEC2=0;
} else if (rT < threshold4) {
      no_FEC2=max_FEC*(1-((threshold4-rT)/(threshold4-
threshold3))));
} else {
      no_FEC2 = max_FEC;
}
//DS-REAFEC decision scheme
if (qlen > threshold2) {
            //congestion gets a larger weight than
wireless status
            no_FEC =  α * no_FEC1 + (1- α)* no_FEC2
}
else {
      if (rT > threshold4)
            //wireless status gets a higher weight than
congestion
            no_FEC = (1-α) * no_FEC1 + α * no_FEC2;
      else no_FEC = maximum(no_FEC1,no_FEC2);
}
```

cur_qlen: current value of qlen
cur_rT : current value of rT

Figure 5.3: DS-REAFEC algorithm to compute the number of FEC packets

5.3 Wireless Model

For the wireless channel, the random uniform error model is considered. The model provides the distributed packets with an error rate p. With the random error model, if a unicast sender wants to transmit data packets to mobile nodes and the wireless channel is not good, the data link layer (MAC) of the network stack can retransmit the data packet at a maximum of N times before the data packet is discarded. Hence, the perceived correct rate at the application-level is given by Equation 5.5

$$p_{correct} = \sum_{i=1}^{N} (1-p)p^{i-1} = 1 - p^N \tag{5.5}$$

where N is the maximum number of retransmission at the data link layer (MAC layer) and p the packet loss probability at the physical layer. Hence, the application level loss probability is obtained from Equation 5.6.

$$p_{effective} = p^N \tag{5.6}$$

However, it should be noted that with this model errors are known to occur in isolation and not to be busrty as it is the case in real wireless networks..

5.4 Simulation Setup

The performance evaluation of the proposed model is conducted using NS2 and the network topology in Figure 5.4.

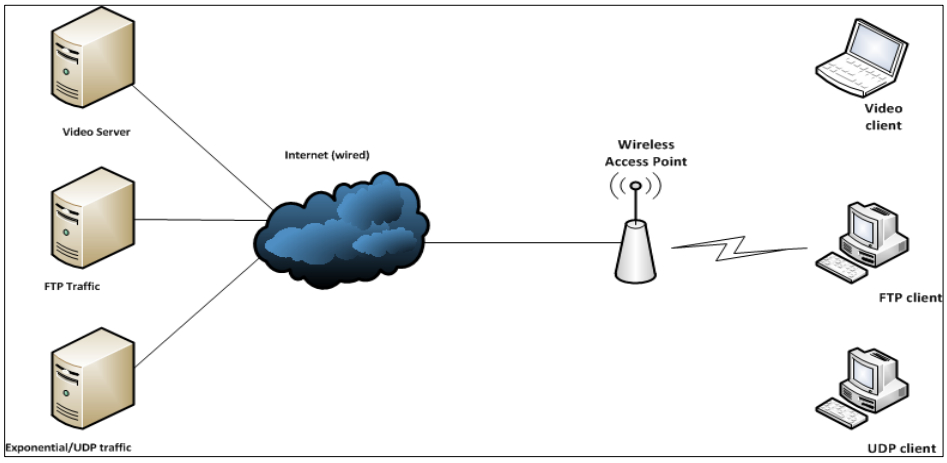

Figure 5.4: Path Model

Here, the simulation is conducted over a light UDP protocol, that is, UDP enhanced with sequence number and timestamps due to lack of support for RTP/RTCP extension for video streaming over wireless networks. The simulation setting is similar to that used in [59] [60] to compare the performance of the proposed model DS-REAFEC with EAFEC, REFAEC and the static FEC (SFEC). The simulation consists of a video client connected to the Internet through an IEEE 802.11b wireless access link. Although IEEE 802.11b today is obsolete, in general terms the results obtained from the model are still valid for later models up to IEEE 802.11n. The wireless access link has a bandwidth of 11 Mbps and queue size of 50 packets. The video client shares network resources with two types of background traffic: an FTP flow transmitted over the transport protocol TCP and an exponential traffic sent over UDP. The video and the two background senders are each connected to the Internet using a wire access link of 10 Mbps. The wire link between the Internet and the base station (access point) has a capacity of 100 Mbps. The highway.yuv video sample is once again used. The video is this time in QCIF format, i.e., 176x144 resolutions format. The file is compressed using H.264 video codec with JM 1.7 [97]. The GoP has the simple profile structure with a length of 12 frames, that is, IPPPPPPPPPPPP. The transmission rate for traffic is 1Mbps; burst and idle time are both set to 0.5 ms.

All the simulated FEC schemes are tested with network parameters set as in [59], that is, with the queue length low and high threshold values set to 10 packets and 40 packets respectively. The packet retransmission time low threshold is five times and the high threshold is 10 times. For FEC schemes, a block is of 8 packets and the maximum number of FEC packets sent is set to 4. Thus, if the sum of packets number in a block and FEC packets numbers for this block is at least equal to eight, then the original video data can be recovered. Otherwise, FEC packets are not useful, and only the video packets are sent to the destination applications.

5.5 Results

Figure 5.5 presents the result of the PSNR for different values of alpha. Overall, DS-EAFEC presents a much better PSNR values than REAFEC, independently of the value alpha used.

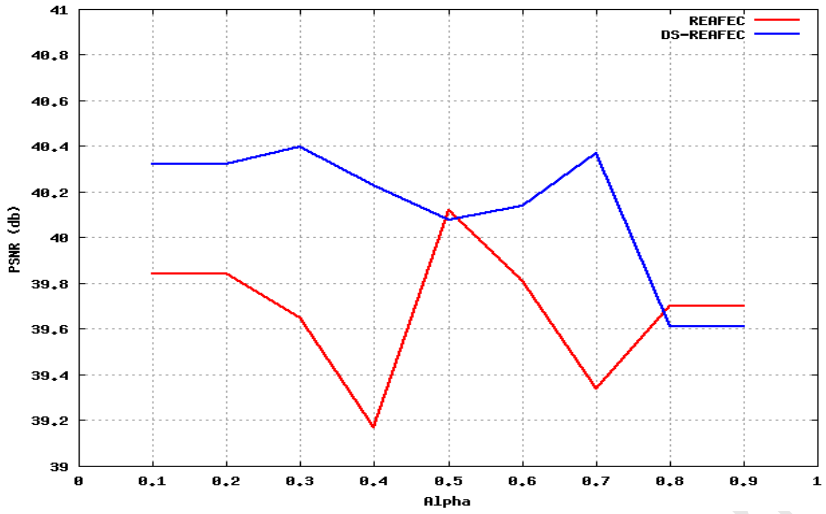

Figure 5.5: Results of PSNR for different α

For DS-EAFEC, the overall performance depends on the dynamics of the system, that is, on how the weight is fluctuated between the two states: the congestion state and the good wireless channel state, and the non-congestion state and bad wireless state during all simulation. In the rest of the simulation the value of alpha for DS-REAFEC is set to 0.7.

