AN END-TO-END SOLUTION FOR HIGH DEFINITION VIDEO CONFERENCING OVER BEST-EFFORT NETWORKS

BY

ABBAS JAVADTALAB

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the Doctor of Philosophy degree
in
Electrical and Computer Engineering

Ottawa-Carleton Institute for
Electrical and Computer Engineering
School of Electrical Engineering and Computer Science
University of Ottawa

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to Dr. Shervin Shirmohammadi for his guidance and assistance in the preparation of this thesis. In addition, special thanks go to Dr. Mona Omidyegabneh, Dr. Mojtaba Hosseini, and Dr. Abdulsalam Yassine, whose familiarity with the concepts of rate control was helpful during the early programming phase of this project.

May 29, 2014

ABSTRACT

Video streaming applications over best-effort networks, such as the Internet, have become very popular among Internet users. Watching live sports and news, renting movies, watching clips online, making video calls, and participating in videoconferences are typical video applications that millions of people use daily. One of the most challenging aspects of video communication is the proper transmission of video in various network bandwidth conditions. Currently, various devices with different processing powers and various connection speeds (2G, 3G, Wi-Fi, and LTE) are used to access video over the Internet, which offers best-effort services only. Skype, ooVoo, Yahoo Messenger, and Zoom are some well-known applications employed on a daily basis by people throughout the world; however, best-effort networks are characterized by dynamic and unpredictable changes in the available bandwidth, which adversely affect the quality of the video. For the average consumer, there is no guarantee of receiving an exact amount of bandwidth for sending or receiving video data. Therefore, the video delivery system must use a bandwidth adaptation mechanism to deliver video content properly. Otherwise, bandwidth variations will lead to degradation in video quality or, in the worst case, disrupt the entire service. This is especially problematic for videoconferencing (VC) because of the bulkiness of the video, the stringent bandwidth demands, and the delay constraints. Furthermore, for business grade VC, which uses high definition videoconferencing (HDVC), user expectations regarding video quality are much higher than they are for ordinary VC. To manage network fluctuations and handle the video traffic, two major components in the system should be improved: the video encoder and the congestion control.

The video encoder is responsible for compressing raw video captured by a camera and generating a bitstream. In addition to the efficiency of the encoder and compression speed, its output flow is also important. Though the nature of video content may make it impossible to generate a constant bitstream for a long period of time, the encoder must generate a flow around the given bitrate.

While the encoder generates the video traffic around the given bitrate, congestion management plays a key role in determining the current available bandwidth. This can be done by analyzing the statistics of the sent/received packets, applying mathematical models, updating parameters, and informing the encoder. The performance of the whole system is related to the in-line collaboration of the encoder

and the congestion management, in which the congestion control system detects and calculates the available bandwidth for a specific period of time, preferably per incoming packet, and informs rate control (RC) to adapt its bitrate in a reasonable time frame, so that the network oscillations do not affect the perceived quality on the decoder side and do not impose adverse effects on the video session.

To address these problems, this thesis proposes a collaborative management architecture that monitors the network situation and manages the encoded video rate. The goal of this architecture is twofold: First, it aims to monitor the available network bandwidth, to predict network behavior and to pass that information to the encoder. So encoder can encode a suitable video bitrate. Second, by using a smart rate controller, it aims for an optimal adaptation of the encoder output bitrate to the bitrate determined by congestion control.

Merging RC operations and network congestion management, to provide a reliable infrastructure for HDVC over the Internet, represents a unique approach. The primary motivation behind this project is that by applying videoconference features, which are explained in the rate controller and congestion management chapter, the HDVC application becomes feasible and reliable for the business grade application even in the best-effort networks such as the Internet.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ABR	Average Bitrate
ADSL	Asymmetric Digital Subscriber Line
AIMD	Additive Increase Multiple Decrease
AVC	Advanced Video Coding
BAM	Bitrate Adjustment Mechanism
CBR	Constant Bitrate
CRF	Constant Rate Factor
CQP	Constant QP
DASH	Dynamic Adaptation Streaming over HTTP
DCRF	Dynamic CRF
DRC	Dynamic Rate Control
ECN	Explicit Congestion Notification
FPS	Frame Per Second
GCC	Google Congestion Control
GOP	Group of Pictures

HD	High Definition
HDVC	High Definition VideoConferencing
ICME	International Conference on Multimedia and Expo
I-frame	Inter Frame
ISP	Internet Service Provider
IP	Internet Protocol
JM	Joint Model
LBC	Local Bandwidth Calculator
LTE	Long-Term Evolution
MAD	Mean Absolute Difference
MRQ	Maximum Required Quality
NADA	Network Assisted Dynamic Adaptation
NAL	Network Abstraction Layer
OTT	One-way Transmit Time
OWD	One Way Delay
PCR	Program Clock Reference
PDF	Probability Density Function

P-frame	Predicted Frame
PSNR	Peak signal-to-Noise Ratio
QP	Quantize Parameter
QoS	Quality of Service
RC	Rate Control
RD	Rate Distortion
RTCP	RTP Control Protocol
RTP	Real-Time Protocol
RTT	Round Trip Time
SDVC	Standard Definition VideoConferencing
SD	Standard Definition
SLOPS	Self-Loading Periodic Stream
SSIM	Structural Similarity
TBQ	Token-Bucket Queue
TFRC	TCP-Friendly Rate Control
TC	Traffic Control
TCP	Transmission Control Protocol
TOPP	Trans of Packet Pair
UDP	User Datagram Protocol

VBV	Video Buffer Verifier
VC	Video Conferencing
VoIP	Voice over IP
VPN	Virtual Private Network

CHAPTER 1

INTRODUCTION

The use of multimedia communications, especially video applications, continues to increase. Watching live sports, movies, and clips from various video service providers, such as YouTube and Netflix, and VC are typical usages of video over the Internet. According to ComScore's 2009 report, people across the globe streamed 41% more video in August 2009 than in August 2008 [1]. One of the primary challenges of video communication is the proper transmission of video in different bandwidths. Recently, various devices with different processing powers and different network connections have been used to access video via the Internet, which offers best-effort services only [2]. This means that for the average consumers, there is no guarantee that an exact amount of bandwidth will be used for sending or receiving video data. Hence, the video delivery system should use a bandwidth adaptation mechanism to deliver the video content in a way that avoids bandwidth variations to cause a significant degradation in video quality or to interrupt the entire service. This issue is particularly significant for VC because of the bulkiness of video and its stringent bandwidth demands. Furthermore, because VC applications are live, they are very sensitive to delay, and users have little tolerance for interruptions or poor video quality.

Current VC systems can be categorized in two major groups: standard definition (SD) and high definition (HD). SD is one of the first standards and has been used for decades. It defines the acceptable level of quality for small video monitors, especially those less than 27 inches [3]. Although this standard is rather old, SD-based devices offer reasonably sharp and smooth pictures; however, when the size of display exceeds 27 inches, the SD standard is unable to satisfy customer requirements, and it is easy to perceive degradation in the quality of the SD video utilizing the 480i format. Users who compare SD video quality on a small monitor with that on a large monitor may assume that the large monitor has a hardware problem because the displayed picture does not appear realistic and has some negative effects, such as jagged lines, blurry outlines, washed-out colors, visual noise, and choppy movements.

In contrast, HD video applications provide better quality, decrease eye fatigue, and show more readable text, especially on a remote whiteboard. High definition videoconferencing (HDVC) is defined as VC in which the video quality is high definition, i.e., 1920 x 1080 pixels (or higher) frame size at 30 frames (or higher) per second progressive scan, commonly referred to as 1080P30. From a user's perspective, HDVC significantly improves the quality of VC in terms of presence, awareness, realism, and details,

providing a rich user experience through detailed facial expression, body language, visual clarity, and fidelity. The HDVC market has recently become significant and is growing rapidly due to the technology's efficiency, practicality, and convenience as well as the economic downturn and security concerns with traveling. Typical customers include hotels, law firms, remote development firms, construction firms, and companies with geographically distributed branches. Current HDVC solutions include Cisco's Telepresence [4], Logitec's Lifesize [5], Polycom's Telepresence [6], and Magor's Telecollaboration [7] line of products.

HDVC requires high bandwidth to operate properly. Therefore, to maintain a constant level of quality, a minimum bandwidth should be guaranteed for a required quality of service (QoS), which is why most current HDVC systems require dedicated networks with a guaranteed QoS [8]; however, requiring a dedicated network affects the flexibility of the system (e.g., from infrastructure availability in all locations) and increases the total cost of the solution (the base model of Cisco Telepresence costs 300K USD, whereas a typical model of the Magor solution costs around 70K USD). By accepting some degrees of flexibility on network bandwidth fluctuations, we can decrease the cost of the network at the expense of not getting good quality all the time. This tradeoff can be optimized if the solution is smart enough to make efficient use of the Internet's best-effort services. Consequently, HDVC can be more practical and more widely deployable if it uses regular high-speed (albeit non-dedicated) Internet connectivity. Internet service providers (ISPs) offer best-effort services with high-bandwidth connectivity that are considerably cheaper than dedicated networks and offer average bitrate instead of dedicated or guaranteed bitrate, with certain restrictions in generating burst data. Recently, some companies, such as Magor Communication, have shown the feasibility of using HDVC over non-dedicated high-speed Internet. To become feasible in this way, HDVCs should address two concerns:

- A rate control that can generate video traffic according to the network available bandwidth.
- A network congestion control, which predicts any upcoming modification on the available bandwidth, calculates available bandwidth, and passes that information to the rate controller, which can then take the appropriate actions.

From the video encoder point of view, the rate control (RC), which adjusts the bitrate of the video, should satisfy the following requirements:

- Make optimal use of the best-effort network; i.e., it should use the available bandwidth as much as is required for high-motion frames and avoid wasting network bandwidth when there is little activity in the video.

- Choose optimum point of quality and bitrate of the video, based on the current available bandwidth and packet loss, which changes dynamically in a best-effort service.

These essential features should be supported by a system that tries to satisfy end users in terms of the video quality.

From the network perspective, HDVC systems can use two different approaches for video streaming: dedicated networks and best-effort networks. Dedicated networks offer reliable infrastructure; however, the customer has to purchase and maintain a dedicated network with a guaranteed QoS for all locations involved in the videoconference on top of the HDVC system. In this case, the network is always available, and there is no need to have a sophisticated congestion control to calculate the available bitrate. Based on the difficulties mentioned before, this is not an option in our research. The other option is best-effort networks. Because HDVC uses best-effort networks such as regular high-speed Internet, the operational cost is cheap and can be easily maintained. But the best effort networks pose the problem of the dynamically changes of the available bandwidth at any given time [2]. In this case, the congestion control plays a key role in success of each HDVC solution. Considering that the video quality is one of the most important factors in HDVC in terms of user experience, available bandwidth variations make the transmission of the HD video challenging. If the available bandwidth decreases, the video bitrate must be quickly adjusted to match it. Otherwise, data will be lost, and the end user will see degraded video. On the other hand, once the available bandwidth increases again, the video bitrate has to quickly adjust to that as well, to ensure that the highest possible quality and fidelity of the video is being presented to the user.

1.1 RESEARCH MOTIVATION

As mentioned in the introduction, the rate and congestion control systems are two critical components for each video conferencing solution. It is useful to know where the best effort concept comes from and why this network called "best effort", then we can get better idea of how to deal with this network and how we can make it useful for HDVC applications.

Historically, IP networks have always been known as “best effort” networks [9]. This technical term means that all IP packets are treated in the same fashion. This type of network attempts to deliver every packet as quickly as possible but makes no priority regarding to the type of service required. This may seem to be a fair approach for many applications. But for certain applications, such as multimedia applications, the “best effort” behavior simply is not good enough. In contrast, it is usually preferred that IP networks provide a different transmission policy to support certain classes of applications in some special cases.

"Better than best effort" is a more suitable way to describe this type of quality of service. Voice over IP (VoIP) and videoconferencing are two examples.

Still, the dominant network of the internet infrastructure is the "best effort" network because of the economic concerns, which dictates a decrease in the cost of communication at the expense of occasionally losing or delaying some packets. This is not a major issue for many applications. For example, a TCP-based application easily manages this situation by resending the lost packets again [10]. There is a major issue with UDP-based applications in which there is no built-in congestion detection/avoidance mechanism in the protocol [11].

The delay and the loss are important indications and they indicate important data about the network. When a packet is delayed in the network, it means that the buffers of the intermediate devices between the source and the destination (routers, switches, etc.) are becoming full, so the packets must wait to be delivered. When a packet is lost, it means that the buffers are already full and cannot accept any new packet. A congestion control/detection method in TCP-based applications analyzes those incidents (delay, loss, etc.) and takes the proper actions by retransmitting the lost packets and decreasing the bitrate. Since there is no such mechanism in a UDP-based application, the application itself should manage those parameters. A TCP friendly rate control (TFRC) method is one successful solution. Currently, almost all video conferencing solutions have some types of components in the system to manage network indications. A videoconferencing session is composed of key elements, and the block diagram of the system is depicted in Figure 1.

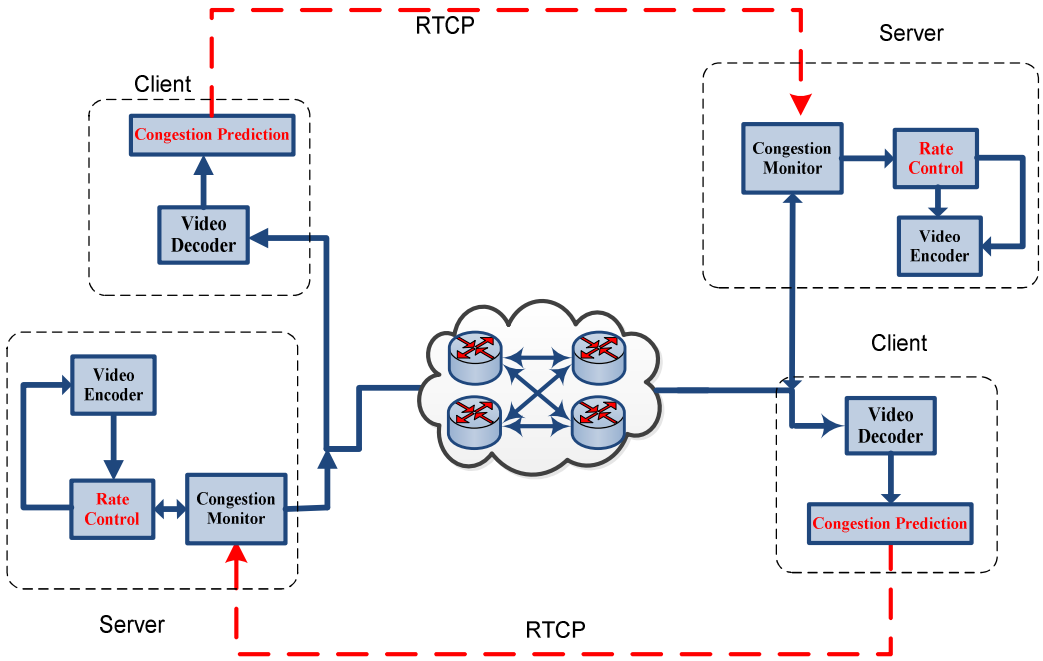


Figure 1: Video Conferencing solution.

Each node has both a client and a server module. The following is a brief description of some key elements:

- The video encoder is responsible for encoding raw video into the compressed bitstream, which is called the network abstraction layer (NAL) units, and for streaming it over the network via real-time protocol (RTP) packets.

- Congestion prediction is located at the decoder-side and attempts to obtain the status of the network (e.g., delay, jitter, and loss) in order to detect the current traffic, predict upcoming congestion, and send notifications to the congestion monitor via real-time control packets (RTCP).

- The congestion monitor receives RTCP packets and calculates the available bandwidth. It determines the encoder bitrate and informs the rate controller, and then the rate controller makes the appropriate adjustments.

- (Rate Controller) RC is responsible for adapting the bitrate of the encoder according to the information provided by congestion control. RC should optimize the rate and quality and

selects an optimum point.

- The congestion detector is used to calculate the congestion parameters (e.g., delay, jitter, and loss), in order to detect the current network traffic, predict upcoming congestion, and send notifications to the congestion prediction and the congestion monitor unit.

- The video decoder reassembles and de-packetizes the incoming packets and presents the video. It also sends the packet statistics to the congestion detector unit.

Managing two main component of the system, RC and congestion prediction, and offering an efficient collaboration between them are the main motivations of this study.

1.2 PROBLEM STATEMENT

Our goal is to provide an efficient end-to-end solution to improve user satisfaction with HDVC over non-dedicated network. To do so, we focus on two major areas: congestion prediction and RC.

First, we analyze the network parameters, such as delay, jitter, and loss to obtain an accurate knowledge of the network condition, which helps to predict the network behavior, and subsequently send our expectation value of the network behavior to the congestion control. Second, we propose an agile rate controller that works closely with the congestion control component to adapt the video bitrate by offering the best possible video quality for the current network situation. The rate controller should provide the following features [12]:

- Use the videoconferencing features and define a proper duration for collecting history. For example, a bitrate used in the previous 30 minutes is not a sufficient reference for calculating the average bitrate. The VC is an ongoing video communication and is different from other video applications, such as video on demand or live videos. In these applications, the videos have been encoded before streaming while in VC this is not the case.

- Allocate the bitrate efficiently. In HDVC, it is necessary to manage the bitrate efficiently because the amount of the transmitted data is high and requires sufficient network resources. Also, HDVC users do not accept low-quality video. Thus, the rate controller should reduce the bitrate quickly when there are few activities during the video in order to preserve the average network usage threshold and assign the bandwidth for high-activity frames instantly in order to preserve the video quality.

- Design the rate controller according to the burst limitation. Although the network may accept the burst, the rate controller must generate uniform traffic and decrease the amount of burst as much as possible; otherwise, some packets may be dropped, or the service provider may apply extra charges to transmit the non-uniform burst traffic.

- Assign the best possible video quality from the beginning. For HDVC, it is very important to provide high video quality as much as possible from the very beginning to motivate participants to use the HDVC system. Because HDVC is often used for business purposes, it creates a negative impression if users receive low-quality video at the beginning and may discourage them from using the system again.

A congestion prediction system should have the following features:

- It should work based on the available packet information, known technically as the drop tail, and it should not impose a small overhead on the system in comparison to the content bitrate itself.

- It should have per-packet accuracy and analyze the network in real time in order to predict the future congestions. In other words, it should be fast enough, so that the whole system (the rate controller and the congestion control system) could adapt to its bitrate intentionally.

Since we do not want to impose much overhead on the system, the congestion prediction system uses information about the video packets themselves as an input such as loss rate, inter-arrival delay, round trip delay. The video packets are known to have a heavy-tailed distribution [13]; this feature should be considered. Otherwise, the system cannot measure the network behavior efficiently.

1.3 MAIN CONTRIBUTIONS

In this study, we have used different approaches to design a suitable rate controller. The work begins in two interrelated areas: congestion prediction and RC.

First, as will be discussed in detail in Chapter 4, we modeled the network behavior and designed a congestion prediction mechanism over the Internet. The main features of our congestion prediction mechanism can be categorized as follows:

- We have proposed a continuous one-way detection method that uses an inter-arrival delay of the received video packets to predict the available bandwidth changes for video streaming over best-

effort networks. We have shown that our method can provide an accurate prediction of the bandwidth changes; this is rather significant because packets travel via multiple paths, and it is very challenging to provide an accurate prediction of packet latency using conventional two-way methods, such as real-time control protocol (RTCP) and the propping schemes.

- We have proposed a per-packet updating scheme at the receiver-end for a fast bandwidth fluctuation prediction and measurement; this is essential in real-time applications like VC. It is important to detect bandwidth changes as quickly as possible to lessen the degradation of the quality at the receiver end due to a packet loss resulting from bandwidth fluctuations. The loss of video packets not only affects lost frames, but also affects all ensuing frames that use the lost frames as a reference due to inter-frame compression.

- We have proposed a realistic model using video data, which has a heavy-tailed distribution [13], to predict congestion. We have modeled the traffic parameters using a Pareto distribution [14] that includes the central component as well as the outliers of the empirical distribution simultaneously. This allows us to use a Bayesian statistical analysis to calibrate the data and rigorously monitor the network conditions. The proposed Bayesian mechanism not only improves the currently available control methods but also can be implemented in other real-time applications if necessary.

- Our proposed method uses arrival packets as a measurement tool and considers only the inter-arrival delay of the packets, which means that our approach does not introduce network overhead to already bulky video traffic that operates at the border of the available bandwidth. Also, our method does not require synchronous clocks at the receiver or the sender-side and is independent of the underlying network [15]. Unlike existing approaches that use explicit congestion notification (ECN), our method considers the network to be a black box, and avoids the need for data collection within the Internet, and is entirely end-to-end.

- We have provided extensive experimental simulations and real-world network implementations [16]. We have tested our method using one of the most demanding video streaming application: HDVC.

Second, we have designed and proposed a smart rate controller called a dynamic rate controller (DRC). It is the successor of the original RC mechanism, called the dynamic constant rate factor (DCRF), which is discussed in Chapter 3.1. The DRC benefits the VC features and uses them to improve its performance. The main features of the DRC are as follows:

- We have proposed a dynamic rate control (DRC) algorithm that can adjust the bitrate of the video within 2 to 6 frames, which is faster than existing RC algorithms. This means that the proposed dynamic rate controller (DRC) is very quick in preventing unwanted side effects on video quality by quickly applying network bandwidth changes to the video encoder. Also, it assigns the best possible video quality from the very beginning. A low video quality at the beginning of a session may discourage participants from continuing.

- Our DRC algorithm provides a balance between expected quality and available bandwidth, which means that the proposed DRC uses network resources efficiently. In our design, if the quality expected by the user is achieved, then there is no need to consume additional bandwidth to increase video quality even if the current bandwidth would allow for it. Furthermore, the DRC sends almost nothing when there is no movement in the video, whereas the competitors send a noticeably higher bitrate.

- The design of our RC introduces the novel concept of allocating a budget, called a future budget, for a current frame, which provides a configurable degree of flexibility to accommodate any sudden change in video scenes and maintains consistency in the overall video quality. In this regard, the proposed algorithm tolerates random bitrate oscillations while maintaining video quality gracefully. This concept offers two significant advantages: First, DRC consistently maintains video quality without surpassing the target video bitrate. Second, during low video activity, the video bitrate is reduced in order to save the bandwidth for potential upcoming high-activity frames.

- We have used moving windows to limit the effect of past frames on the current frame, which means that a burst that occurred previously does not have an effect on the current bit allocation of the current frames. For instance, the amount of bitrate consumed in the previous 30 minutes is not a suitable reference for calculating the average bitrate, because the network conditions are too dynamic to provide a long-term history. Along with the future budget concept, the use of moving windows for a short-term bitrate history allows our algorithm to be more tolerant to video scene fluctuations.

- Our proposed algorithm uses the allowable burst frequency and duration given by the network manager to determine the amount of tolerance in order to maintain optimal video quality for high-activity frames; this means that the frequency of the burst and its duration generated by the DRC can be modified and set. Although bursts are occasionally accepted by ISPs in best-effort networks, the burst duration must be limited according to the policy of ISPs or can be mitigated by small amount of buffering. Moreover, the burst amount should be decreased as much as possible. Otherwise, some packets may be dropped, or the ISP may apply extra charges to transmit non-uniform burst traffic.

We discuss RC and the congestion prediction in Chapters 3 and 4, respectively.

1.4 ROAD MAP

The remainder of the thesis is organized as follows. In Chapter 2, the motivation for RC and the congestion prediction is discussed. Chapter 3 highlights the design of the two RCs implemented for HDVC (DRCF and DRC), and Chapter 4 explains the design of the proposed congestion prediction system. Chapter 5 presents the simulation setup and the experiment results. The thesis concludes with a discussion of avenues for future research.

1.5 SCHOLASTIC ACHIEVEMENTS

The research conducted for this thesis has so far produced a variety of scholastic achievements and publications, which are as follows:

Journal papers

1. A. Javadtalab, M. Semsarzadeh, A. Khanchi, S. Shirmohammadi, and A. Yassine, "Continuous One-Way Detection of Available Bandwidth Changes for Video Streaming over Best Effort Networks," *IEEE Transactions on Instrumentation and Measurement*, 2014 (Accepted).

2. A. Javdtalab, S. Shirmohammadi, M. Omidyeganeh, A. Yassine, and M. Hosseini, "A Dynamic Rate control Algorithm for H.264 High Definition Video Conferencing," *ACM Transactions on Multimedia Computing, Communications and Applications* (Revision in review).

Patents

3. A. Javadtalab and S. Shirmohammadi, "Dynamic Rate Control Algorithm for High Definition Video Conferencing," US Patent Pending 15265-000018/US, Oct 1, 2012.

4. A. Khanchi, M. Semsarzadeh, A. Javadtalab, and S. Shirmohammadi, "Network Congestion Prediction," US Patent Pending 61/639,531, April 29, 2013.

Conference papers related to thesis

5. A. Khanchi, M. Semsarzadeh, A. Javadtalab, and S. Shirmohammadi, "Continuous One-Way Available Bandwidth Change Detection in High Definition Video Conferencing," in ACM

Workshop on Network and Operating Systems Support for Digital Audio and Video, 2013, pp. 25–30.

6. M. Hemmati, A. Javadtalab, A. A. N. Shirehjini, S. Shirmohammadi, and T. Arici, "Game as Video: Bit Rate Reduction through Adaptive Object Encoding," 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), Oslo, Norway, 2013.

7. A. Javadtalab, M. Omidyeganeh, S. Shirmohammadi, and M. Hosseini, "A Rate Control Algorithm for X264 High Definition Video Conferencing," Proc. Workshop on Acoustics and Video Coding and Communication, in Proc. IEEE International Conference on Multimedia & Expo, Barcelona, Spain, 2011.

8. A. Javadtalab, L. Abbadi, M. Omidyeganeh, C. Adams, S. Shirmohammadi, and A. El Saddik, "Transparent Non-Intrusive Multimodal Biometric System for Videoconference Using the Fusion of Face and Ear Recognition," Proc. Annual Conference on Privacy, Security and Trust, Montreal, QC, Canada, 2011, pp. 87–92.

9. A. Javadtalab, M. Omidyeganeh, S. Shirmohammadi, and M. Hosseini, "On the Suitability of Current x264 Rate Controller Algorithms for High Definition Video Conferencing," in International Symposium on Artificial Intelligence and Signal Processing (AISP), 2011, pp. 107–112.

Demo

10. A. Javadtalab, S. Shirmohammadi, and M. Hosseini, "Demo Paper: A Fast-Adjusting High-Quality Rate Control Algorithm for HD Video Streaming," in IEEE International Conference on Multimedia and Expo, 2013, pp. 3–4.

11. A. Javadtalab, X. Zhu, and S. Shirmohammadi, "A Fast-Adjusting Rate Control Algorithm Using Network-Assisted Scheme for HD Video Streaming," Proc. IEEE International Symposium on Multimedia, Taichung, Taiwan, December 10–12, 2014.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 RATE CONTROL

The video bitrate is controlled by the RC algorithm in the codec and used to adjust the bitrate of the video. Existing RC algorithms work relatively well for video-on-demand services, such as YouTube, in which buffering and video interruption is somewhat tolerable, or for standard definition (SD) VC applications, in which occasional degradation of the video is not considered as a major failure. However, they have certain shortcomings that make them unsuitable for HDVC systems. As it is shown in this chapter, they do not allow fine-grained control over the video bitrate, do not adjust the video bitrate to the bandwidth variations quickly enough, or do not produce optimal video quality for a given available bandwidth.

In comparison to SD video streaming and conferencing, HDVC consumes more bandwidth and is also associated with higher user expectations, because the video is displayed on larger screen sizes, typically more than 27 inches [3], which make it easy to observe degradations in the video quality. Moreover, HDVC users have little tolerance for interruptions during the video session. In addition to the participants, HDVC sometimes shows whiteboard information, and the text on the board must be readable for the remote viewers, which demands a certain level of video quality. Finally, HDVC sessions are specific, live applications with very low end-to-end delay thresholds [17][53] of ideally 100ms no more than 200ms; hence, although buffering or other techniques that cause delays, such as retransmission of lost data, work on demand systems, they are impractical for HDVC.

A typical HDVC session consists of remote users participating in a conference in which some participants use a whiteboard. The background is mostly fixed. In addition, users can talk to one another, share documents, share a desktop, and write on the remotely viewable whiteboard. The face-to-face connection becomes very important, and HDVC solutions must provide a degree of realism in which a user is able to "read" other users based on their facial expressions. For example, a high-ranking sales representative of a company may want to recognize changes in the facial expressions of the remote party upon receiving a price offer in order to determine whether the price is too high or too low. Eye contact,

facial expressions, and even slight twitches and other reactions must be visible in these scenarios. Due to this high level of fidelity in HDVC, as soon as the available bandwidth decreases, the information must be received by the sender immediately to decrease the video bitrate to match the available bandwidth. Otherwise, there will be higher degradation in video quality at the receiver-end due to lost packets. Large buffering or other non-live techniques such as pausing the video, requesting and waiting for a refresh (key) frame are not practical for latency reasons. Also, because HD is bulkier than SD (the typical required bandwidth for SD is 500 Kbps, whereas for HD starts from 1.5 Mbps), this problem is more pronounced in HDVC than in SDVC. If during an HDVC session the available bandwidth change from 4 Mbps to 3 Mbps and the sender is still sending the video at 4 Mbps, the video will have a low quality for the receiver-end, and a sensitive event, such as contract negotiation or a demonstration of an important point, happening at that moment will be missed. For SDVC, bandwidth changes from 4 Mbps to 3 Mbps will not have much effect on the session because the video bitrate is probably no more than 1 Mbps. HDVC is more sensitive to such changes because its video bitrate is usually around the border of the available bandwidth. The converse is also true. If the bandwidth goes back from 3 Mbps to 4 Mbps, valuable resources are wasted because the sender is still sending the video at the lower quality of 3 Mbps. Therefore, the video bitrate should be increased to 4 Mbps to maximize the quality.

HDVC applications are therefore different from standard definition videoconferencing (SDVC) or watching a movie, even in HD mode, which leads to specific requirements for the RC algorithms in an HDVC session:

1. They should produce an average video bitrate no more than the current available bandwidth.

2. They should not use long-term bitrate history. For instance, the amount of bitrate consumed in the previous 30 minutes is not a suitable reference for calculating the average bitrate, because the network conditions are too dynamic to provide a long-term history.

3. They should assign bitrate efficiently. For example, the video bitrate should be reduced during low-activity frames in order to reduce bandwidth usage for potential upcoming high-activity frames. In this case, RC should keep track of the quality as well as the bitrate. So if the required quality is achieved, RC should not increase the bitrate. This approach gives RC enough flexibility to maintain video quality in case of sudden movements in the scene.

4. They should manage burst generations. Although bursts are occasionally accepted by ISPs in best-effort networks, the burst duration should be limited according to the policy of ISP.

Moreover, the burst amount should be controlled; otherwise, some packets may be dropped or delayed, which is considered as dropped in the session, or the ISP may apply extra charges to transmit non-uniform burst traffic.

5. They should assign the best possible video quality from the very beginning. Low video quality at the beginning of a session may discourage participants from continuing the session.

All rate controllers use the rate distortion (RD) model. The RD model --[18] ,[19], [20], [21], [22], and [23]-- in H.264/AVC is the upgraded version of RD implemented from the previous standards, such as MPEG-4 and H.263. The standard is responsible only for determining the bitstream syntax and the decoder process; it does not provide any particular specification for encoders [24] [25] [26] [27]. This flexibility offers an opportunity for developers to create their own RD with a specific efficiency. Thus far, many companies have implemented the commercial version of the H.264 encoder in software or hardware [28]. The x264 encoder [29] is one of the open source implementations gathering much attention and is incorporated in many popular applications, such as FFDShow, FFmpeg, MEncoder, and YouTube. In addition, its performance is reported to be near to or in some cases even better than some commercial applications [30].

RD efficiency is tuned by many parameters existing in the frames and macroblocks [31], such as frame types (I, P, or B), number of frames, macroblock modes (e.g., INTRA, INTER, or SKIP), and motion estimation methods (e.g., full search and three step search). RC, in technical terms, must configure those parameters to reach the determined target bitrate or quality. The rate controller can be employed at different levels of encoding, called granularity levels, such as groups of pictures (GOPs), frames, or even macroblocks.

Based on different granularity levels, researchers have tried to develop new ideas. At the macroblock level, Shuijiong et al. proposed a model based on the mean absolute difference (MAD) for H.264 video [32] [33] [34]; the model is updated on the basis of temporal ordering. The main issue is that it does not offer a closed solution, although it can be used as a complement to the other rate controllers.