In the second simulation, the performance of the DS-REAFEC is tested using different network conditions. The different conditions are obtained by varying the packet error rate from 0.1 to 0.7. Figures 5.6, 5.7 and 5.8 respectively present the average number of packets loss, the average number of FEC and the average PSNR for DS-REAFEC, REAFEC, EAFEC and SFEC schemes. The average values are obtained after 16 independent replications of the simulation. Each replication is started from the same initial conditions and terminated after the entire video clip is transmitted. The independence of the replications is accomplished by using different seeds to initialize the random number generators in each replication. The confidence interval is presented in Appendix B.

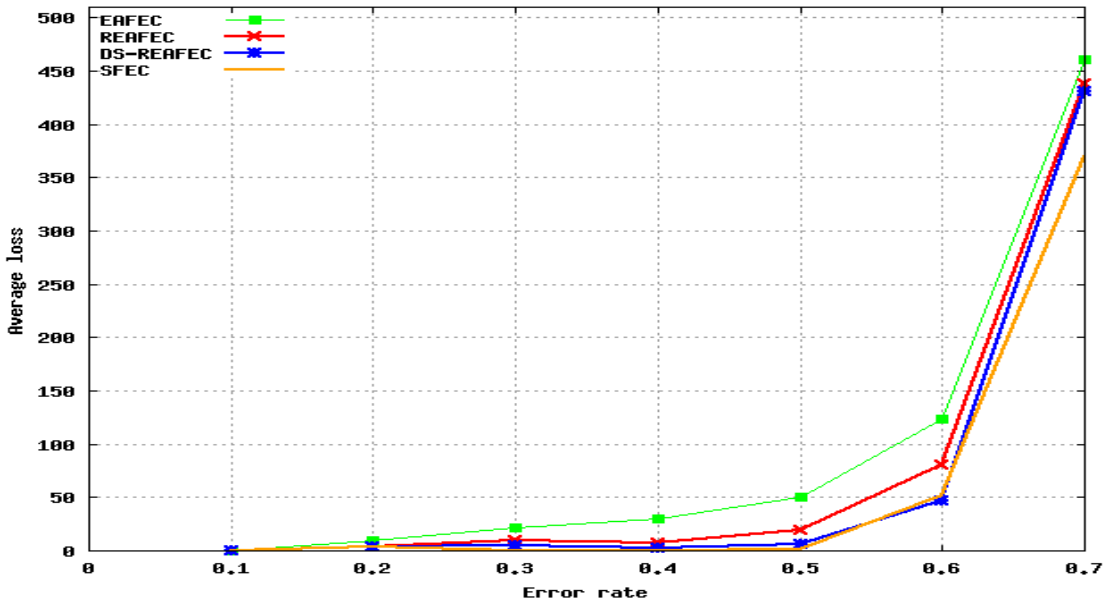

Figure 5.6: Average packet loss numbers for different FEC schemes

The plot in Figure 5.6 shows that DS-EAFEC scheme is more stable with varying packet error rate than EAFEC and REAFEC schemes. For example, when the packet error rate is 0.6, the video session has lost on average 47 packets when using the DS-REAFEC scheme, while the average number of packet loss for REAFEC is and EAFEC has exceeded 80 packets.

However, the performance improvement of DS-REAFEC comes with the cost of a higher number of redundant FEC packets as illustrated in Figure 5.7. If the network packet loss rate is less than 0.2, the number of redundant FEC places is irrelevant since the achieved average PSNR for all the schemes is 40.40 (see Figure 5.8) as no video packet is lost. In this case EAFEC is the best solution of the three schemes since it places less redundant data. With the less redundant data network resources are well utilised. However, EAFEC efficiency decreases with the increase of the packet error rate. For example, when the loss rate is equal or greater than 0.3, although the achieved average PSNRs for all the three schemes are still in the range [38, 40], the PSNR per frame (instantt PSNR) tells a completely different history. Instant PSNR reflects the real-time impression that the viewer has of the video clip. The results of the instant PSNR are shown in Figures 5.9 and 5.10 and are obtained with α set to 0.7 for DS-REAFEC and the packet loss rate to 0.3.

From the plots, it can be seen that DS-REAFEC shows better instant PSNR than EAFEC and REAFEC. This result shows that the extra number of FEC obtained from the scheme is necessary to recover important information to keep the quality of the video at an excellent level (see MOS).

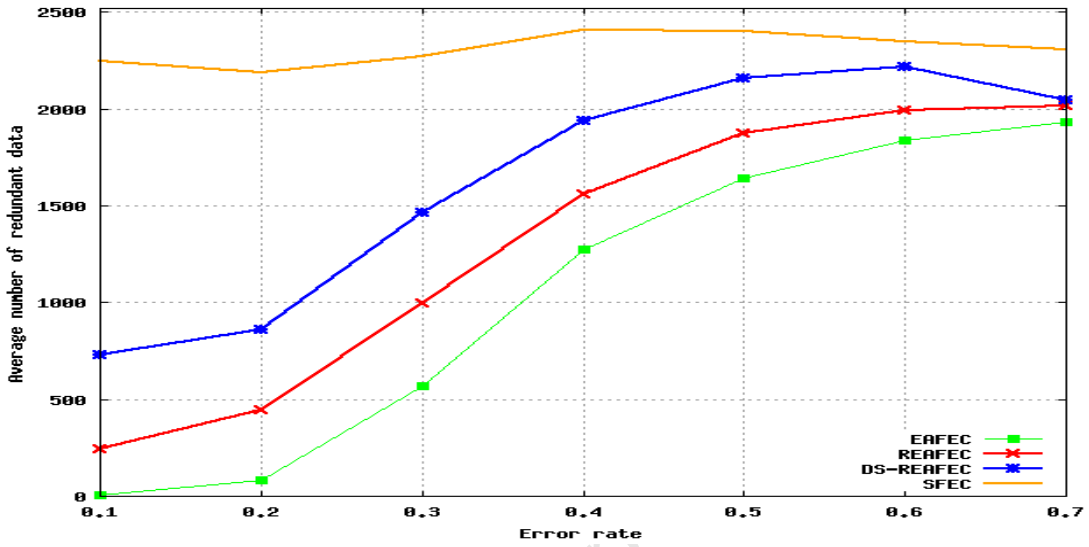

Figure 5.7: Average number of Redundant packets for different FEC schemes

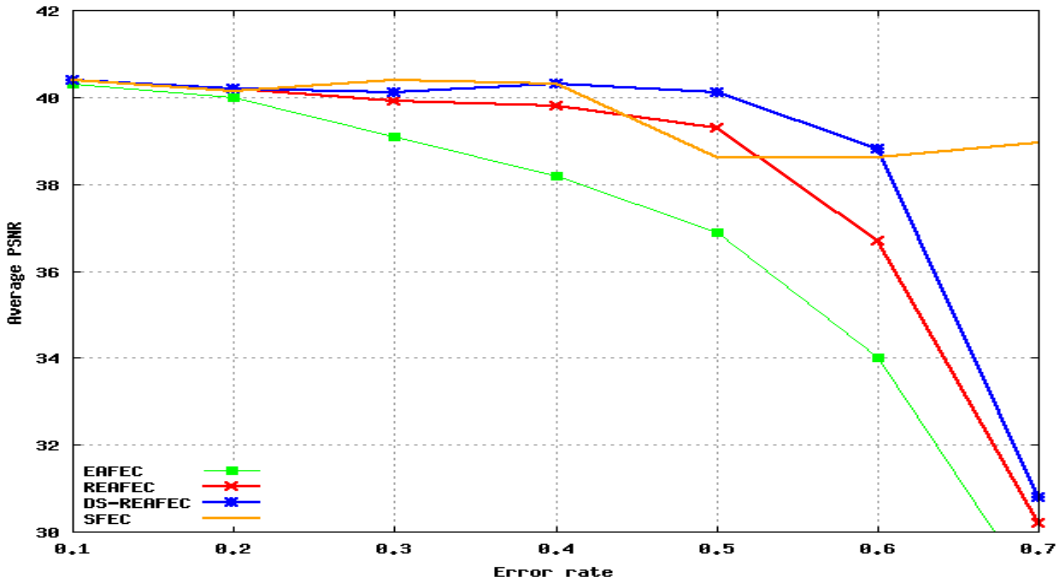

Figure 5.8: PSNR of different FEC schemes

Moreover, when there are sudden changes, i.e., dips in the PSNR values of REAFEC and EAFEC, DS-REAFEC presents a much better result. The PSNR of DS-REAFEC is maintained

around 40.2 db for all the video frames, providing the receiver with an image with a much better quality than REAFEC and EAFEC.