At the frame level, Lee et al. [35] proposed an RC to manage the hierarchical B-frame encoding. They developed an initial quantization parameter (QP) calculation method at the frame level by designing a frame level bit allocation. The results are suitable for offline video encoding but not necessarily for a live session. In [36], Wang et al. proposed frame level RC using game theory concepts. Their work is also suitable for offline videos. A similar concept has been developed ([37], [38], and [39]) for temporal scalable

video coding, frame skip, and MAD ratio.

At the GOP level, Wu et al. [40] attempted to provide an optimal value for QP to maintain video-quality consistency between two consecutive GOPs. They used five video sequences to develop their method and to create a 2D matrix used as a lookup table. This is similar to the DCRF method (which is discussed in the next chapter) [41], in which the CRF value is predicted. For offline video, the I-frame is an issue because its size is larger than the P-frame, so RC may need to deal with this problem and try to optimize it [42]. Similar work has been done in [43] focusing on a cloud-based environment. Instead of focusing on one granularity level and trying to optimize RC based on it, it is possible to optimize RC based on all granularity levels. For example, Jiang et al. [44] proposed an RC for low-delay real-time coding using all granularity levels to create a low-delay RC; however, their RC does not adapt itself to the network bandwidth constraints such as bandwidth fluctuations.

Other researchers have proposed suboptimal solutions for RC. For example, Chung-Ming et al. [45] proposed an RC that can be applied to audio and visual contents simultaneously. Because users are more sensitive to audio than video, more audio packets than video packets should be sent. Moreover, by using a conditional transmission method for lost packets, they showed that the overall result is more tolerable for users; however, their method does not offer a complete solution that can reach the given bitrate and jumps to the new bitrate.

The other approach to design RC involves focusing on the region of interest [46] [47] [48]. Here, instead of the whole frame, a subset of the frame is encoded with higher bitrate, and the rest of the frame is encoded at a lower bitrate (e.g., the face [49] [50] [51] [52]). These methods are orthogonal to the work presented in this thesis as they can be combined.

The end-to-end solution proposed by Zhang et al. [53] offers a practical video communication solution with some modifications on the rate allocation system on the server-side and adaptive playback adjustment on the receiver-end. The solution enables clients to receive the HD video stream over the network by taking the advantage of the program clock reference (PCR) embedded in the video stream to control the transmission rate and to reduce the client buffer requirements. Their solution is suitable for the cable/satellite TV or the video on demand over the cable/satellite network, in which the network bandwidth is high most of the time and fixed during the video session. Their method can also allow a small time-shift delay during video streaming, but it cannot be applied to HDVC over the Internet because it is not guaranteed to be within the available network bandwidth or acceptable delay duration. In [54], The main objective of their work was to manage packet loss and network throughput in the wireless network rather

than the quality of the received frame at the receiver-end. They have experimentally shown that the packet loss ratio is not monotonically increased by the rate all the time.

In short, no existing method proposes all five of the features mentioned in this study. This was the main reason behind our work and our development of the HDVC-specific RC. We analyze the current rate controllers in x264 in detail in the following sections as they represent some of the common RC methods available today.

2.1.1 RATE CONTROL IN X264

x264 and the joint model (JM) are two main open source implementations of H.264, and many researchers have evaluated their ideas using these frameworks; however, JM is slow for HD video and is mostly used in studies that focus on specific parts of the encoding process of H.264, such as the quantization, motion vectors, and the entropy. In contrast, x264 is faster than JM and is more suitable for real-time applications, such as HDVC, in which the end-to-end delay should ideally not exceed more than 100 milliseconds [55]. Hence, we have used x264 as the main framework for this thesis.

RC in x264 can be categorized into two main schemes: 2-pass and 1-pass. The 2-pass scheme is generally used for offline applications due to the extra latency introduced – x264 allocates a bitrate to each frame globally during the first pass while collecting certain statistics about the frames such as frame activities, the encoded frame size, etc. During the second pass, it precisely assigns a final bitrate based on the total video size and its complexity[56], where complexity is the predicted frame size for a frame at constant QP method.

In the 1-pass mode, the whole process is completed in one stage. This imposes some restrictions on the encoder. First, the information about the upcoming frames, such as frame activities and the possible frame size, is unavailable, and the encoder can therefore only use information about the current and the previous frames. Second, the encoding time is very limited and its speed should be fast. Consequently, the less complex methods are preferred. Even though the 2-pass scheme offers better quality than the 1-pass scheme for the same bitrate, the 1-pass RC is used in actual HDVC products due to its real-time capabilities.

The 1-pass scheme itself can be classified into three RC algorithms: the constant quantizer parameter (CQP), the average bitrate (ABR), and the constant rate factor (CRF). In[56], an additional method is introduced: constant bitrate (CBR); however, CBR is simply a combination of the previously

mentioned RCs.

Each method can be deployed with or without the use of a video buffer verifier (VBV). A VBV is the hypothetical queuing model of the decoder used on the encoder side. It ensures that the encoder generates the video bitrate according to the decoder capabilities. Otherwise, the output of the encoder may not be decodable at the decoder-end due to the decoder buffer overflow. In this chapter, each algorithm is briefly described.

2.1.1.1 Constant QP

CQP is the simplest 1-pass RC. It uses the quantization parameter (QP) for input. This value is then modified based on the frame type (I, P, or B) and it is constant for every frame type during a session. It is very simple and fast, but the final bitrate is not considered in the CQP algorithm. Thus, it is not suitable for HDVC because the bitrate will fluctuate significantly depending on the QP and the video content variation (complex and fast moving scenes producing very high bitrates).

2.1.1.2 Average bitrate

ABR attempts to maintain an average video bitrate equal to a desired bitrate. To maintain this bitrate, ABR calculates the value of the current frame's QP as follows [57]:

$$Q_{n+1} = \frac{\sum_{i=1}^{n} B_i Q_i}{\sum_{i=1}^{n} W_i} \quad (1)$$

where Q_i, B_i, and W_i denote the value of QP, the number of consumed bits, and the desired bits for frame i, respectively. The desired bits are the quota of each frame and are related to the bitrate; they are constant during the video session. Restarting the algorithm in the middle of the video session is the only way to change the bitrate.

In equation (1), the QP for the current frame is determined by QPs, consumed bits, and quota values of all the previous frames from the beginning. This has the following drawbacks:

- ABR depends on long-term history. Even the very first frame affects the quality. This is unsuitable for HDVC because the effect of the long past affects the overall quality.
- During low-activity frames, ABR increases the quality of the frames to maintain the video bitrate. Therefore, during low-activity frames, frame quality increases on a continuous basis

even though the user's perceived quality may not increase beyond a certain threshold. In addition, if the target bitrate is still not reached during low-activity frames, the bitrate budget underflows are preserved for the future. Once the low-activity period concludes and transitions into a high-activity period, a huge spike in bitrate will be produced. Even worse, the high bitrate will be maintained until all cumulative unused bitrates are consumed. Additionally, when going from low to high activity, ABR will increase the QP abruptly and will reduce the current bitrate significantly to keep the average bitrate constant. The longer the duration of the low-activity period, the more drastic the effects will be. As a result, a significant degradation in video quality is perceived by the user during transitions from low-activity to high-activity frames.

- Any unforeseen video activity, such as a scene change, is not tolerated smoothly by ABR, and the effect is perceivable in the subsequent frames. The reason is that the recovery time is, as equation (1) shows, rather long because any modification on the numerator is directly related to the QP.

The first problem can be solved by restricting the past history to only a specific time period, called a "window," and to apply equation (1) for the window only [57]. Although this improves the ABR performance, the other problems remain, which makes ABR unsuitable for HDVC.

2.1.1.3 Constant rate factor

CRF attempts to maintain the quality instead of the bitrate. The quality is adjusted by a value called the scaling factor, or the CRF value. This scaling factor is an integer value between 0 and 51. Setting the CRF value to 0 (near lossless) produces the highest quality and to 51 the lowest quality. Because CRF is a quality-based RC, an increase or decrease in the bitrate is directly related to the video activities. CRF instantly assigns higher bitrates to high-motion frames and lower bitrates to low-motion frames. It is also history-insensitive and does not consider previous frames. Thus, CRF fulfills features 2, 3, 4, and 5 from section 2.1 but it fails to satisfy feature 1 because it does not consider bitrate. Although increasing the quality factor leads to a bitrate decrease, the amount of reduction cannot be fine-tuned and varies for different frames. For example, Table 1 shows the relation between the CRF value and the generated bitrate for a video sequence. A snapshot of this sequence is presented in Figure 2. It can be observed that there is no way to reach the aforementioned target bitrate of exactly 3.5 Mbps by changing the CRF value. This makes CRF inappropriate for HDVC.

Figure 2: A snapshots of a video sequence.

Table 1: The relationship between the quality and the bitrate in CRF for the video in Figure 2.

CRF Value	CRF Bitrate without VBV (Kb/s)	CRF Bitrate with VBV (Kb/s)
19	4646	4660
20	3279	3187
21	2814	2782

2.2 CONGESTION PREDICTION

One of the main challenges of utilizing best-effort networks, such as the Internet, for video streaming is detecting network bandwidth modifications and adjusting video bitrate variations accordingly [58]. For further illustration, consider an HDVC system between two locations over a best-effort network as shown in Figure 3. This is a part of the whole solution presented in Figure 1. The video content is captured at the encoder end and sent through the network after packetizing the video data.

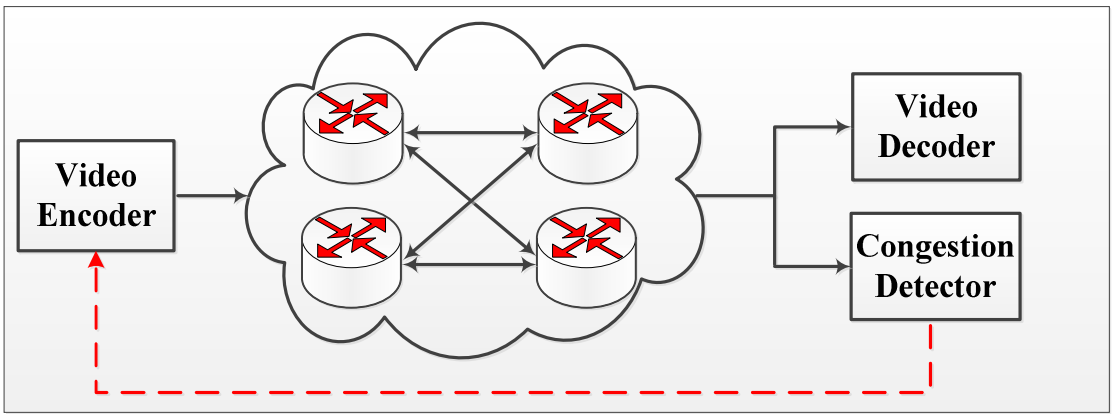


Figure 3: Block diagram of the HDVC system over best-effort networks between two locations.

As Figure 3 shows, the video decoder is responsible for reassembling the received video packets and displaying the video. The congestion detector, which works in conjunction with the video decoder, collects the packet statistics, such as inter-arrival delay, jitter, and packet loss. Based on the information that the congestion detection mechanism discovers, it can estimate whether or not there is congestion in the network. If the network is congested, then the packets are either dropped or delayed. In this case, information should be sent to the sender immediately so it can decrease the video bitrate to match the available bandwidth. In addition, if the bitrate of the video is lower than the available bandwidth, then the highest possible quality given the network capacity is not provided. For example, if the available bandwidth changes from 3 Mbps to 4 Mbps while the sender is still sending the video at the lower quality of 3 Mbps, then valuable resources are wasted. Therefore, in this case it would be necessary to increase the video bitrate to 4 Mbps to maximize the video quality.

The challenge of transmitting HD video over best-effort networks is attributable mainly to the fact that the available bandwidth in these networks is unstable over time and highly dynamic due to the competing cross traffic. Indeed, in best-effort networks, the bandwidth is not guaranteed, and that imposes additional requirements related to adjusting the video bitrate according to the available bandwidth at a given moment to maintain the QoS of the HDVC system. Real-time applications such as VC require a system that provides accurate and fast detection of bandwidth changes in order to adapt to those changes. The only way to provide sufficient time to make the necessary bitrate changes is to have a system that indicates bandwidth changes in the network as soon as they occur. The application then decides how to use those indications, which is the way it currently works for the live conversational applications. In the following section, we provide a more detailed description.

2.2.1 TCP FRIENDLINESS AND CURRENT RTCP-BASED APPROACHES

A goal for some congestion control algorithms is such that the video bitrate should be adjusted in a manner that makes its transmission TCP friendly [59]. The concern is to protect TCP flows from aggressive flows [60]. At the same time, since TCP is the most popular transport protocol, the changes in the video bitrate should be adapted so that the video flow itself is not suppressed by other TCP flows.

In [61], Floyd et al. proposed a model for adapting bitrate to network fluctuations for a unicast application by relying on the packet loss probability and the round trip time (RTT) measurements. This model has replaced the additive-increase/multiplicative-decrease (AIMD) approach in the regular TCP and has been shown to be TCP compatible. The throughput equation as provided in [62] is as follows:

$$X_{pps} = \frac{1}{RTT\left(\sqrt{\frac{2p}{3}} + 12\sqrt{\frac{3p}{8}}p(1+32p^2)\right)} \qquad (2)$$

where X_{pps} is the sending rate in packets per second, RTT is the round trip time in seconds, and p is the loss event rate between 0 and 1 of the number of loss events as a fraction of the number of transmitted packets. The sending rate in this scheme depends on the RTT and the packet loss event rate, and any increase in these two parameters will decrease the sending rate. The equation (2) not only performs congestion detection through monitoring RTT and p, but also provides a sending rate in response to the detected bandwidth fluctuations.

Other studies have expanded on this TCP-friendly scheme for various applications and network topologies. A comprehensive survey of these studies can be found in [60]. For example, the study in [63] extended the TCP-friendly scheme to multicast applications. In [63], a RTT-based prediction method was developed based on the packet loss probability and was shown to be TCP friendly. The network performance evaluation in [63] was based on comparing the true current state and the forecasted state at the end of the preceding control period.

2.2.2 TRAFFIC STOCHASTIC MEASUREMENTS

Another area of the related research concerns how to measure the statistical characteristics of the Internet traffic. It is well known that the data generated by Internet traffic shows long-range dependence and is multifractal, heavy tailed, and bursty [64][65] [66] [67]. Not only does empirical traffic data happen to be heavy tailed, but the central component and the tail of the data may illustrate different characteristics. In this regard, the work in [68] has modeled the traffic by considering the superposition of multiple flows with different characteristics. In this method, fractional gamma distribution was used to model the central portion of the traffic flow, which did not include heavy-tailed events (the tail part). To remedy this problem, a contaminated process has been implemented that modeled the outliers that cause the heavy-tailed phenomena, and the superposition of these two processes (one for the central portion and one for the outliers) was then used to model the traffic flow. We have avoided this approach and modeled the traffic parameters with the Pareto distribution, which includes the central portion as well as the outliers of the empirical distribution simultaneously. This provides us with enough mathematical tools to rigorously monitor the network conditions through a Bayesian analysis of the data.

The work in [69] assessed network failures by engaging confidence intervals for network surveillance. In this approach, binomial distribution has been used to measure the percentage of measurements that take place in a reasonable confidence interval. Our approach complements this method and could be combined with it to make congestion detections more accurate.

In addition to these models, packet loss measurements have been studied in [70], [71], [72], and [73] for services based on a virtual private network (VPN). In [70], a large-deviation technique has been used to estimate the packet loss of Gaussian traffic. In [71], a Kalman filter has been employed to measure packet loss. Also, in [72] and [73], control techniques are implemented to assess the packet loss using a Gaussian distribution for the network traffic.

One major difference between these approaches and our proposed technique is that we use a heavy-tailed Pareto distribution that, in contrast to light-tailed Gaussian, gamma, or binomial distributions, more realistically identifies the statistical properties of the Internet data.

2.2.3 AVAILABLE-BANDWIDTH MEASUREMENT

We present some of the existing methods capable of measuring the actual amount of available bandwidth (as opposed to detecting bandwidth changes). The objective is to show that no existing method can measure the amount of available bandwidth in real time and without significant overhead, which is necessary for our HDVC scenario.

In [74], the Delphi algorithm is introduced. This algorithm uses trains of exponentially spaced probing packets to estimate available bandwidth. Unique to this algorithm is a multifractal parametric model for the cross traffic that identifies its multiscale statistical properties. This method measures only stationary bandwidth and it is more suitable for a single hop. Also, it consumes considerable overhead.

The method proposed in [75], called TOPP, sends out several packet pairs that are well separated in time. It implements the linear regression and determines the available bandwidth when there is a delay. This method introduces the overhead of probe packets and can itself cause congestion in low bandwidth conditions.

SLoPS [76] manipulates the streaming rate of the trains of packets until it determines the available bandwidth, and it relies on the analysis of one-way delays of probing packets. If a train of k packets are sent, then the difference between the one-way delays of consecutive packets is monitored in order to measure the available bandwidth. If the differences are positive, the sending rate is above the available bandwidth, and if the differences are equal to zero, the available bandwidth is understood to be the same as the sending rate. This method is capable of measuring stationary and the fluctuating bandwidth, but like the method introduced in [75], it imposes significant overhead and can cause congestion itself.

The pathChirp method uses exponentially time-spaced packets to probe the available bandwidth and monitors the packet in the train at which the queuing delay begins to increase [77]. As a result, a point estimation of the available stationary bandwidth is provided, but at the cost of considerable overhead.

Pathload is based on the theory that one-way delays of a periodic packet stream will show an increasing trend if the stream rate is larger than the available bandwidth [78], which is basically the same theory used in the other methods. Although this method does not consume overhead, it suffers from a large time convergence.

Spruce has used the difference in time spacing between subsequent packets before and after the

bottleneck to weigh the capacity and estimate the available bandwidth [79]. This method is highly dependent on the precise scheduling of the probe traffic and only measures the stationary bandwidth.

In [80], the queuing delay at the bottleneck and the minimum one-way transit time is used to define a parameter, β, that estimates the proportion of total resources that were consumed by the bottleneck. Values of β close to 1 mean that the entire bottleneck bandwidth was available, and values close to zero indicate that the entire bottleneck is used by the competing cross traffic. The network assisted dynamic adaptation (NADA) [81][82][15] is the recent surge to determine the available bitrate. It uses three factors to determine the available bitrate: the one-way delay (OWD), packet loss and congestion marking. NADA determines the available bandwidth for any given time. However it has some constraints: first, the accuracy of the whole system is dependent on the accuracy of the clocks of two computers (client and server) and the frequency of RTCP packets sent back to the sender. Otherwise the clock skew leads to a negative impact on the system performance. Second, NADA imposes constant overhead to the system as some extra information should be transferred over RTP packets. In the [15], we have shown that DRC and NADA can collaborate with each other efficiently

Google congestion control (GCC) is another path [83]. GCC considers the packet loss on the sender side and the packet delay on the receiver side and tries to calculate the available bandwidth. It also defines the state machine to detect the overuse of the bandwidth and tries to adapt the bitrate accordingly. GCC calculates available bandwidth on both the sender and receiver and takes the minimum value of both whereas the method proposed here can avoid competing calculations by relying only on inter-arrival delay on the receiver side.

In summary, some existing methods can measure the amount of the available bandwidth, but they are not real-time methods and cannot be used in dynamic settings. Most of the approaches assume that available bandwidth is stationary ([75], [74], [77], and [79]) and consequently measure the available bandwidth over (possibly) a few RTTs. In the next chapter, we explain how our approach continuously monitors bandwidth fluctuations in real time, which makes it suitable for our HDVC scenario. The approaches that measure the bandwidth fluctuations, such as [78] and [76], introduce the overhead and the traffic congestion for the low bandwidth. In contrast, our method does not consume much overhead (by not introducing any probing packets), it is fast, and it does not cause any congestion (by not introducing any probing packets). Other limitations with the methods discussed relate to the clock resolution and synchronization between the sender and the receiver, the changes in bottlenecks, the multiple bottlenecks, and the existence of multichannel bottlenecks. These issues could lead to an underestimation or an overestimation of the available bandwidth. Therefore, our method uses statistical inference extensively to

minimize the effect of possible outliers. Furthermore, it avoids employing probing packets. Although probing packets could deliver accurate information about the stationary available bandwidth, there is no guarantee that in very short time periods, these packets travel the same route as video packets. Therefore, our method uses video packets at the receiver-end to infer bandwidth fluctuations. Because HDVC packets are usually sent according to RTP/UDP protocol, we will not have accurate information about the time spacing of packets at the sender; however, we have used Bayesian tools to calibrate the data.

CHAPTER 3

RATE CONTROL

In this chapter, we describe the new RC that is designed specifically for HDVC and that possesses the five features outlined in chapter 2. First, we discuss DCRF, which is an improvement of the CRF method. We then discuss DRC, which is suitable for HDVC.

3.1 DCRF

CRF fulfills the HDVC requirement mentioned in the introduction by decreasing the generated bitrate while there is little movement in the video and assigning more bitrate for high-motion frames or for sending I-frames instantly; however, it has significant weaknesses. Our DCRF method modifies the CRF method in a way that retains the desirable CRF features and addresses its weaknesses.

In essence, our RC method modifies CRF method by using a look-up table to predict the CRF value. This table stores the CRF encoding history. When a video frame is encoded, the CRF value and the generated bitrate are written in a table. Although the exact amount of the generated bitrate for each CRF value changes from one video sample to another, this process provides a rough approximation of the relationship between the CRF value and the generated bitrate, especially in the HDVC environment that consists people communicating each other most of the time.

DCRF uses a buffer called a local bandwidth calculator (LBC). The LBC calculates the recently generated bitrate using the encoder. The granularity can range from a few milliseconds to a few seconds. This feature is intended to improve the average bitrate concept to satisfy the bandwidth requirements. DCRF also takes the maximum and the minimum acceptable bandwidths as an input and applies them to the method. These values are set by the user.

The size of the buffer depends on the frame rate and the granularity value. The granularity value is configurable by the user to indicate the amount of the time needed for storing the recent frame information. The buffer size is therefore calculated using equation (3):

$$buffer\_size = frame\ rate \times local\ window\_size \quad (3)$$

The algorithm works as follows: DCRF uses the bitrate and the granularity value as an input. Then, it finds the closest possible value in the table and selects its corresponding quality. This quality value is an approximation at the beginning, and it is continuously updated thereafter. The buffer size is calculated according to the equation (3). DCRF saves the frame information in each encoding state. When the buffer becomes full, DCRF identifies the rate generated up to that point, increases or decreases the CRF value as soon as it determines that the bitrate is above the maximum threshold or below the minimum threshold, and applies the modification to the encoder. In addition, the part of the buffer that keeps the history of the previous frames is flushed. Then, the encoder collects new bitrate information from the arriving frames. The amount of the frame information (such as the frame size and the QP value) that is removed from the buffer is determined by the high-preemption and low-preemption thresholds when the local bitrate reaches the high and low boundaries, respectively. The flowchart in Figure 4 provides more details about the DCRF method. More details about DCRF can be found in [41].

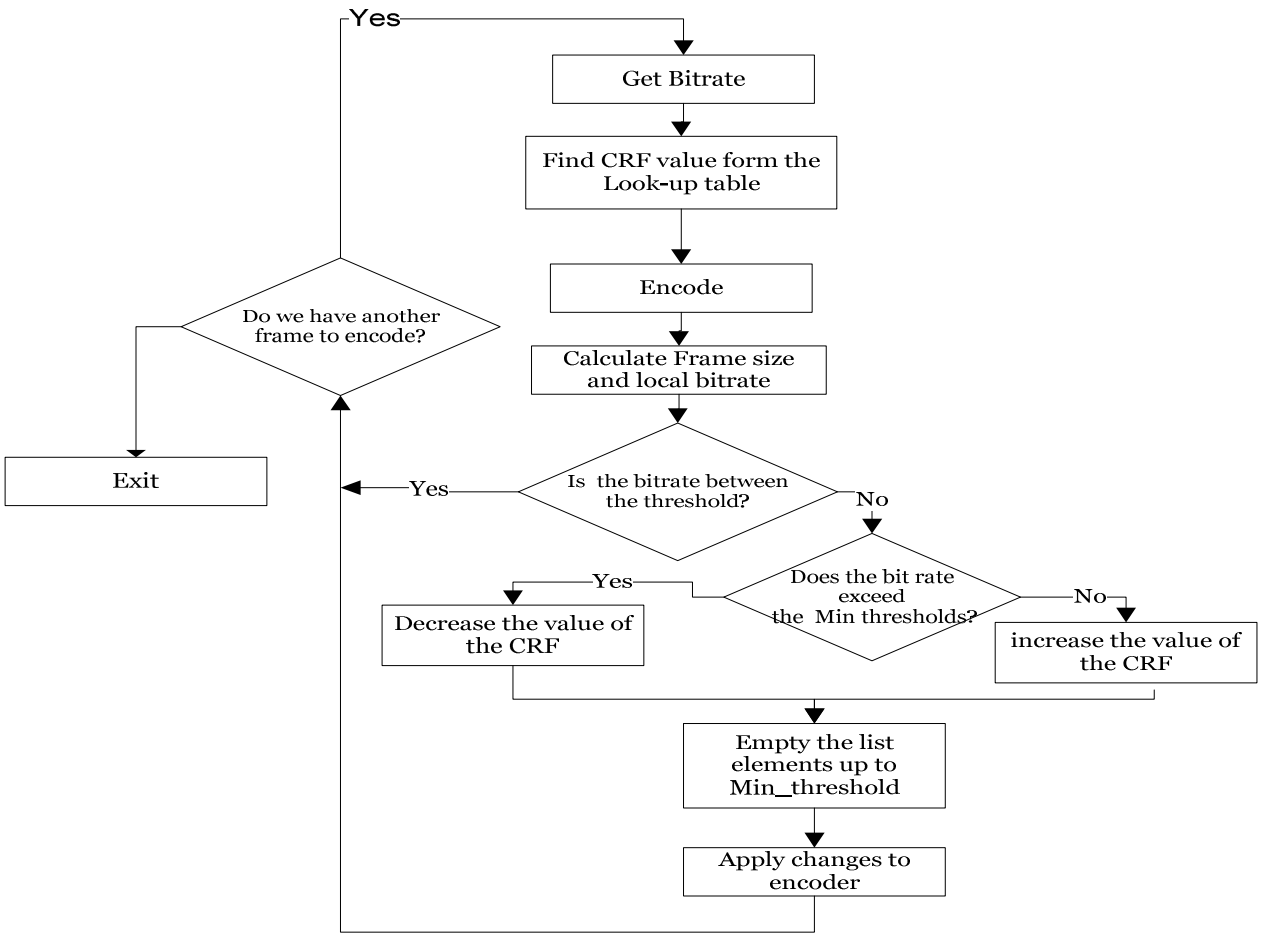


Figure 4: DCRF block diagram.

3.2 **D R C**

Instead of using the history of previous frames as in ABR or keeping the quality constant as in CRF or DCRF, we propose a new RC for HDVC called DRC, which makes significant contributions to HDVC. The overall algorithm for DRC is shown in Figure 5 and is discussed in detail in this section.

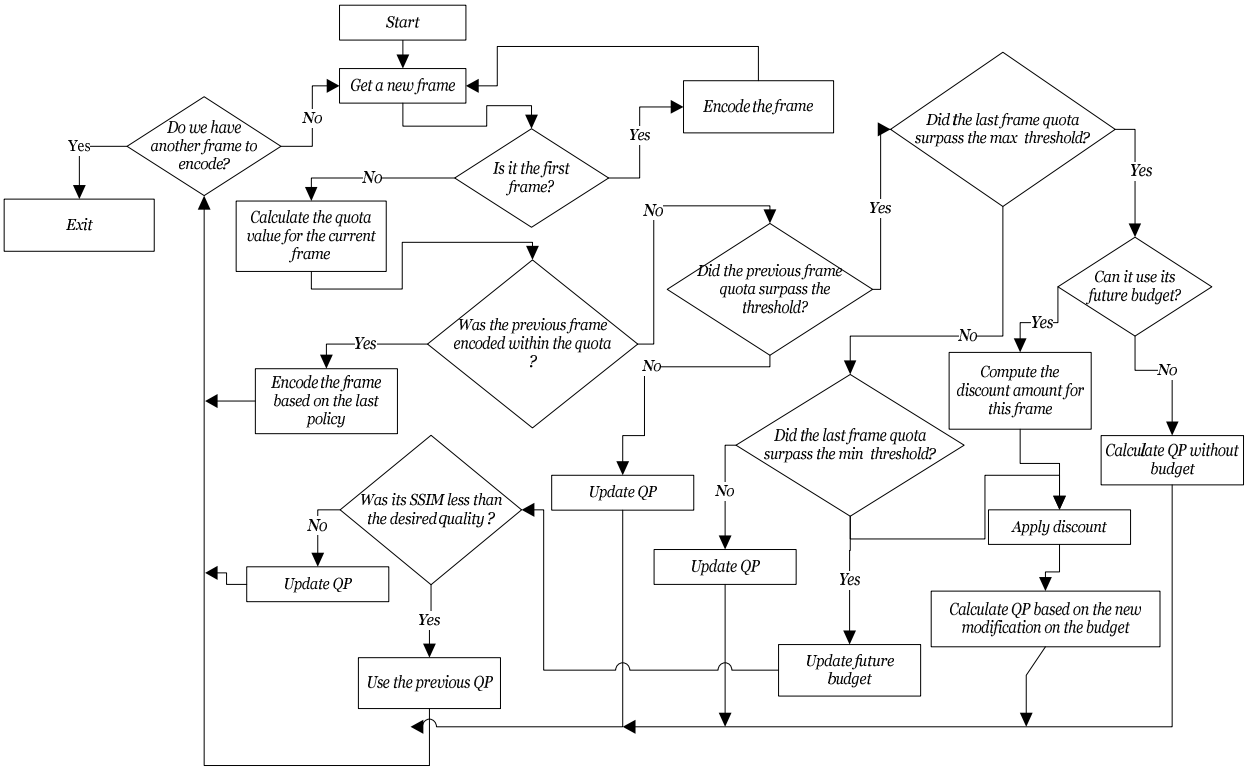


Figure 5: DRC algorithm in a steady state (i.e., available bandwidth is not changing).

First, the algorithm defines the concept of a "future budget," which is used in addition to the information of the previous frames' to allocate bits for the current frame. It also uses the window method introduced in [57] to limit the effect of the history of the previous frames on the current frame. The window size can be customized to fit the requirements of various applications. For example, the window size for HDVC should be kept low (30 seconds). The allocation algorithm for DRC can therefore be formulated as follows:

$$Q_{n+1} = \begin{cases} Q_n(\text{Future budget}), & \text{Future budget} \geq 0 \\ Q_n(\text{ABR}) \quad , & Future\ budget < 0 \end{cases} \tag{4}$$

As can be seen in equation (4), DRC behaves like ABR during high-motion frames. The Q_{n+1} (Future budget) calculation method is described in Figure 5. The update QP block is responsible for calculating QP based on the ABR method. The future information plays a key role in DRC. Consider the DRC algorithm in two situations:

- The bitrate is fixed, and the network delivers the determined amount of the bitrate with

some normal variations.

- The bitrate must be changed due to a change in the available bandwidth, so RC should reach the new bitrate as soon as possible.

In the first case, the most important issue is how to produce a bitrate compatible with the network behavior. Otherwise, the encoder will not transmit the video packet properly. From the encoder point of view, the network is characterized by the available bitrate, the burst size, and the burst duration. This does not mean that each frame should be encoded to a fixed size but that the encoder should use those network characteristics to offer enough flexibility for encoding the frames. For example, some frames could be encoded above the normal size due to the activity in the scene in order to maintain video quality while complying with the burst limit. To do so, DRC allocates an initial value to RC at the very beginning, called the future budget. The amount of the future budget is determined by the network characteristics. This value is also affected by other constraints, such as the frame rate, the minimum video quality, and the maximum video quality. Here, the minimum video quality determines the quality range suitable for HDVC users and is set by the user as an input. The concept of the budget is very similar to the monthly wage system for employees that determine their monthly expenditures. They can pay their monthly expenses and then deposit the rest of their salary in the bank for future "rainy days" when their necessary expenditures exceed their salary, or they can borrow from the bank if they do not have enough savings. Similarly, DRC uses the budget as the initial savings at the beginning of the video encoding process. DRC can then consume a percent of the savings calculated according to the available budget at a given moment when the bitrate usage is over the limit in order to prevent degrading the video quality. This budget allows the encoder to tolerate some fluctuations of the output, which are normal and acceptable ISP as bursts; however, if this over-the-limit consumption continues for too long, the budget will be consumed, and the video bitrate will be degraded to follow the target video bitrate. At this point, the behavior of the DRC becomes similar to that of ABR. However, in exchange for generating fewer bits than the target bitrate, DRC increases the video quality up to a maximum required quality (MRQ) for VC. If at this point the bitrate is still lower than the quota allocated for a frame, then the future budget is increased. The amount of the increase is equal to the difference between the quota and the size of the consumed frame. To avoid a huge burst in the future, the future budget is never increased beyond the initial budget. Therefore, in this respect DRC behaves differently than ABR, which increases video quality until it reaches the target bitrate, even beyond the MRQ.