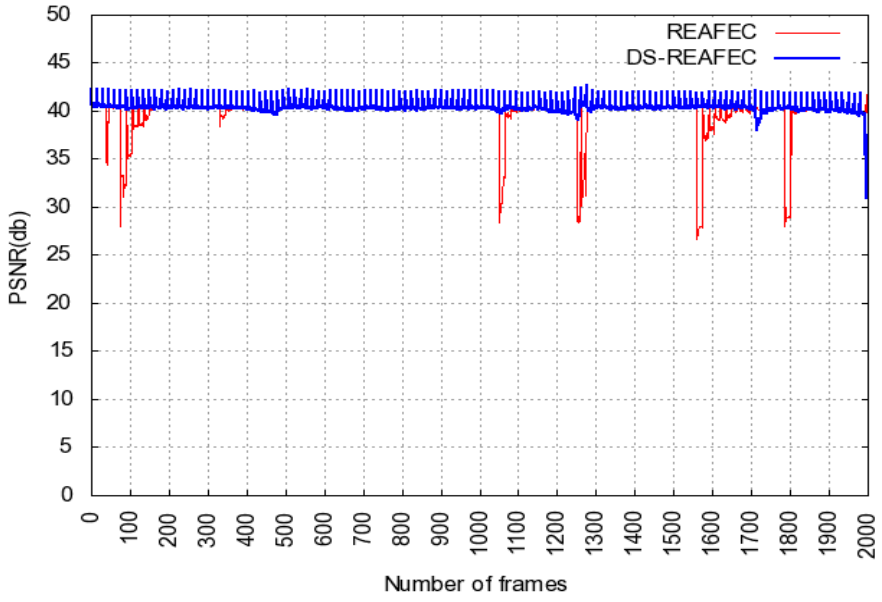

Figure 5.9: Comparison of PSNR of DS-REAFEC (α=0.7) and REAFEC

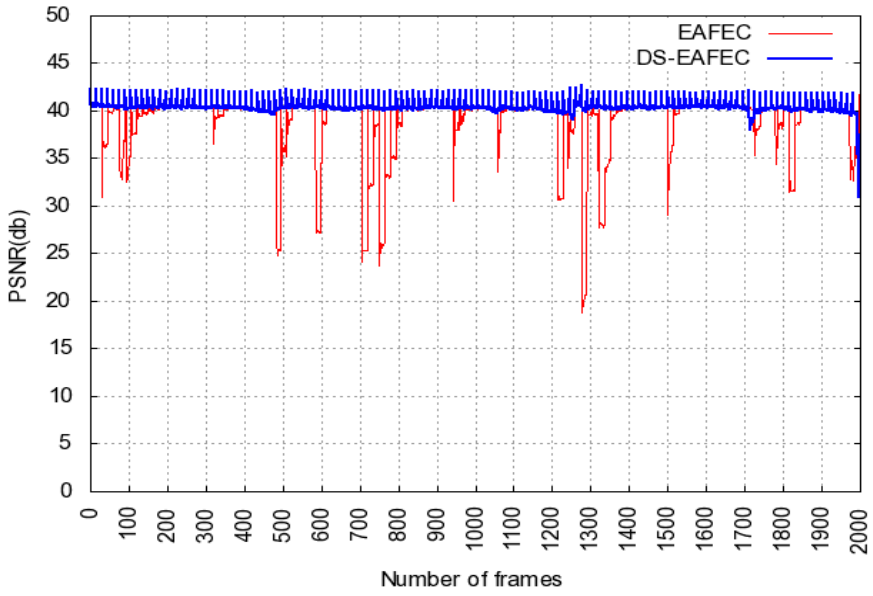

Figure 5.10: Comparison of PSNR of DS-REAFEC (α=0.7) and EAFEC

5.6 Summary

This chapter discusses an adaptive FEC mechanism to improve video delivery over wireless networks when not overloading the network traffic load with a large redundant data overhead. The adaptive FEC mechanism is implemented at the base station and uses the base station buffer length and the packet retransmission times to decide on the appropriate number of FEC packets to send along with video data packets. The simulation results show that the proposed protocol improves the video quality better than two other existing schemes when the wireless network condition is poor.

Chapter 6. Conclusion and Future Work

6.1 Conclusions

The main objective of this project has been to design a protocol to regulate transmission rate for streaming video over best-effort networks such as today's Internet. The aim of the protocol is to guarantee bandwidth sharing between streaming sessions and competing TCP flows, while still ensuring continuous display of the video clips at an acceptable quality level. Thus, this study proposed a protocol called RRB-SIMD that runs on top of the real-time transport protocol and uses its sister protocol, real-time control protocol for feedback reports. The main idea of RRB-SIMD is to alter the output bit rate at the video encoder to match the available throughput of the network, every time that a change in the network's available throughput is detected and reported by the receiver end. The computation of the shared transmission rate is done at the receiver, and the result sent back to the sender for an appropriate action. Since, the majority of the Internet traffic is still TCP based; RRB-SIMD's sending rate is computed in a TCP-friendly way using the square increase multiplicative decrease rule. In addition to packet loss for congestion detection, RRB-SIMD includes a combined average cumulative jitter and delay factor to detect change on the network's available throughput prior to a loss of a packet.