In the second case, the DRC algorithm introduces a bitrate adjustment mechanism (BAM) to respond to the changes in the network bandwidth within 2 to 6 frames. Generally, RC would interrupt the

session and restart its algorithm with the new bitrate, which is similar to the ABR methods introduced by Semsarzadeh [57]. The interruptions and the sudden changes in the quality are perceivable by observers and are therefore undesirable. Ideally, the rate controller should be able to adjust the bitrate to the new target bitrate as soon as possible. This prompt change is applied when CRF changes the quality factor; however, CRF can produce only a desired quality, not a desired bitrate, as shown in Figure 4. Thus, this method is also unsuitable for HDVC.

Alternatively, the DRC algorithm interrupts its ordinary operations when it receives a new bitrate command and activates BAM. BAM performs the following operations:

1. It calculates a new QP value based on the new target bitrate.
2. It encodes the new frames according to the new QP value (at this point, BAM is in the "transient state").
3. It checks the current bitrate in comparison with the new target bitrate.
4. If they are in the same range (within a predefined threshold), BAM exits the "transient state" and enters the "steady state."
5. If they are not in the same range, BAM calculates a new QP value according to the equation (1). Afterward, it moves to step 3 for the next frame.

Because the algorithm does not have any previous history, the variations between steps can be high, allowing the algorithm to reach or get close to the new target bitrate quickly. Upon reaching a bitrate close to the new target bitrate, the last produced QP with the closest value to the new target bitrate is selected as the QP reference for the new target bitrate and is employed as the initial new state, called the "steady state". The "steady state" is very similar to the "encoder state" at the beginning where it sets a QP value for the specific bitrate. The details are shown in Figure 6.

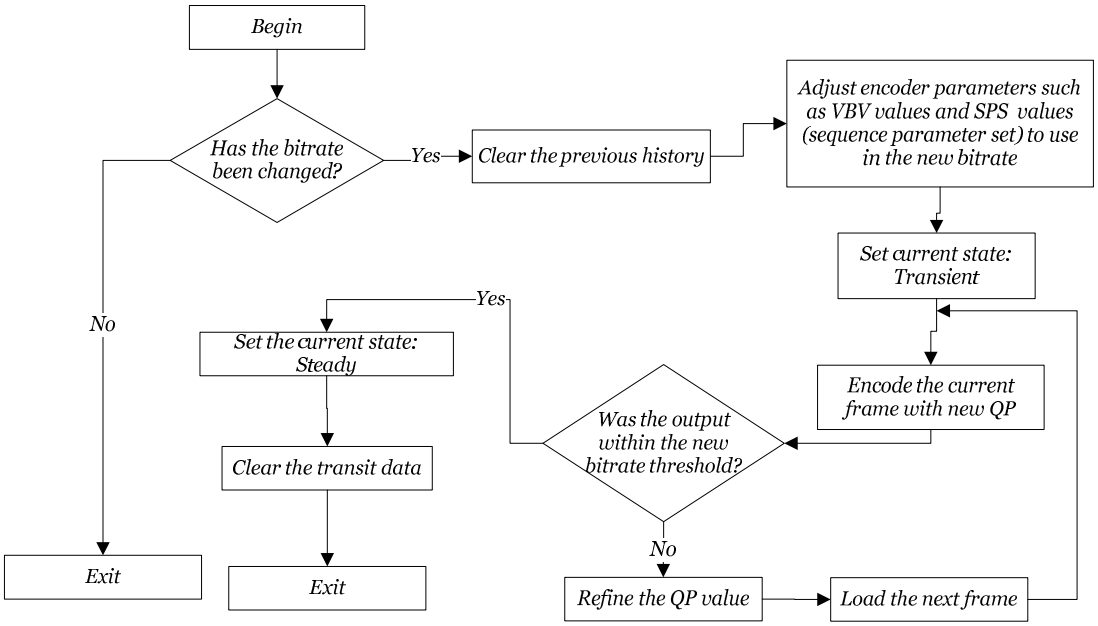


Figure 6: The bitrate adjustment mechanism (BAM) that becomes active when video bitrate must be adapted to match a change in the available bandwidth.

It should be noted that DRC works properly if the available bandwidth changes can be detected and reported to the encoder. In other words, a congestion monitor system must monitor the available bandwidth and report to the encoder that the video bitrate must be changed. For example, there may be a change from 4 Mbps to 3 Mbps. We assume that the bandwidth changes are identified and sent to the encoder using either traditional protocols, such as RTP/RTCP, or more advanced and faster methods, such as the one proposed in [84], or the one presented in the next chapter which have been designed specifically for HDVC.

CHAPTER 4

CONGESTION PREDICTION

In this chapter, we introduce our proposed weighted inter-arrival method and present our statistical model and Bayesian approach. The aim of this chapter is to describe the technical tools necessary for the bandwidth and the congestion detection mechanism.

We first consider only one queue of video packets that travel from a single sender to a single receiver through a best-effort network. At the receiver-end, the packets that have arrived are timestamped. We define the random process of inter-arrival times, D_i, as the difference in packet spacing at the receiver-end for a pair of subsequent packets. If R_i is the arrival time in the time-stamp units for packet i, then:

$$D_i = R_i - R_{i-1} \tag{5}$$

Note that:

$$\begin{aligned} D_i &= (OTT_i + T_i) - (OTT_{i-1} + T_{i-1}) \\ &= (OTT_i - OTT_{i-1}) + (T_i - T_{i-1}) \end{aligned} \tag{6}$$

where OTT_i and T_i are the one-way transmit time and the sending time of the i^{th} packet, respectively. Because the video packets are sent by RTP, the receiver has no information about the sending times. Obviously, out-of-order delivery is a problem here, but we use a statistical inference to minimize the impact of this issue.

The basic idea behind our method is that OTT shows an increasing trend when the available bandwidth drops. This situation arises when the queue size of the intermediate devices, such as routers, increases. Studies regarding to the OTT measurements can be found in [85] and [86]. Due to the variable packet sizes transmitted over the network, we use a weighted inter-arrival time, J_i, as defined in equation (7):

$$J_i = \frac{D_i}{\text{Size of packet i}} \tag{7}$$

As in existing studies on the Internet data analysis (e.g., [87] and [80]) and as confirmed by our measurements of HDVC packets, the random variable J_i is heavy tailed and approximately fits a Pareto distribution, as shown in Figure 7. Giorgi et al. [68] modeled network traffic by counting the number of data units (i.e., packets) that flow through the observation point during a given time slot of length T; therefore, the authors of that paper used a gamma distribution. Our research focuses on the weighted inter-arrival time of packets, and instead a Pareto distribution is adequate for modeling purposes. This is consistent with the fact that most Internet data is self-similar.

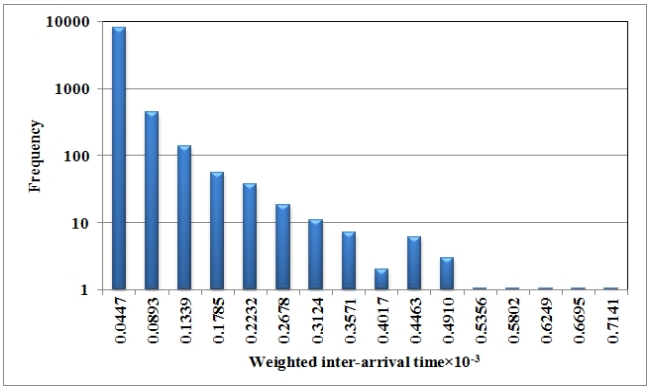


Figure 7: Logarithmic histogram of the weighted inter-arrival time for video packets of a sample video trace.

The probability density function (PDF) of a Pareto random variable with parameters ρ and x_M is as follows:

$$f(x \mid \rho, x_M) = \frac{\rho\, x_M^{\rho}}{x^{\rho+1}}, \quad x \geq x_M > 0, \rho > 0, \qquad (8)$$

where x_M is the minimum value of the random variable and ρ is the shape parameter. Similarly, for the random variable J_i, which corresponds to the video packets, we assume a Pareto distribution with a fixed or a random variable shape parameter ρ.

In the following sections, we illustrate the difference between assuming a fixed and a variable shape parameter with respect to the applicability of the measuring bandwidth variation.

4.1 SHAPE PARAMETER AS A FIXED VALUE

For our application, we have assumed that throughout an HDVC session, for all i's, the random variable J_i has a Pareto distribution with a fixed shape parameter. Based on this assumption, we have built our congestion prediction method using a fixed parameter, ρ. The characteristics of Pareto distribution regarding the fixed parameter ρ varies as follows [88]:

- $\rho > 1$: finite average.
- $\rho > 2$: finite variance.
- $\rho > 3$: finite skewness.
- $\rho > 4$: finite kurtosis.

Due to the nature of the Internet traffic, most Internet data with a Pareto distribution has the shape parameter $\rho < 2$, and, consequently, there is little chance for the existence of mean and variance [87] [89]. Because there is no guarantee for the existence of statistical central tendency (mean) and dispersion (variance, standard deviation) parameters, developing any method based on these parameters will lead to the unreliable solutions. In addition, Internet data is known to exhibit self-similarity, burstiness, and long-range dependence, and having a fixed shape parameter does not address these characteristics. Hence, we conclude that assuming a fixed value for a shape parameter will not clarify the behavior of bandwidth variations.

4.2 SHAPE PARAMETER AS A RANDOM VARIABLE

We have assumed that the shape parameter ρ is a random variable that varies over time. This is a reasonable assumption given the complex nature of Internet communications and data traffic as explained in [87] and [80]. This time-varying perspective harnesses the unpredictable behavior of the bandwidth variations. In other words, although J_i follows a Pareto distribution, there is no reason to accept a universal value for the shape parameter, because this parameter may change over time, even during a single HDVC session.

In order to show how a random variable ρ helps to detect bandwidth variations, we extracted the Pareto PDF for two different ρ values as shown in Figure 8.

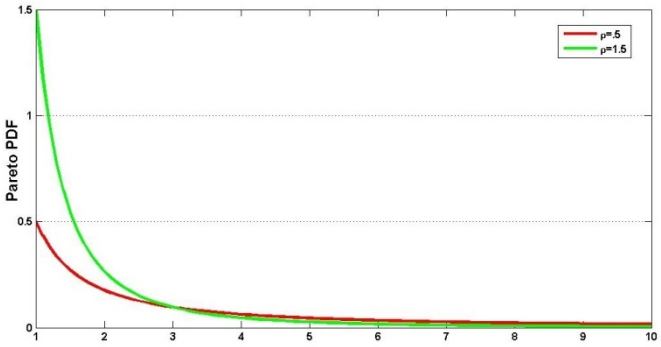


Figure 8: Probability density functions of Pareto distributions for ρ = 0.5 and ρ = 1.5.

As Figure 8 shows, in the case of $\rho = 1.5$, the distribution has a lighter tail, and the chance of encountering larger weighted inter-arrival times decreases. For $\rho = 0.5$, the distribution has a fatter tail, and there may be larger weighted inter-arrival times. In addition, in order to demonstrate the effect of x_M on density, Figure 9 shows how a change in the minimum value from $x_M = 0.001$ to $x_M = 0.002$ would alter the PDF when $\rho = 1.5$. Therefore, a stochastic shape parameter can be used to detect and manage the bandwidth variations. In this thesis, we have proposed the use of Bayesian statistics to model the bandwidth variations and detect the congestion. The details of the Bayesian method are presented in the next section.

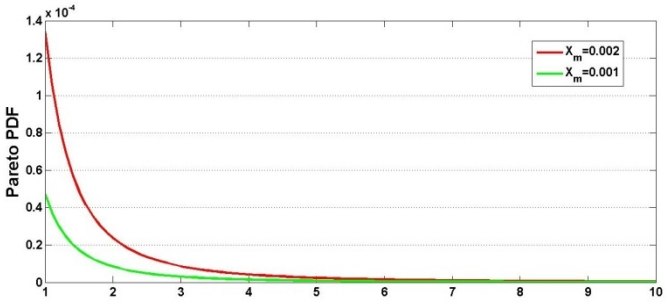


Figure 9: Probability density functions of Pareto distributions for $x_M = 0.001$ and $x_M = 0.002$.

4.3 THE PROPOSED BAYESIAN METHOD

In this section, we provide the details of the proposed Bayesian method. Specifically, this section presents how the weighted inter-arrival times can be used to update the distribution of ρ. In section 4.3.1, we illustrate the steps of our measurement scheme.

Assume that we have a reasonable estimate of the minimum parameter x_M. This information can be obtained from previous experiences, or it can simply be a sensible number (x_M= 0.000004, for instance), although it should not be much smaller than the actual minimum. From this point, the only unknown factor in the Pareto distribution is the shape parameter ρ, which is then considered to be the new random variable. As a random variable, the shape parameter needs a distribution by itself. In Bayesian statistics, the conjugate distribution for the shape parameter of a Pareto distribution with a known minimum, x_M, is the gamma distribution [90]. The choice of the gamma distribution guarantees that as we dynamically update the information according to the weighted inter-arrival times, the distribution of ρ will not deviate from the gamma distribution. We assume that $\rho \sim G(a, b)$, where $G(a, b)$ is the gamma distribution with a mean and variance of ab and ab^2, respectively [88].

Next, we show how the weighted inter-arrival times can be used to update the distribution of ρ. We then introduce the measurement scheme.

At the beginning of the session, we assume that the shape parameter denoted by ρ_0 follows the gamma distribution, $G(a_0, b_0)$, where a_0 and b_0 are predetermined and fixed. These fixed numbers can be chosen based on the prior knowledge, or we can choose some logical fixed positive numbers and anticipate that the system will correct itself in a short period of time. Note that initially, the mathematical expectation of ρ is $a_0 b_0$.

After the arrival of two packets, J_1 is calculated; packet numbering begins from zero. The likelihood function of $L(\rho|J_1)$, proportional to ρ, is shown in (9).

$$L(P|J_1) = \begin{cases} \frac{\rho {x_M}^{\rho}}{j^{\rho+1}} & \text{IF } J_1 > x_M \\ 0 & Otherwise \end{cases} \qquad (9)$$

Therefore, the posterior distribution of the shape parameter based on the observed data, J_1, is as follows:

$$G\left(a_0 + 1, \left[\frac{1}{b_0} + \ln\frac{J_1}{x_M}\right]^{-1}\right) \qquad (10)$$

As a result, based on the numerical value of J_1, ρ follows a gamma distribution with the average presented in (11).

$$E(\rho) = (a_0 + 1)[\frac{1}{b_0} + \ln\frac{J_1}{x_M}]^{-1} \quad (11)$$

Once the next packet arrives and J_2 is calculated, it will be used to obtain a new estimation of the Pareto shape parameter, and its corresponding average is shown in equations (12) and (13), respectively.

$$\rho \sim G\left(a_1 + 1, \left[\frac{1}{b_1} + \ln\frac{J_2}{x_M}\right]^{-1}\right) \quad (12)$$

$$E(\rho) = (a_1 + 1)[\frac{1}{b_1} + \ln\frac{J_2}{x_M}]^{-1} \quad (13)$$

This scheme is continued, and after each packet arrival, the expectation of the shape parameter is calculated, which provides a sequence of mathematical expectations of the shape parameters. Note that for simplicity, we can write the equations as follows:

$$b_n^{-1} = \frac{1}{b_0} + \ln\left(\frac{\prod_1^n J_i}{x_M^n}\right) \quad (14)$$

$$E_n = E(\rho) = (a_0 + n)b_n \quad (15)$$

As equations (14) and (15) illustrate, the effect of x_M is minimal algebraically, because x_M is involved in the calculations through the $\ln(.)$ function, which is a slowly varying function. In addition, our detection of bandwidth variations is based on sharp changes in the expected values of ρ. A small change in determining x_M would not have a significant impact on our detection, because all E_n's would be adjusted to the fixed x_M.

Moreover, note that if the bandwidth stays more or less constant, we expect the values of $\ln\left(\frac{\prod_1^n J_i}{x_M^n}\right)$ not to change drastically and the only change in E_n to occur through $a_0 + n$, which varies linearly. Therefore, if the bandwidth is fixed, we should encounter a linear E_n.

Our data shows that the sequence E_i is highly associated with the reliability of the network. In the next chapter, we show that bandwidth variations can be measured by utilizing the moving average of $E(\rho)$.

In order to express the relationship between E_n and the network status, we have extracted $E(\rho)$ based on the arrival time of video packets when the video bitrate varies between 0.75 Mpbs and 2 Mbps. The resulting $E(\rho)$ is shown in Figure 10. Although at the large scale the network condition follows $E(\rho)$ with an apparent linear trend, at the small scale, the nonlinearity of $E(\rho)$ makes it impossible to apply a

linear analysis. Because bandwidth variation has to be detected with millisecond-level accuracy, we have proposed to use the moving average of E(ρ) instead of its instant values or the average rate.

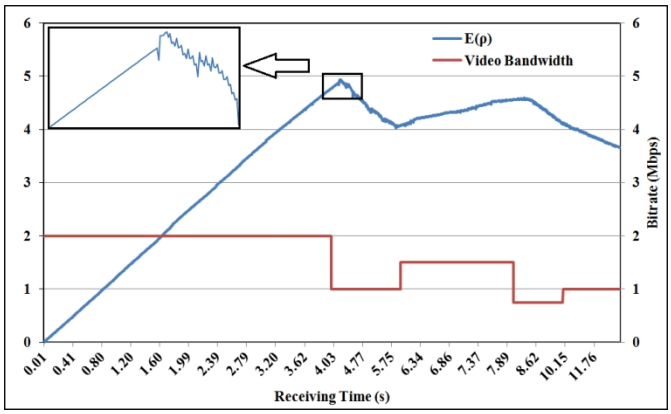


Figure 10: The reaction of E(ρ) to bitrate changes.

4.3.1 MEASUREMENT SCHEME

In a previous study [84], our analysis of actual HDVC traces have shown that variations of E(ρ) depend highly on the bandwidth fluctuations. Therefore, we propose a measurement scheme based on the comparing moving averages of E(ρ). The comparing moving averages is a well-known technique used for the prediction of stock markets and the financial applications [91].

We propose to generate a numerical sequence of differences of the last moving averages of orders n_1 and n_2 (where $rn_1 < n_2$) following each packet arrival. When the i^{th} packet arrives ($i > n_2$), the difference is determined as shown in equation (16).

$$Diff_i = MA_i(n_1) - MA_i(n_2) \tag{16}$$

where,

$$MA_i(n) = \frac{\sum_{i-n+1}^{i} E_k}{n} \quad , i \geq n \tag{17}$$

After determining the value of Diff, it will be simple to detect network congestion. The steps of List 1 demonstrate the overall scheme.

- ***Step 1 – Measure raw network information:***

  Determine arrival time, D_i, and size of packets upon arrival at the receiver's end, and extract J_i using (7).

- ***Step 2 – Update $E(\rho)$:***

  Use (15) to update the expected value of ρ.

  $$E_n = E(\rho) = (a_0 + n)b_n$$

- ***Step 3 – Evaluate Diff:***

  Evaluate $Diff$ using (16).

  $$Diff_i = MA_i(n_1) - MA_i(n_2)$$

- ***Step 4 – Generate lower and upper bounds:***

  Generate lower and upper bounds every fraction of a second.

- ***Step 5 – Monitor bound crossings:***

  Monitor lower and upper bound crossings, and detect bandwidth changes when a crossing occurs.

List 1: The proposed method for measuring bandwidth changes.

An acute decline in the value of the $Diff$ indicates a decrease in the available bandwidth; conversely, a rise of the $Diff$ value shows increase the available bandwidth. As shown in List 1, after extracting the Diff_i value, we can detect bandwidth changes by comparing Diff_i fluctuations with lower and upper bounds, represented by $\text{Diff}_{\text{i}-\Delta_2} - \Delta_1$ and $Diff_{\text{i}-\Delta_2} + \Delta_1$, respectively. Here, Δ_2 is the time interval in which the bounds are fixed. Once Diff_i crosses the lower (or upper) bound, it indicates a decrease (or an increase) in the bandwidth. After steps 1, 2, and 3, once $Diff$ is sampled, the lower and upper bounds will be generated according to the following rules.

- If the first nonzero decimal of $Diff$ is of order 10^{-2}, the bounds are proposed to be $\text{Diff} - 0.02$ and $\text{Diff} + 0.05$. We have used a closer lower bound to increase sensitivity with HDVC in mind.

- Otherwise, if the first nonzero decimal of $Diff$ is of order $10^{-\text{n}}$ where $\text{n} \geq 3$, the bounds are proposed to be $\text{Diff} \pm 1.5(10^{-\text{n}+1})$.

In order to avoid computation complications, we may choose a lower bound for $1.5(10^{(-n+1)})$. To avoid computational complexity, all calculations should be restarted periodically using the initial values. It should be noted that the numbers 0.02, 0.05, and 1.5 in the above rules have been determined experimentally using dozens of actual HDVC traces in which it was found that these numbers do not depend on the specific video but on the network behavior [84]. These numbers are quite important, and their choice affects the sensitivity of our congestion detection and the measurement method. It is also important to note that this method is a one-shut scheme and that it resumes its regular operation after the encoder responds to the notification.

CHAPTER 5

SIMULATION RESULTS

In this Chapter, we will present the result of our simulation. We define a testbed and test our method under different scenarios. We start with DRC and then we will evaluate the congestion prediction algorithm.

5.1 EXPERIMENT SETUP

We have implemented DRC as an RC in x264. We then have used 11 video sequences from actual HDVC sessions as well as traces from real the HDVC sessions provided by our industrial partner, Magor Telepresence, HDVC system. The snapshots of some of those videos are shown in Figure 11. We also used the SD video sequence "talking_head" from the SFU video test sequence, because it is similar to HDVC sessions (people talking in front of a fixed background) and is a standard reference.

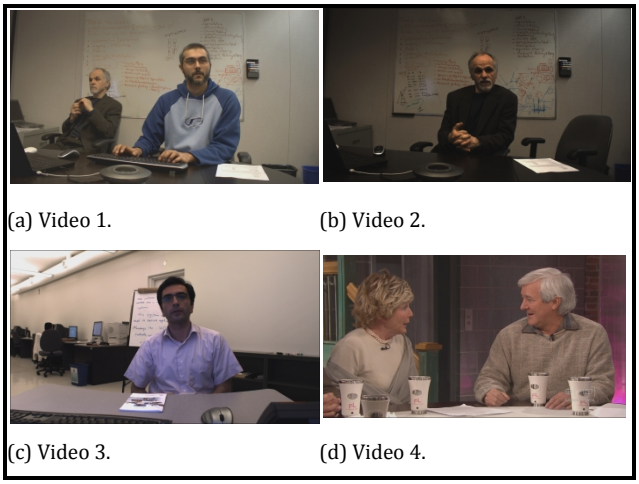


Figure 11: The snapshots of some video sequences.

Our test environment is shown in Figure 12. The encoder and the decoder are responsible for encoding and decoding the HDVC sequence, whereas a proxy server is in charge of simulating the network policies. The network bandwidth is determined at the proxy server using the traffic control (TC) commands. TC is a Linux program developed to manage network traffic. We have use the token bucket queue (TBQ)

algorithm to assign the bitrate to the encoder-decoder links. A live demo of this test environment has been presented at the ICME 2013 conference [16].

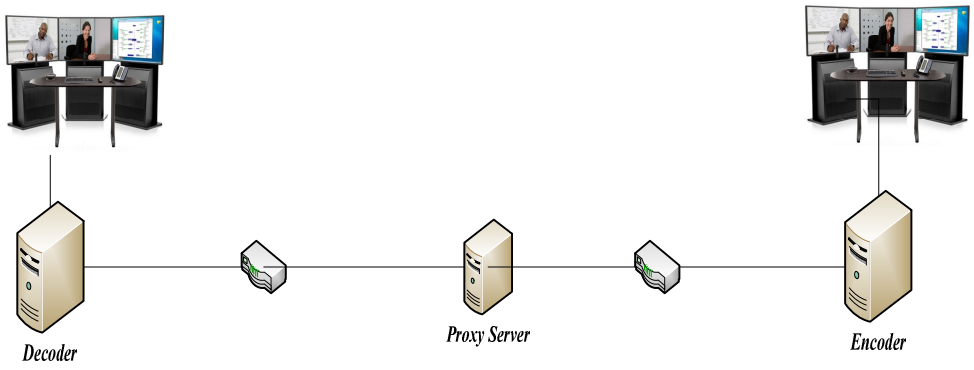


Figure 12: Simulation architecture.

The DRC results are compared to other RCs, which have already been implemented in x264, in terms of the bandwidth, the frame size, PSNR values for Y, and the structural similarities (SSIM). Each comparison has been performed for the different types of HDVC environments, from the low-activity sessions to the high-activity sessions. Even the noisy background has been evaluated.

In the following sections, the performance of three RC methods is evaluated in terms of the matched bitrate and the video quality. Simulation results are evaluated in-depth for a typical-activity video and subsequently for other more extreme video types. The typical-activity video is one in which a person is sitting in front of the camera, the background has minor modifications, and the person talks, moves her/his head, and raises her/his hands. Moreover, there is a whiteboard behind the person with some text written on it. The video is then encoded and transmitted to the decoder. The decoder displays it and stores decoded sequences in the separate directory. The original video sequence and the decoded video sequence are then compared. All simulations have been conducted for two primary situations: a fixed bitrate, in which the bitrate does not change during the session, which is presented in section 5.1.1, and a variable bitrate in which the bitrate changes from 2 Mbps to 1.5 Mbps, which is presented in section 5.1.2. The results are shown in Figures 13 to 16, and the results of the desired bitrate are targeted at 2.5 Mbps. Next, we have measured how different RC algorithms react, which is discussed in the following sections.

5.1.1 FIXED BANDWIDTH

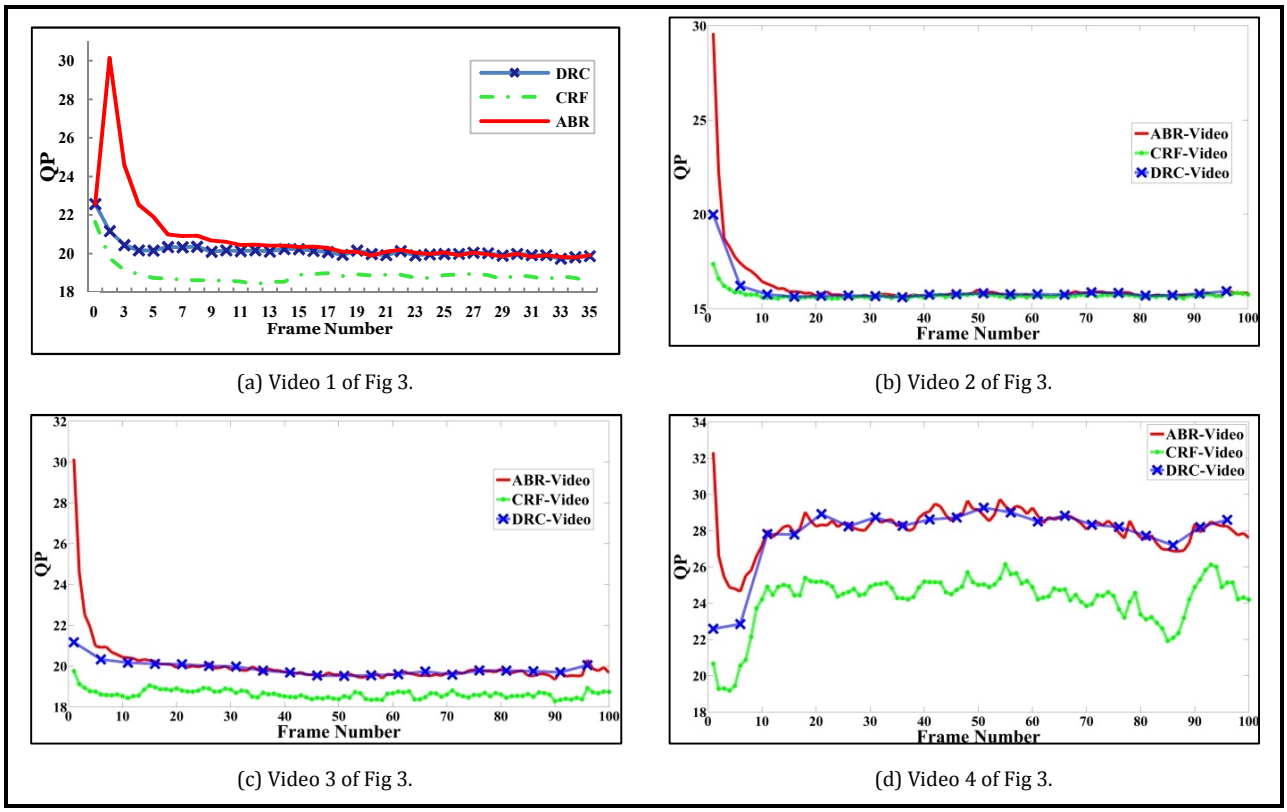
In this part of the simulation, the available network bandwidth is fixed. Our objective is to identify how the DRC and the other RCs utilize the current bandwidth and how RCs affect the following parameters: QP, frame size, SSIM, and PSNR. It should be noted that with a fixed bandwidth, we do not expect a significantly superior performance from DRC in comparison with other RCs, because the real strength of DRC becomes apparent when the available bandwidth dynamically changes (as in the real Internet). DRC reacts to those changes better than other RCs, which is demonstrated in section 5.1.2. We have chosen to include our results from the fixed bandwidth scenario in this chapter for two reasons. First, the results provide a baseline for comparison with the variable bandwidth scenarios. Second, the results show the performance of DRC against other RCs when the available bandwidth is fixed, especially in the HDVC-specific scenarios, as shown in section 5.1.1.5.

The results are shown in Figures 13 to 16, in which the desired bitrate has been set at 2.5 Mbps. We should note that because the first frame is encoded as an I-frame in exactly the same way for both ABR and DRC, the first frame has been shown only in part (a) of Figures 13 to 16 in order to increase the readability of figures.

5.1.1.1 Quantizer parameter

The QP values per frame for each method are illustrated in Figure 13: the higher the QP value, the higher the perceived degradation of the video quality.

DRC offers a consistent QP value throughout the video sequence and is very similar to CRF in the given bitrate, which is very good because it behaves similarly to CRF. However, CRF does not reach the desired video bitrate. ABR has huge QP oscillations at the beginning because it does not have any previous history. This conservative feature imposes degradation on the video quality at the beginning, although the quality is recovered after a few frames. This is important because it may create a problem whenever the encoder wants to jump to a new bitrate.



(a) Video 1 of Fig 3.

(b) Video 2 of Fig 3.

(c) Video 3 of Fig 3.

(d) Video 4 of Fig 3.

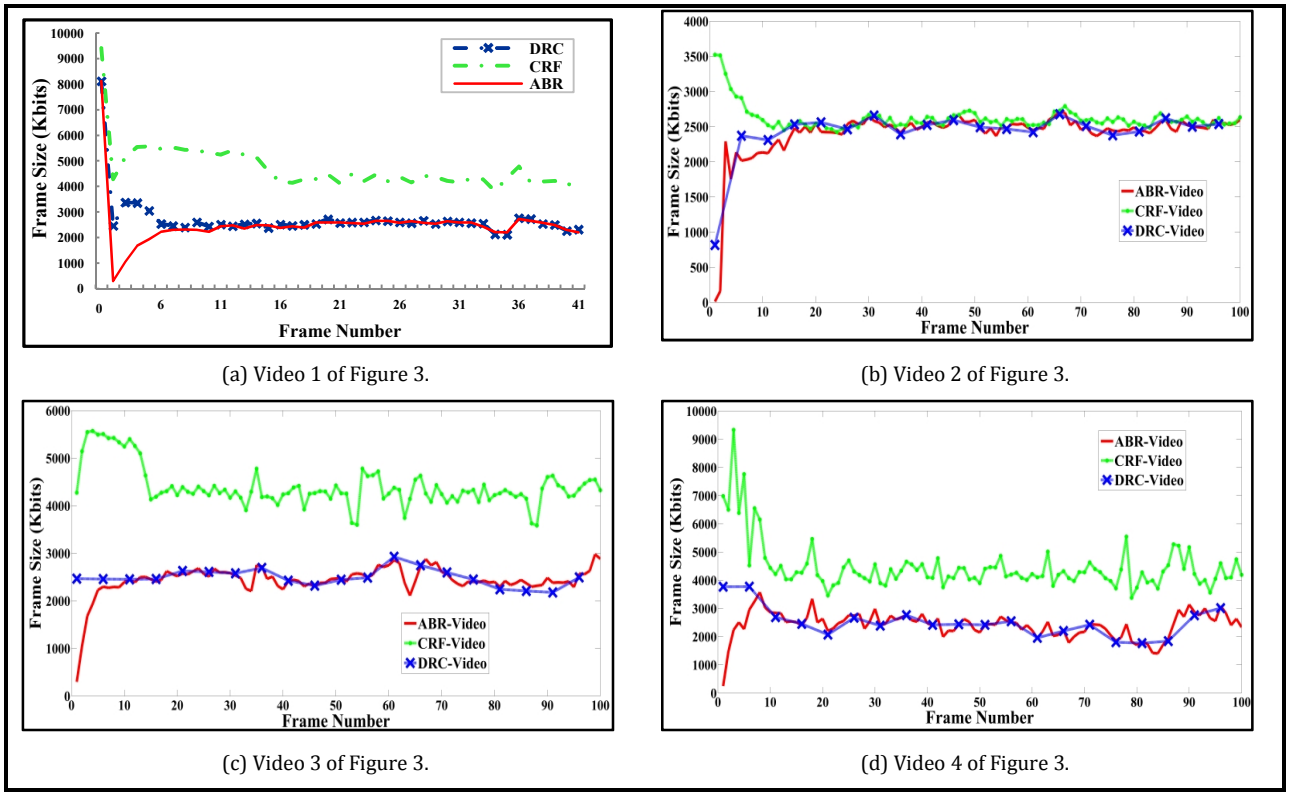
Figure 13: QP values.