Furthermore, to improve the aggressiveness and responsiveness of a RRB-SIMD flow to change in network's available throughput, a proposed composite filter that better captures the non-linear process of the round-trip time than the current filter used in TCP's implementation, and represents it in a recursive way is used in the implementation of RRB-SIMD. With the composite filter, RRB-SIMD is provided better ability to detect changes in the available network throughput than with the classical TCP's filter.

Live experiments were set up to measure the performance of the proposed composite filter independently to RRB-SIMD. The experiments were conducted using the ping protocol and the open source media streaming protocol live555. The benchmark used, consists of a server and a host located at distant buildings. The host was either probing the server with ICMP echo packets or streaming a video clip from the server. The experiments were conducted during working days and busy hours to emulate the conditions of a busy network. The results from the

experiments show that the composite filter recovers faster than the TCP filter to sudden changes in network's round-trip time delay.

It was also demonstrated through the course of this project that filters with such a capability provide congestion control protocols with the ability to react fast to changes in the network available throughput, resulting in improved delay, jitter and packet loss rate for the video session. These conclusions were based on the use of the TFRC congestion control protocol and two adaptive filters with cumulative sum detection, namely the adaptive Kalman filter and a proposed adaptive TCP filter. The proposed adaptive TCP filter was built using the classical TCP filter enhanced with the cumulative sum detection scheme derived from the adaptive recursive least mean algorithm of Gustafsson.

For RRB-SIMD, the performance evaluation was obtained using various simulations with the network simulator NS2 and trace files derived from real video clips similar to those find on the Internet. RRB-SIMD was compared to the established standard TFRC protocol, and the following conclusions were drawn from the results:

- RRB-SIMD is TCP-friendly, as the shared bandwidth was almost equally distributed between RRB-SIMD and TCP flows.
- Overall, RRB-SIMD has realised a much better performance compared to TFRC in terms of loss frame rate and PSNR results, providing the end user with an image with better quality than TFRC.
- Overall, the delays for both protocol were close and within the acceptance range of any conversational media applications (<150 ms).
- Overall, the average cumulative jitter of RRB-SIMD was lower than the average cumulative jitter of TFRC. With a low average cumulative jitter a short decoder buffer is needed at the receiver to cope with the variation in the packets arrival time.

Moreover, for wireless networks where packet error rate is high, and often due to channel impairments, an adaptive FEC scheme called DS-REAFEC was proposed to enhance the performance of RRB-SIMD. The DS-REAFEC scheme is built independently of RRB-SIMD, and as such, can be used in combination with any other congestion control protocol.

DS-REAFEC is implemented at the base station and uses the traffic load level and the wireless channel state, to decide on the number of FEC packets that should be added along with video data to allow recovery of lost video packets at the destination ends. With the DS-REAFEC scheme, the number of FEC packets is added such that network congestion is minimized, while the end user is still provided with sufficient redundant information to recover from packets loss caused by wireless lines. In this way the quality level of the video clips is still maintains at an acceptable level.

The performance evaluation of the DS-REAFEC was conducted using simulations and, the results were compared to the results of two existing adaptive FEC mechanisms: EAFEC and REAFEC. From the results, it was concluded that, DS-REAFEC is not advisable for wireless networks with low packet error rate (≤ 0.2), since it tends to add a high number of FECs for less improvement in terms of video quality. However, when, the packet error rate exceeds 0.2, DS-REAFEC shows a much better results in term of user-perceived quality than EAFEC and REAFEC. Additionally, it was also noted that on frame by frame basis, DS-REAFEC provides video clips with sudden scenes changes better protection than REAFEC and EAFEC.

6.2 Future Work

There are issues that have arisen during the course of this study with the potential of improving video delivery over the Internet in general and in particular the performance of the proposed RRB-SIMD protocol. However, they have fallen outside of the time limit of the project. One of the issues is the limitation of the Evalvid-RA tool to support wireless transmission of video content over the real-time transport protocol RTP/RTCP. The performance evaluation of RRB-SIMD, with a mechanism that can differentiate loss due to congestion from that due to wireless channel impairments in order to test the performance of the protocol over wireless networks is an aspect that needs to be investigated in future.

The low and high threshold values play a determinant role in the performance of the RRB-SIMD protocol. In this thesis, these values were set manually after many runs of a simulated model. For each run, different values of the low and high threshold were tested, and the couple of low and high threshold values that had realised the best trade-off between achieved throughput and lowest frame loss rate were considered. It is important, in future, to investigate the best possible way to set up the two parameters.

Another area of investigation that is left as a future work is the friendliness of the proposed RRB-SIMD towards other RRB-SIMD flows. Although, the friendliness of the square increase and multiplicative decrease flows towards other square increase multiplicative decrease flows was established using the RB-SIMD protocol, it will still be interesting to test the friendliness result of RRB-SIMD flows towards other competing RRB-SIMD flows. This study is left for the future.

Finally, an interesting aspect of the cumulative jitter scheme that also needs to be investigated is a measurement study on the degree of correlation between the congestion level at the bottleneck and the cumulative jitter. The existing literature is lacking such a study. Since jitter is caused by various factors such as the processing at routers, such a study can only be conducted using a real network. Thus, the need for appropriate equipment is recommended when taking this direction.

References

[1] L. Kleinrock, "History of the Internet and its Flexible Future," *IEEE Wireless Communications*, Vol. 15, No. 1, 2008, pp. 8-18.

[2] L. Kleinrock, "An Early History of the Internet," *IEEE Communications Magazine*, Vol. 48, No. 8, 2010, pp. 26-36.

[3] N. Nimpuno, "Internet Management –Past, Present, Future", in U-connect: 4th International Seminar on The Internet Resources Management and Technical Trends, Almaty, Kazakhstan, Sept. 2006, pp. 12-13.

[4] A. C. Weaver, "The Internet and the World Wide Web", *Proceedings of the 23rd International Conference on Industrial Electronics and Control and Instrumentation (IECON)*, New Orleans, USA, Aug. 2002, pp. 1529-1540.

[5] D. Wu, Y. T. Hou, W. Zhu, Y. Zhang and J. P. Peha, "Streaming Video over the Internet: Approaches and Directions," *IEEE Transactions on Circuits and systems for video technology*, Vol. 11, No. 1, Feb. 2001, pp. 1-19.

[6] Cisco systems, "Cisco Visual Networking: Forecast and Methodology Vision," 2009-2014, white paper, June 2010.

[7] ISO/IEC JTC 1/SC 29/WG 11, "Information technology-coding of audio-visual objects, part 1: systems, part 2: visual, part3: audio," FCD 14496, Dec. 1998.

[8] Al Bovik, "Handbook of Image and Video Processing", *Academic Press Series in Communications, Networking and Multimedia*, 2000, pp. 3-20.

[9] M. A. El-Gendy, A. Bose, and K. G. Shin, "Evolution of the Internet QoS and Support for Soft real-Time Applications", *Proceedings of the IEEE*, Vol. 91, No. 7, July 2003, pp. 1086-1103.

[10] M. Elaoud, and P. Remannathan, "Adaptive use of Error Correcting Codes for Real-Time Communication in Wireless Networks," *Proceedings of INFCOM*, 1998, pp. 548-555

[11] L. Rizzo, "Effective erasure Codes of Reliable Computer Communication Protocols," *Computer Communication review*, Vol. 27, No. 2, 1997, pp. 24-36.

[12] H. Hamdi, J. W. Roberts, and P. Rolin, "Rate Control for VBR Video Coders in Broad-band Networks," *IEEE journal on selected areas in communications*, Vol. 15, No. 6, Aug. 2002, pp. 1040-1051.

[13] A. Lie, "Enhancing Rate Adaptive IP Streaming media Performance with the use of Active Queue Management," PhD thesis, Norwegian University of Science and Technology, Faculty of Information Technology, Mathematics and Electrical Engineering, 2008.

[14] Z. Wang, "Internet QoS: Architecture and Mechanisms for Quality of Service," San Mateo, CA: Morgan Kaufmann, 2001.

[15] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An overview," IETF, RFC 1633, June 1994.

[16] S. Blake, D. Black, M. Carlson, E. Davis, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," IETF, RFC 2475, Dec. 1998.

[17] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guarantee Quality of Service," RFC 2212, Sept. 1997.

[18] J. Wroclawski, "Specification of the Controlled-load Network Element Service," RFC 2211, Sept. 1997.

[19] E. Kusmierek, and D. H. C. Du, "Streaming Video delivery over the Internet with Adaptive End-to-end QoS," *Elsevier Journal of systems and software*, Vol. 75, No. 3, 2005, pp. 237-252.

[20] S. Floyd, M. Handley, and J. Padhye, "A Comparison of Equation based and AIMD Congestion Control", *ACIRI*, May 2002, pp. 1-12.

[21] V. Jacobson, "Congestion Avoidance and Control," *Proceedings of ACM SIGCOMM*, Aug. 1998, pp. 314-329.

[22] S. Floyd and K. Fall, "Promoting the use of End-to-end Congestion Control in the Internet," *Proceedings of ACM/IEEE Transactions in Networking*, May 1999, pp. 458-472.

[23] M. P. Ferrara, M. Fleury, M. Ghanbari, and K. Guild, "A Measurement Study of Packet Loss versus Delay in Congestion Detection for Video Streaming," *Proceedings in IEEE International Conference on Multimedia and Expo*, Aug. 2008, pp. 449-452.

[24] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation based Congestion Control for Unicast Applications," *Proceedings of ACM SIGCOMM*, Aug. 2000, pp. 45-58.

[25] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-Friendly SIMD Congestion Control and its Convergence Behaviour," *Proceedings of the 9th international conference network protocols*, 2001, pp. 156-164.

[26] J. Sivarajah, D. W. Armitage, and N. M. Allinson, "New TCP-Friendly, Rate-Based Transport Protocol for Media Streaming Applications," *Proceedings of IEE*

communication, Vol. 151, No. 3, 2004, pp. 280-286.

[27] D. Chiu and R. Jain, "Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks," *Journal of Computer Networks and ISDN*, Vol. 17, No. 1, June 1989, pp. 1-14.

[28] D. Bansal, and H. Balakrishnan, "Binomial Congestion Control Algorithms," *Proceedings of 20th annual joint conference of the IEEE computer and communication societies, INFOCOM*, 2001, pp. 631-640.

[29] J. Padye, V. Firoio, D. Towsley, and J. Kurose, "Modelling TCP-Reno Performance," A Simple Model and its Empirical Validation," *IEEE/ACM Networking*, Vol. 8, Apr. 2000, pp. 133-145.

[30] T. Stockammer, H. Jenkae, and G. Kuhn, "Streaming Video over Variable Bit-Rate Wireless Channels," *IEEE Transaction on multimedia*, Vol. 6, No. 2, Apr. 2004.

[31] D. Wu, Y. T. Hou and Y-Q. Zhang, "Transporting Real-time Video over the Internet: Challenge and Approaches," *Proceedings of the IEEE*, Vol. 88, No. 12, Dec. 2000, pp. 1855-1875.

[32] S. M. W Group, "Synchronized Multimedia Integration Language (SMIL)," 1.0 *Spec_cati Portal, the ACM digital library*, 2003, pp. 121-123.

[33] N. Feamster, and H. Balakrishnan, "Packet Loss Recovery for Streaming Video," *MIT laboratory for computer science*, Cambridge, MA published in SiteSeer, http://www. Nms.lcs.mit.edu/projects/videocm/, 2003.

[34] Q. Xia, "Rate Adaptation and Resource Allocation for Wireless Networks," PhD thesis, Hong-Kong University of Science and Technology, July 2008.

[35] I. E. G. Richardson, "H.264 and MPEG-4 Video Compression," *John Wiley & Sons Ltd*, England, UK, 2003.

[36] I. E. G. Richardson, "Video Codec Design: Developing Image and Video Compression," *John Wiley & Sons Ltd*, England, UK, 2002.

[37] S. Nagori, and A. Jain, "Introduction to Video Compression Techniques," http://iris.ee.iisc.ernet.in/Resources/docs/MSRIT/Soyeb_Introduction.ppt, accessed November 2008.

[38] A. Lie, and J. Klaue, "Evalvid-RA: Trace Driven Simulation of Rate Adaptive MPEG-4 VBR Video," *Journal of Multimedia System*, Vol. 14, 2008, pp. 33-50.

[39] M. Hamdi, J. W. Roberts, and P. Rolin, "Rate Control for VBR Video Coders in Broad-band Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 6, Aug. 1997, pp. 1040-1051.

[40] Q. Wang, K. Long, S. Cheng, and R. Zhang, "TCP-Friendly Congestion Control Schemes in the Internet," *Proceedings of the International conference on Info-Tech. and Info-net,* 2001, pp. 211-216.

[41] Y. R. Yang, M.S. Kim, and S. S. Lam, "Transient behaviour of TCP-Friendly Congestion Control Protocols", Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001, Vol. 3, 2001, pp.1716-1725.

[42] Y. R. Yang, and S. S. Lam, "General AIMD Congestion Control," *Proceedings of 8th International conference on network protocols (ICNP)*, Osaka, Japan, 2000, Available from http://www.cs.utexas.edu/users/lam/NRL/TechReports/273.

[43] P. Papadimitriou, and C. Zhang, "Optimization of AIMD Congestion Control for Media-Streaming Applications," *Journal of Internet Engineering*, Vol. 1, No. 2, 2007, pp. 71-81.

[44] M. Welzl, "User-Centric Evaluation of TCP-friendly Congestion Control for Real-Time Video Transmission," e & I Elektrotechnik und Informationstechnik, Vol. 122, No. 6, 2005, pp. 221-224.