5.1.1.2 Frame size

In order to compare DRC to the current RC methods, it is necessary to generate a similar bitrate for all of them. Therefore, the CRF value is set to 21 for the bitrate of 2.7 Mbps without using VBV. This is the best possible value to generate the closest bitrate to 2 Mbps. DRC and ABR bitrates are 2 Mbps. The first frame is encoded as an I-frame, and the following frames are encoded as P-frames, which is why the first frame has a bigger size in Figure 14(a); however, the sizes of the following frames vary according to the policy. In order to obtain a more accurate evaluation of the performance of RC, Figure 14(b, c, and d) displays the same result as Figure 14(a) but begins at frame 1 instead of frame 0. In real-world HDVC systems, sending I-frames is rare because a session does not usually contain many scene changes and the majority of frames are P-frames.

Figure 14 shows the frame size variations. This is a quantity that is directly related to the bitrate. Each frame is generated in a certain period of time, and a collection of frames generated in a time unit (second) defines the bitrate. According to the figures, DRC has a pattern similar to CRF. This means that DRC decreases the amount of the bitrate when there is little movement in the video and vice versa, which is

similar to CRF. ABR generates a small bitrate at first (because it does not have the previous knowledge about each frame) and then increases the bitrate to reach the average rate. This causes the video quality of ABR to be low at the beginning, which can be seen in Figure 14(b, c, and d), and violates part 5 of the HDVC features as described in section 2.1. On the other hand, CRF generates larger frame sizes because it cannot reach the desired bitrate. This means that packets will be dropped using the CRF method. Hence, DRC can generate the video according to the network bandwidth.



(a) Video 1 of Figure 3.

(b) Video 2 of Figure 3.

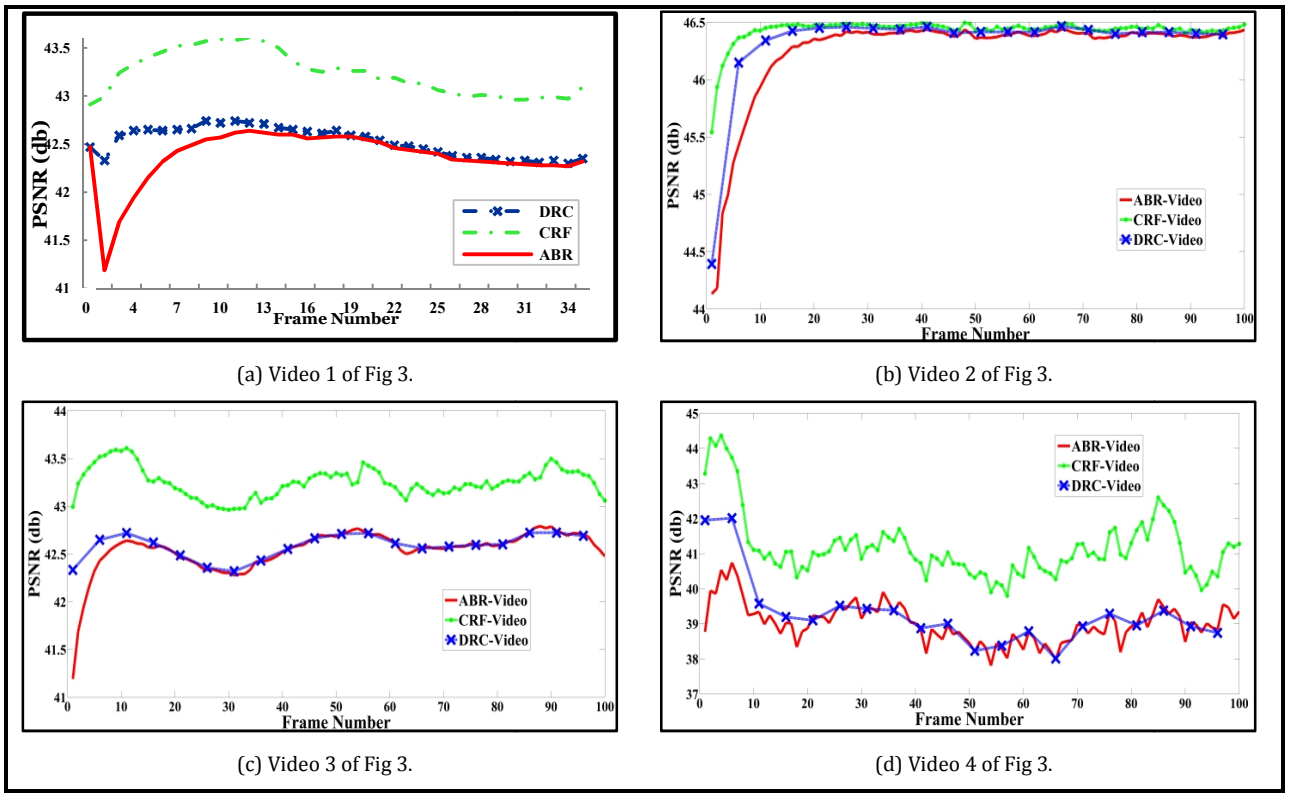
(c) Video 3 of Figure 3.

(d) Video 4 of Figure 3.

Figure 14: Frame sizes of different policies with I-frame (a) and without I-frame (b, c, and d).

5.1.1.3 PSNR

As can be seen in Figure 15, ABR has the lowest quality at first, though the quality improves as time passes. The quality of DRC cannot be compared with that of CRF because their bitrates are not the same (CRF does not adhere to the desired bitrate and therefore suffers losses through the network). But the quality of DRC is better than that of ABR. It should be considered that the DRC offers consistent quality throughout the video session.



(a) Video 1 of Fig 3.

(b) Video 2 of Fig 3.

(c) Video 3 of Fig 3.

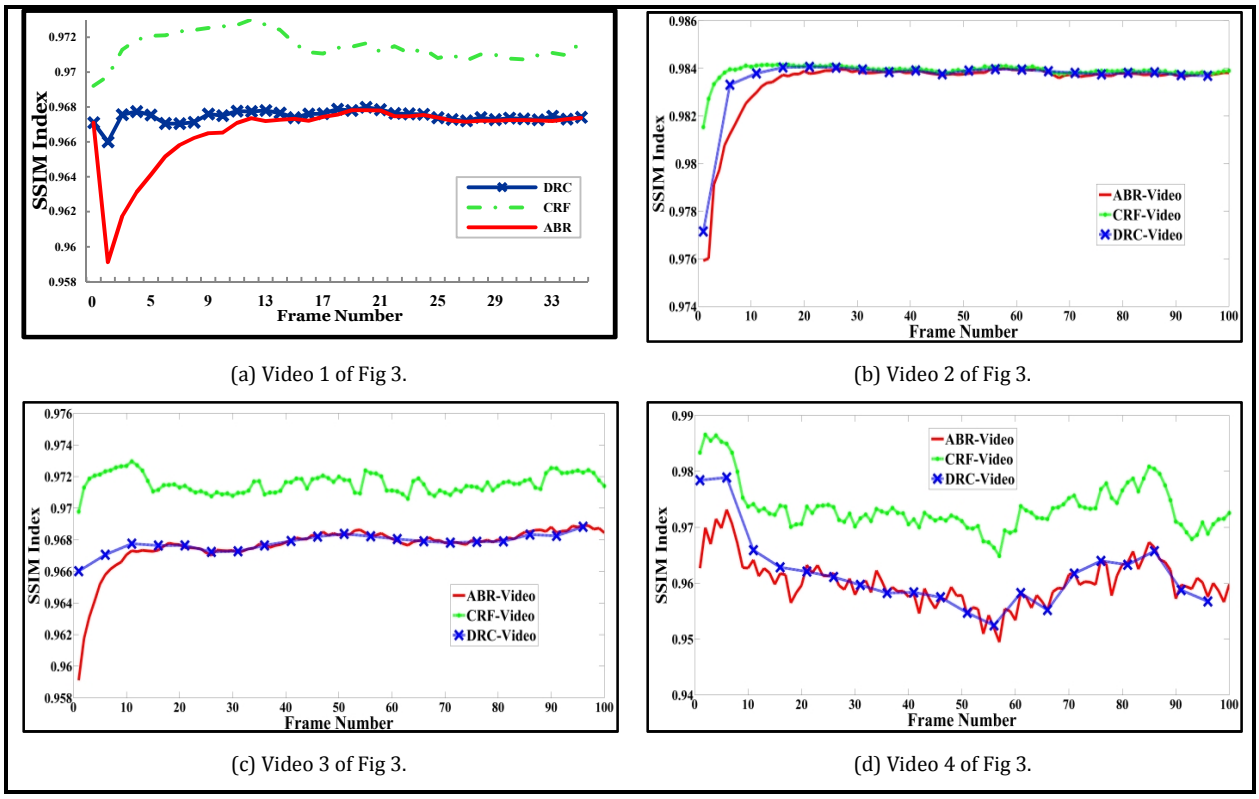
(d) Video 4 of Fig 3.

Figure 15: PSNR values (without I-frame).

5.1.1.4 SSIM

In this section, RCs are compared based on structural similarity (SSIM) [92] values. As can be seen in Figure 16, the quality generated by ABR is lowest at the beginning. As expected, CRF has the highest quality, but the quality of CRF is not "valid" because it generates more bitrate than the network could handle and would suffer from packet losses. Thus, DRC offers the best possible quality in the given bitrate.

DRC behaves similarly to CRF. This is very interesting considering that bitrate of CRF cannot be adjusted, whereas DRC can generate the desired bitrate.



(a) Video 1 of Fig 3.

(b) Video 2 of Fig 3.

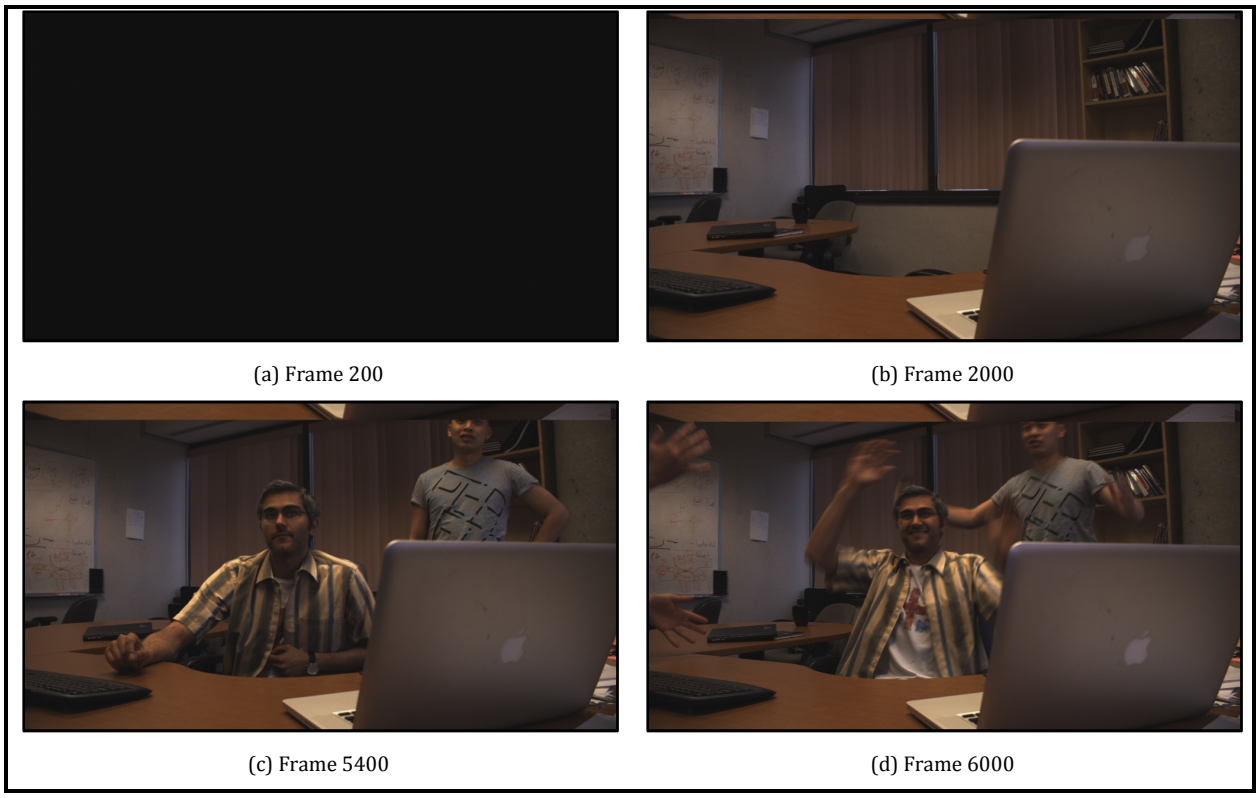
(c) Video 3 of Fig 3.

(d) Video 4 of Fig 3.

Figure 16: SSIM values (without I-frame).

5.1.1.5 HDVC-specific videos

In this section, a more complex scenario that usually occurs in HDVC sessions is investigated. A room is kept dark for a period of time (simulating an empty room); the light is then turned on, followed by low-activity motions and finally high-activity motions. The video is then transmitted over the network. Snapshot of this sequence is shown in Figure 17 and Table 2.



(a) Frame 200 (b) Frame 2000

(c) Frame 5400 (d) Frame 6000

Figure 17: Specific frames of real HDVC.

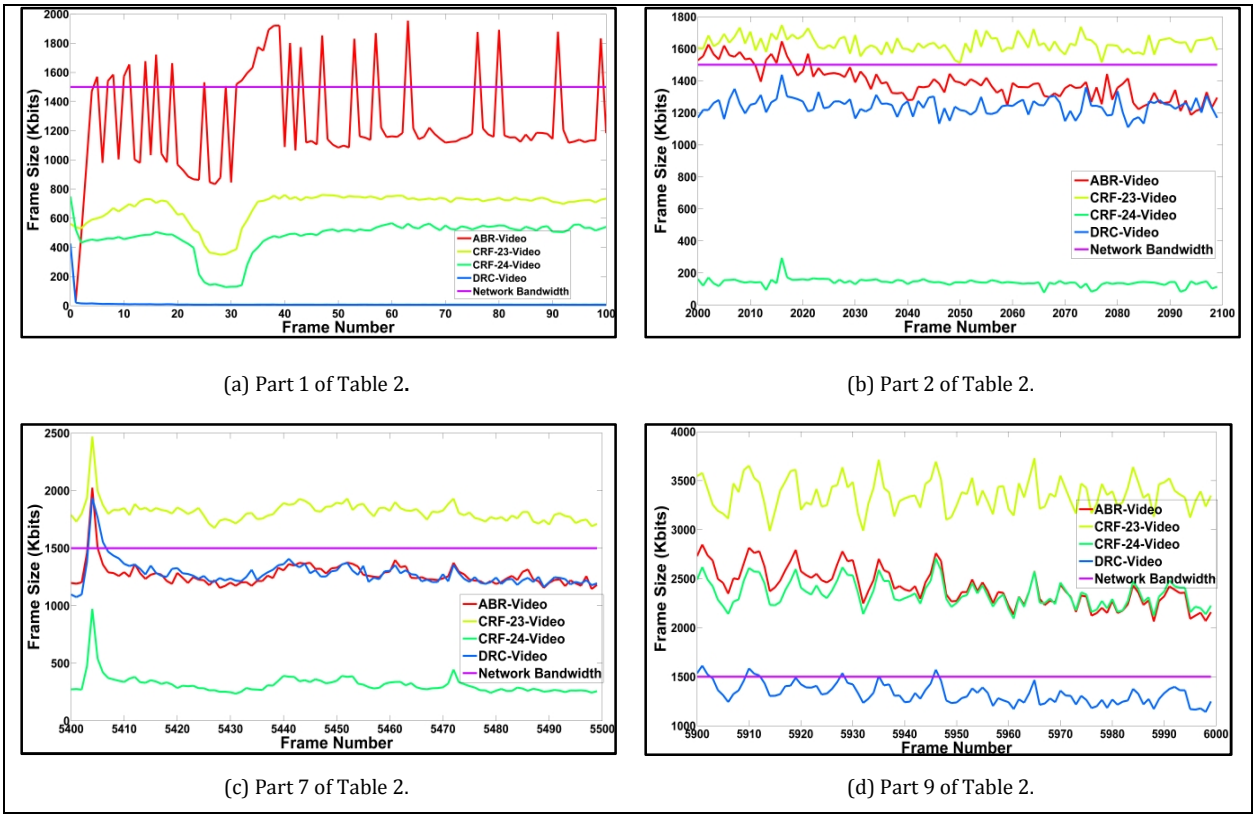
Table 2: Types of video activity descriptions.