[45] R. Serral-Gracia, A. Cabellos-Aparicio, and J. Domingo-Pascual, "Packet Loss Estimation using Distributed Adaptive Sampling," *Workshops on networks operation and management symposium*, Apr. 2008, pp. 124-131.

[46] P. Karn, and C. Partridge, "Improving Round Trip Estimates in Reliable Transport Protocols," *ACM Transaction on computer systems*, Vol. 9, No. 4, 1991, pp. 364–373.

[47] E. Kohler, M. Handley, and S. Floyd, "Designing DCCP: Congestion Control without Reliability," *Proceedings ACM SIGCOMM2006*, Pizza, Italy, Sept. 2006, pp. 27-38.

[48] R. Rejaie, M. Handley, and D. Estrin, "RAP: End to End Rate-based Congestion Control Mechanism for Real-Time Streams in the Internet," *Proceedings of IEEE INFOCOM*, Mar. 1999, pp. 1337-1345.

[49] D. Sisalem, and H. Schulzrinne, "The Loss-delay based Adjustment Algorithm: A TCP-Friendly Adaptation Scheme," *Proceedings of NOSSDAV*, Cambridge, UK, 1998, pp. 1-15.

[50] I. F. Akyildiz, O. B. Akan, and G. Morabito, "A Rate Control Scheme for Adaptive Real-Time Applications in IP networks with Lossy Links and Long Round-Trip Times," *IEEE/ACM Transactions on Networking*, Vol.13, No.3, June 2005, pp. 554-567.

[51] H. Lee, and C. Choi, "A Loss Discrimination Scheme for TFRC in Last Hop Wireless Networks," *IEEE Wireless Communications and Networking Conference*, March 2007, pp. 3082-3086.

[52] P. Huang, K. Chu, H. Lo, W. Lee, and T. Wu, "A Novel Adaptive FEC and Interleaving Architecture for H.264/SVC Wireless Video Transmission," *Proceedings of the 5th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Sept. 2009, pp. 989-992.

[53] D. Barman, and I. Matta, "Effectiveness of Loss Labeling in Improving TCP Performance in Wired/Wireless networks," *Proceedings of the 10th IEEE International Conference on Network Protocols*, Nov. 2002, pp. 2- 11.

[54] S. Biaz, and N. Vaidya, "Discriminating Congestion Losses from Wireless Losses using inter-arrival times at the receiver," *IEEE Symposium*, Mar. 1999, pp. 10-17.

[55] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H. Tokuda, "Achieving Moderate Fairness for UDP Flows by Path-status Classification," *Proceedings of 25th Annual IEEE Conference on Local Computer Networks (LCN)*, 2000, pp. 252-261.

[56] S. Cen, P. C. Cosman, and G. M. Voelker, "End-to-end Differentiation of Congestion and Wireless Losses," *IEEE/ACM Transactions on Networking*, Vol. 11, No. 5, Oct. 2003, pp. 703- 717.

[57] O. B. Akan, and I. F. Akyildiz, "ARC: the Analytical Rate Control Scheme for Real-time Traffic in Wireless Networks," *IEEE/ACM Transactions on Networking*, Vol. 12, No. 4, Aug. 2004, pp. 634- 644.

[58] C. H. Lin, "A Self-regulated Redundancy Control Scheme for Wireless Networks," *IEEE ICNS*, July 2006, pp. 106-111.

[59] C. H. Li, C. H. Ke, C. K. Sheih, and N. K Chilamkurti, "An enhanced Adaptive FEC Mechanism for Video Delivery over Wireless Networks," *International conference on networking and services (ICNS)*, July 2006, pp. 989-992.

[60] H. Du, Y. Liu, C. Guo, and Y. Liu, "Research on Adaptive FEC for Video Delivery over WLAN," *Proceedings of the International conference on wireless communication, networking and mobile computing (WiCom)*, Sept. 2009, pp. 1-4.

[61] M. F. Tsai, C. H. Ke, T. H. Wu, C. K. Shieh, and W. S. Hwang, "Burst-aware Adaptive Forward Error Correction in Video Streaming over Wireless Networks," *Proceedings of the 10th IEEE International conference on high performance computing and communications (HPCC)*, Sept. 2008, pp. 625-628.

[62] Ping protocols, http://www.skbuff.net/iputils, accessed May 2010.

[63] S. B. Moon, J. Kurso, P. Skelly, and D. Towesly, "Estimation and Removal of Clock Skew from Network Delay Measurement," *Tech Report98-43,department of Computer Science, University of Massachusetts at Amherst*, MA 01003, 1998.

[64] S. G. Yoo, K .T .Chong, and H. S. KIM, "Development of Predictive TFRC with Neural Network.," PaCT 2005, *Lecture Notes on Computer Science*, 3606, 2005, pp. 193–205.

[65] K. Jacobsson, H. Hjalmarsson, N. Moller, K. H. Johansson, "Round-trip Time Estimation in Communication Networks using Adaptive Kalman Filtering", Available from http://www.s3.kth.se/~kallej/papers/network_rm04.pdf.

[66] K. Jacobsson, H. Hjalmarsson, N. Moller, K. H. Johansson, "Estimation of RTT and Bandwidth for Congestion Control Applications in Communication Networks," *Proceedings of IEEE conference on decision and control*, 2004.

[67] L. Gustafsson, "Adaptive Filtering and Change Detection," Wiley, England, 2000, pp. 1-90.

[68] G. A. Lusilao-Zodi, J. N. Kakande, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "Towards Improved TCP-Friendly Rate Adaptation for Real-time Applications using Adaptive Filters," *Proceedings of the South-Africa Telecommunication Network and Applications Conference (SATNAC),* Spier Estate, South-Africa, Sept. 2010, pp. 293-298.

[69] G. A. Lusilao-Zodi, J. N. Kakande, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "Performance Evaluation of TCP-Friendly Rate Control Enhanced with Adaptive Filters," *International Journal of Advance in Computing Technology (IJACT),* Vol. 3, No. 1, Feb. 2011, pp. 13-22.

[70] R. Camilo, L. Gamez, P. Mart, M. Velasco, J. M. Fuertes, "Wireless Network Delay Estimation for Time-sensitive Applications," research report ESAII-RR-06-12, automatic control department, Technical University of Catalonia, Spain, 2006.

[71] J. But, and G. Armitage "Implementing Encrypted Streaming Video in a Distributed Server Environment," *Proceedings of the ACM SIGCHI International Conference on advances in computer entertainment Technology*, 2005, pp. 322-325.

[72] NS2, LBL, http://isi.edu/nsnam/ns/, accessed Oct. 2010.

[73] C. Bouras, A. Gkamas, and G. Kioumourtzis, "Extending the Functionality of RTP/RTCP Implementation in the Network Simulator (NS-2) to support TCP-friendly congestion control," *Proceedings of the 1st International conference on simulation tools and techniques for communications, networks and systems and workshops*, 2008, pp. 1-6.

[74] X. Chen, and F. Lu, "An Improved Rate Control Scheme for H.264 Based on Frame Complexity Estimation," *Journal of Convergence Information Technology*, Vol. 5, No. 10, 2010, pp. 117-123.

[75] G. A. Lusilao-Zodi, M. E Dlodlo, G. De Jager, and K. L. Ferguson, "Round-Trip Time Estimation in Telecommunication Networks using Composite Expanding and Fading Memory Polynomials," *15th IEEE Mediterranean Electro-technical Conference (MELECON 2010)*, Apr. 2010, pp. 1581-1585.

[76] N. Morrison, "Filter Engineering for Radar and Tracking: Gauss-Newton, Swerling, Kalman and Polynomial," http://www.rrsg.uct.ac.za/tracksmooth.tgz, accessed Feb. 2009.

[77] Live555 Streaming Media, http://www.live555.com/liveMedia, accessed 04 Apr. 2010.

[78] G. A. Lusilao-Zodi, M. E. Dlodlo, G. de Jager and K. L. Ferguson, "RRB-SIMD: RTP Rate-Based SIMD Protocol for Media Streaming Applications over the Internet," *Proceedings of the 9th Annual Conference on Communication Networks and Services Research* (CNSR), Ontario, May 2011, pp. 69-76.

[79] F. Hartanto, and H. R. Sirisena, "Cumulative Inter-ADU Jitter Concept and Its Application," *Proceedings of the 20th International Conference on Computer Communications and Networks*, USA, 2001, pp. 531-534.

[80] L. Tionardi, and F. Hartanto, "The Use of Cumulative Inter-Frame for Adapting Video Transmission Rate," *Proceedings of TENCON*, Bangalore, India, 2003, pp. 364-368.

[81] C. Bouras, A. Gkamas, and Gergios Kiomourtzis, "Adaptive Smooth Multicast Protocol for Multimedia Data Transmission," *International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, 2008, pp 269-276.

[82] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Application," RFC3550, July 2003.

[83] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid: A Framework for Video Transmission and Quality Evaluation," *Proceedings of the 13th International conference on modelling techniques and tools for computer performance evaluation*, Illinois, 2003, pp. 255-272.

[84] FFmpeg, http://sourceforge.net/projects/ffmpeg/, accessed Dec. 2009.

[85] ITU-T Recommendation P. 800.1, "Series P: Telephone Transmission Quality, Telephone Installations, Local Line Networks. Methods for Objective and Subjective Assessment of Quality. Mean Opinion Square (MOS) Terminology", 2003.

[86] M. A. Rohaly, J. P. Corriveau, J. M. Libert, and A. A. Webster, "Video Quality Experts Group: Current Results and Future Direction," *In SPIE Visual Communications and Image Processing*, Perth, Australia, June 2000, pp. 742-753.

[87] K. Park, G. Kim, and M. Crovella, "On the Relationship between File Sizes, Transport Protocols and Self-similar Networks Traffic," *Proceedings of ICNP*, 1996, pp. 171-180.

[88] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson, "Self-similarity through High Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," *Proceedings of ACM SIGCOMM*, 1995, pp. 71-86.

[89] S. P. Chan, and M. T. Sun, "Network Condition Detection for Video Transport over Wireless Internet," *Proceedings of ISCAS*, 2006, pp.3093-3096.

[90] P. Zhang, Y. Jin, H. Shen, and G. Bai, "Performance Study of an Enhanced Adaptive FEC for Wireless Media Streaming," *Proceedings of Wireless Communications, Networking and Mobile Computing (WICOM)*, 2008, pp. 12-14.

[91] F. Zhai, Y. Eisenberg, and T. N. Pappas, "Rate-Distortion Optimized Product Code Forward Error Correction for Video Transmission over IP-Based Wireless Networks," *Proceedings of IEEE ICASSP*, 2004, pp. 857-860.

[92] Q. Zhang, and S. Kassam, "Hybrid ARQ with Selective Combining for fading Channels," *IEEE JSAC*, Vol. 17, No. 5, 1999, pp. 867-880.

[93] M. Zheng, "An Adaptive Forward Error Correction Algorithm for Streaming Video," *Journal of Software*, Vol. 15, No. 9, 2004, pp. 1405-1412.

[94] R. EI Azouzi, T. Peyre, and A. Benslimane, "Optimal Design of Hybrid FEC/ARQ Schemes for Real-Time Applications in Wireless Networks," *Proceedings of ACM WmuNep*, 2006, pp. 11-18.

[95] V. Subramanian, S. Kalyanaraman, and K. K. Raamakrishnan, "Hybrid Packet FEC and Retransmission-based Erasure Recovery Mechanisms (HARQ) for Lossy Networks: Analysis and Design," *Proceedings of WISARD*, January 2007, Bangalore, India (Invited paper), pp. 1-8.

[96] S. Makharia, D. Raychaudhuri, and W. Mingquan, "Experimental Study on Wireless Multicast Scalability using Merge Hybrid ARQ with Staggered Adaptive FEC," *Proceedinds of World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2008, pp. 1-12.

[97] MPEG-4 and H263 Video Traces for Network Performance Evaluation, http://www.tkn.tu-berlin.de/research/trace/trace.html, accessed 4 Nov. 2010.

[98] J. Banks, J. S. Carson, B. L. Nelson and D. M. Nicol, "Discrete-Event System Simulation," Prentice Hall international series in industrial and systems engineering, Prentice Hall edition, 2001, pp. 282-292.

[99] J. Kurose, K. W. Ross, "Computer Networking: A top Down Approach Featuring the Internet," *3rd Edition Addisson-Wesley*, 2005, pp. 565-650.

[100] J. N. Kakande, "Real-time Protocols for Internet Applications", MSc thesis, University of Cape Town, 2010.

Appendix A: Polynomial Filtering

A.1 Expanding Memory Polynomial (EMP) Filters

0^{th} –degree

$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \alpha e_n \qquad \alpha = 1/(n+1)$$

1^{st} – degree

$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + \beta e_n \qquad \beta = 6/(n+2)^{(2)}$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} + \alpha e_n \qquad \alpha = 2(n+2)/(n+2)^{(2)}$$

2^{nd} –degree

$$\hat{z}_{2_{n+1.n}} = \hat{z}_{2_{n.n-1}} + \gamma e_n \qquad \gamma = 30/(n+3)^{(3)}$$
$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + 2\hat{z}_{2_{n+1.n}} + \beta e_n \qquad \beta = 18(2n+1)/(n+3)^{(3)}$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} - \hat{z}_{2_{n+1.n}} + \alpha e_n \qquad \alpha = 3(3n^2+3n+2)/(n+3)^{(3)}$$

3^{rd}–degree

$$\hat{z}_{3_{n+1.n}} = \hat{z}_{3_{n.n-1}} + \delta e_n$$
$$\hat{z}_{2_{n+1.n}} = \hat{z}_{2_{n.n-1}} + 3\hat{z}_{3_{n+1.n}} + \gamma e_n$$
$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + 2\hat{z}_{2_{n+1.n}} - 3\hat{z}_{3_{n+1.n}} + \beta e_n$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} - \hat{z}_{2_{n+1.n}} + \hat{z}_{3_{n+1.n}} + \alpha e_n$$

$$\delta = 140/(n+4)^{(4)}$$
$$\gamma = 120(2n+1)/(n+4)^{(4)}$$
$$\beta = 20(6n^2+6n+5)/(n+4)^{(4)}$$
$$\alpha = 8(2n^3+3n^2+7n+3)/(n+4)^{(4)}$$

where

$$(n+2)^{(2)} = (n+2)(n+1)$$
$$(n+3)^{(3)} = (n+3)(n+2)(n+1)$$
$$(n+4)^{(4)} = (n+4)(n+3)(n+2)(n+1)$$

A.2: Fading Memory Polynomial (FMP) Filters

0^{th} –degree

$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \alpha e_n \qquad \alpha = (1-\theta)$$

1^{st} – degree

$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + \beta e_n \qquad \beta = (1-\theta)^2$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} + \alpha e_n \qquad \alpha = (1-\theta^2)$$

2^{nd} –degree

$$\hat{z}_{2_{n+1.n}} = \hat{z}_{2_{n.n-1}} + \gamma e_n \qquad \gamma = {}^{1}/_{2}\,(1-\theta)^3$$
$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + 2\hat{z}_{2_{n+1.n}} + \beta e_n \qquad \beta = {}^{3}/_{2}\,(1-\theta)^2(1+\theta)$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} - \hat{z}_{2_{n+1.n}} + \alpha e_n \qquad \alpha = (1-\theta^3)$$

3^{rd}–degree

$$\hat{z}_{3_{n+1.n}} = \hat{z}_{3_{n.n-1}} + \delta e_n$$
$$\hat{z}_{2_{n+1.n}} = \hat{z}_{2_{n.n-1}} + 3\hat{z}_{3_{n+1.n}} + \gamma e_n$$
$$\hat{z}_{1_{n+1.n}} = \hat{z}_{1_{n.n-1}} + 2\hat{z}_{2_{n+1.n}} - 3\hat{z}_{3_{n+1.n}} + \beta e_n$$
$$\hat{z}_{0_{n+1.n}} = \hat{z}_{0_{n.n-1}} + \hat{z}_{1_{n+1.n}} - \hat{z}_{2_{n+1.n}} + \hat{z}_{3_{n+1.n}} + \alpha e_n$$

$$\delta = {}^{1}/_{6}\,(1-\theta)^4$$
$$\gamma = (1-\theta^3)(1+\theta)$$
$$\beta = {}^{1}/_{6}\,(1-\theta)^2(11+14\theta+11\theta^2)$$
$$\alpha = (1-\theta^4)$$

B: Confidence Interval [98]

This section discusses the confidence interval of the average number of redundant FEC packets added along with video data. The confidence intervals for the average number of packets loss and the average PSNR can be obtained in a similar way.

Let S_i denote for example the number of FEC packets recording during the nth replication of the simulation. Since the replication are independent, $S_1, S_2, \cdots, S_N$ are independent identically distributed random variables. Denote their mean by μ and variance by σ^2. Then

$$\bar{S}(N) = \sum_{n=1}^{N} S_n/N$$

is an unbiased estimator of μ and

$$Var(N) = \sum_{n=1}^{N} (S_i - \bar{S}(N))^2/(N-1)$$

is an unbiased estimator of σ^2. The $(1-\alpha)$ confidence interval half-length for μ is given by

$$\varphi(N,\alpha) = t_{N-1,1-\alpha/2}\sqrt{Var(N)}/\sqrt{N}$$

Where $0 < \alpha < 1$and $t_{N-1,1-\alpha/2}$ is the $1-\alpha/2$ critical point of the t-distribution with $N-1$ degrees of freedom.

If $\epsilon(N) = |\bar{S}(N) - \mu|/\mu$ denote the relative error in the estimate of μ. The estimate $\bar{S}(N)$ will have a relative error of at most $\phi(N) = \varphi(N,\alpha)/\bar{S}(N) - \varphi(N,\alpha)$ with a probability of approximately $1-\varphi$.

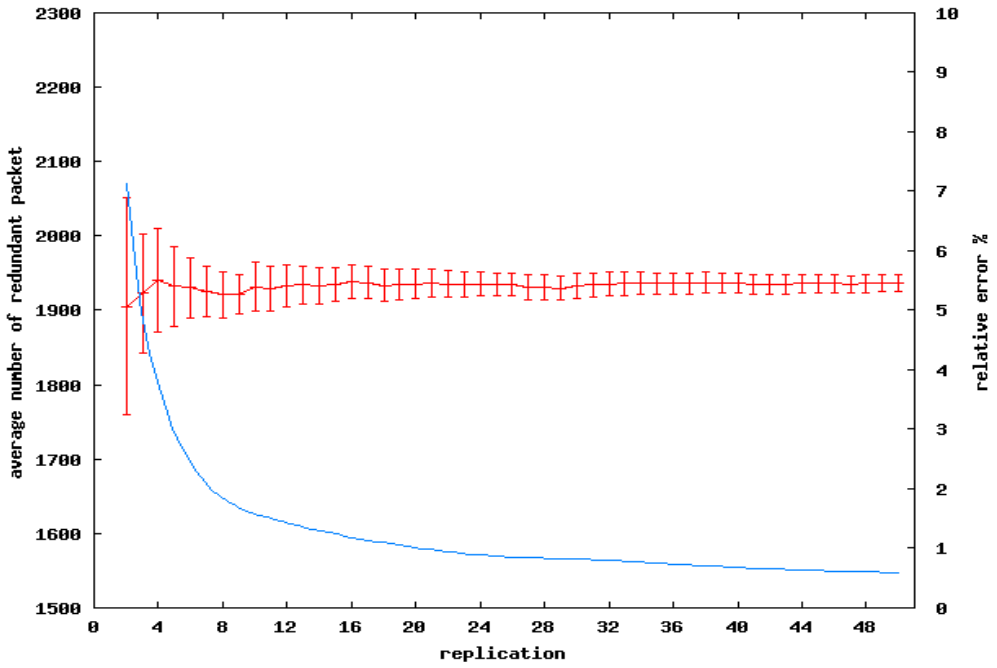

Figure B.1: 95% confidence intervals and the relative error for the expected number of FEC packets as a function of the number of independent replications of the simulation.

Figure B.1 shows the estimates $\bar{S}(N)$ (red line) and the relative errors $\phi(N)$ (blue line) as function of the number N of replications of the simulation with the packet error rate set to 0.4. After 16 replications of the simulation, the average number of FEC packets is $\overline{S}\,(16) = 1939$. The 95% confidence interval half-length $\varphi(16,0.05)$ is 0.017 and the relative error $\phi(16)$ 1.17%.