Part	Frame number	Activity type
1	0–1889	Dark area
2	1890–3659	Lights on with no person in the room
3	3660–3834	A person is coming into the room (medium activity).
4	3835–3869	A person is sitting on the chair (medium activity).
5	3870–5214	A person (alone) starts talking (low activity).
6	5215–5359	Another person is coming into the room (medium activity).
7	5360–5713	Neither person moves, nor they start talking.
8	5714–5848	One of the persons moves his hand rapidly (high activity).
9	5849–6344	Both of them move their hands rapidly (highest activity).
10	6345–6473	Small movements (low activity)

To illustrate the performance of the CRF method more clearly, two CRF simulations with two consecutive QP values that are close to the target bitrate were conducted. In Figure 19, CRF-23 is CRF with the quality value of 23, and CRF-24 is CRF with the quality value of 24.

All frames have been analyzed, and the results of 100 frames of some parts (1, 2, 7, and 9) of Table 2 are illustrated here to show the efficiency of the proposed algorithm. For the first 1,889 frames, as shown in part 1 of Table 2, there is no light in the conference room. In this situation, CRF-23 and CRF-24 send a very low bitrate, as seen in Figure18 (a) and 19 (a), because there is no movement or change in the frames. However, the bitrate of ABR is substantial because it tries to maintain an average bitrate around 1.5 Mbps. In contrast, DRC transmits almost nothing. The main reason for this is that DRC adjusts itself based on the video quality. Since the quality is high enough, it decreases the video bitrate. The purple line in Figure 18 and Figure 19 represents the network bandwidth limit.

During the light-but-empty-room portion of the video, shown in part 2 of Table 2 and Figures 18(b) and 19(b), CRF-24 sends less video bitrate than the others but has very low video quality. CRF-23 has high quality but violates the network bandwidth, which will lead to lower quality due to the packet loss. DRC not only is within the limit of the network bandwidth but also produces higher quality video at that bitrate.



(a) Part 1 of Table 2.

(b) Part 2 of Table 2.

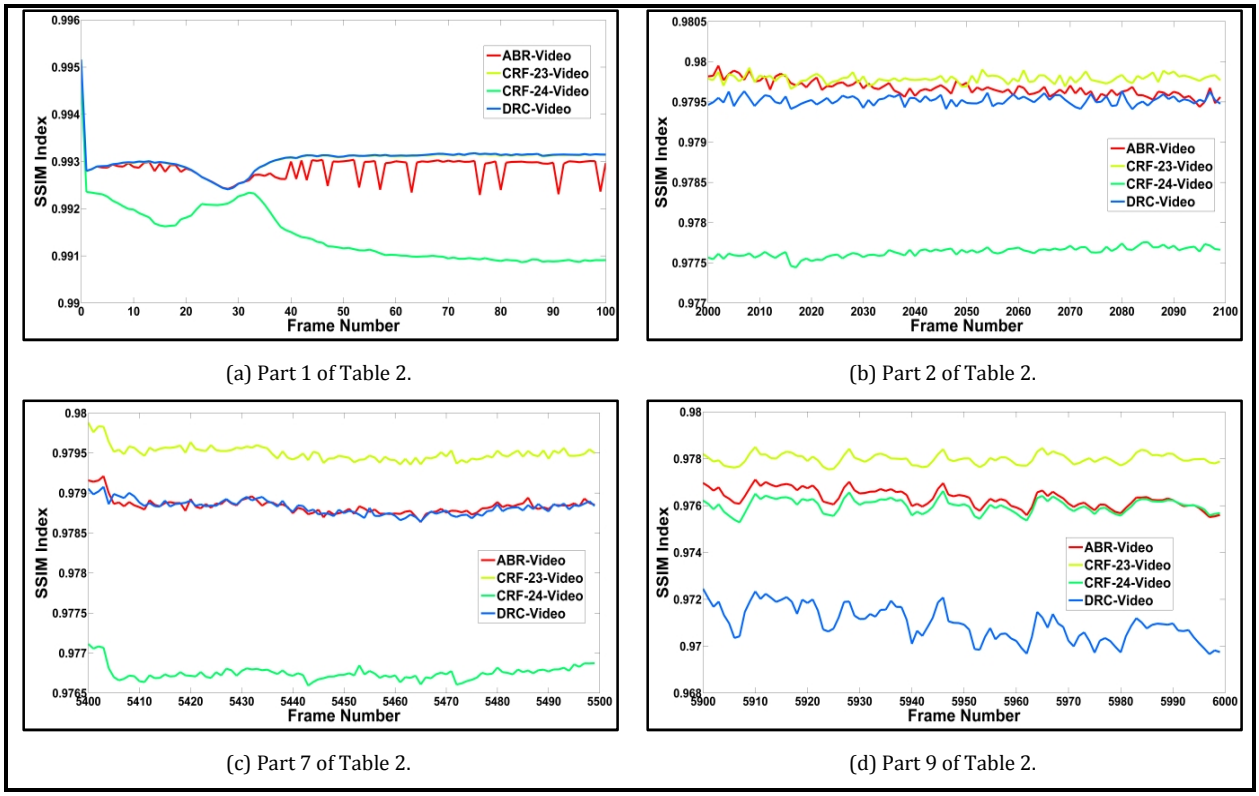
(c) Part 7 of Table 2.

(d) Part 9 of Table 2.

Figure 18: Frame sizes generated during different activities.

In the high-activity scenario, shown in part 7 of the Table 2, Figure 18(c) and Figure 19(c), DRC behaves similar to the ABR. This is promising because DRC maintains the video quality in the range of the available network bandwidth.

In the highest-activity scenario which is shown in part 9 of the Table 2, Figure 18(d) and Figure 19(d), DRC decreases video quality to satisfy bandwidth constraints. This is important in enabling the HDVC system to support high-motion activity.



(a) Part 1 of Table 2.

(b) Part 2 of Table 2.

(c) Part 7 of Table 2.

(d) Part 9 of Table 2.

Figure 19: SSIM produced in different activities.

5.1.2 DYNAMICALLY CHANGING BANDWIDTH

In the previous section, DRC has been compared with other RCs when the network bandwidth is fixed. DRC offers better performance in various conditions, but the true value of DRC is even more tangible when the available network bandwidth changes. The faster an RC adjusts to a new bitrate, the less likely it is for it to produce a negative outcome in the video session. In our first set of simulations, the available network bandwidth (and hence the desired video bitrate) has been changed from 2.5 Mbps to 2 Mbps at frame 101. The frame rate in this simulation is 10 frames per second (FPS). In the following sections, DRC has been evaluated under different measurements in terms of the network fluctuations.

5.1.2.1 QP

The rate modification occurs at frame 101. As can be seen in Figure 20, ABR has a large change in QP value. Therefore, there is a huge impact on the video quality, and it takes about 13 frames to reach to new bitrate. CRF has only a few changes because it cannot find a suitable QP value for the new bitrate and fails to change the video bitrate, whereas the DRC has the fastest adjustment and reaches the new bitrate at frame 105 (in 4 frames).

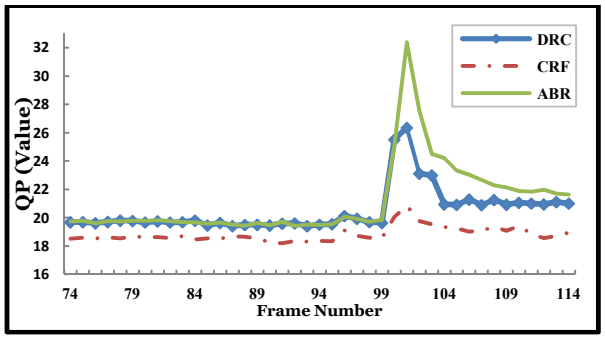


Figure 20: QP values when the bitrate has been changed.

5.1.2.2 Frame size

Figure 21 shows that ABR generates a large frame size at frame 101 because it closes the encoder and reopens it after the reconfiguration and flushing its history buffers (requiring an I-frame). CRF exhibits the same behavior, but it never reaches the new bitrate at all. DRC reaches the new bitrate within a few frames, which is an important characteristic of DRC.

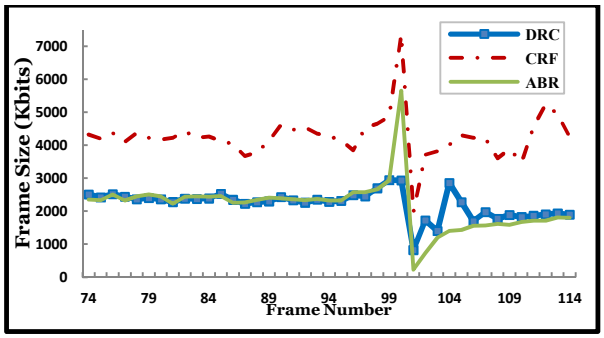


Figure 21: Frame size values when bitrate has been changed.

5.1.2.3 PSNR

PSNR values are shown in Figure 22. As expected, ABR initially has a large negative impact on the quality of the frames (especially in frame 101), and after some time the video quality increases. CRF has some fluctuations around the frame 101 as well; however, the result cannot be compared with that of the other RCs because CRF fails to adjust the desired bitrate. Again, DRC is able to recover the video quality and reach the final bitrate much faster than ABR.

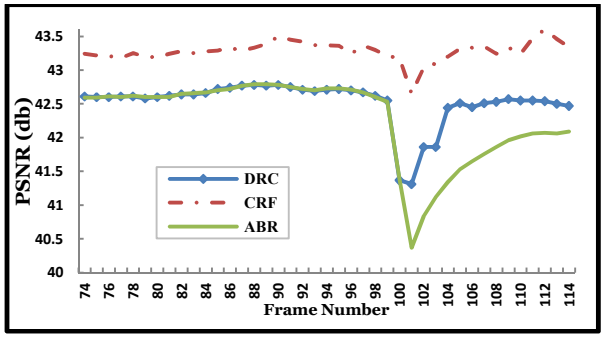


Figure 22: PSNR value when the bitrate has been changed.

5.1.2.4 SSIM

The SSIM variations are similar to the PSNR variations. As can be seen in Figure 23, the quality

of ABR reaches the new bitrate only after a long period of time, whereas DRC reaches the new bitrate only after 4 frames. Moreover, CRF fails to reach the new bitrate.

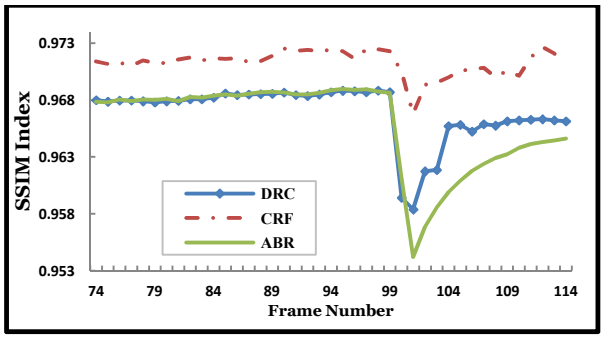


Figure 23: SSIM value when the bitrate has been changed.

5.1.3 CONSECUTIVELY BANDWIDTH CHANGES

For the second simulation, we first demonstrate two consecutive bitrate modifications at frames 100 and 200. The video has been initially encoded at 1 Mbps. The video bitrate has then been increased to 1.5 Mbps at frame 100. The video bitrate has been increased again at frame 200 to 2.5 Mbps. This simulation is conducted with DRC and other competitors (ABR and CRF). Figure 24 shows the PSNR values for different methods.

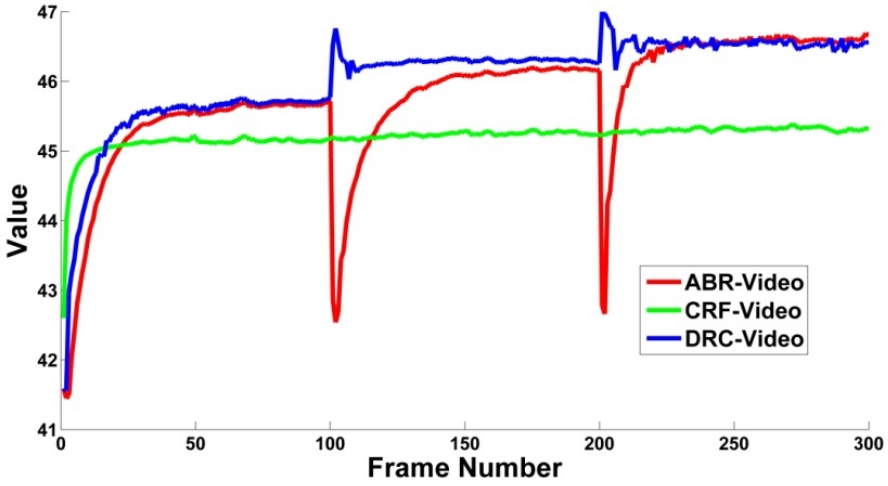


Figure 24: PSNR value when the bitrate changes multiple times.

Next, we have repeated the first simulation with two consecutive decreasing bitrate modifications at frames 100 and 200. The video is initially encoded at 1.5 Mbps, decreased to 1 Mbps at frame 100, and decreased again to 700 Kbps at frame 200. This simulation is conducted with DRC and other competitors (ABR and CRF). Figure 25 shows the PSNR values for different methods.

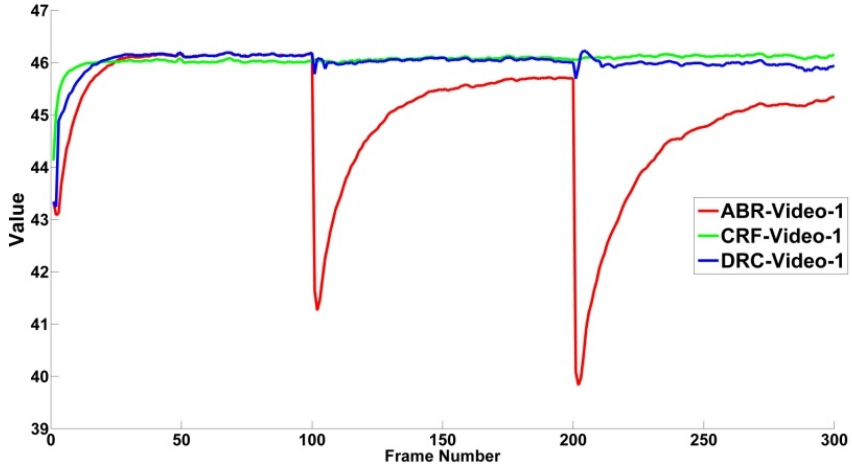


Figure 25: PSNR value when the bitrate changes multiple times.

As can be seen in Figure 25, DRC offers consistent and better quality than ABR and CRF. CRF is unable to adapt to the new bitrate, and ABR imposes huge video quality fluctuations.

5.1.4 SUBJECTIVE TESTS

The encoded frames have been subjectively evaluated by employees of Magor Corp. and students of the DISCOVER lab at the University of Ottawa. Because of market interests, Magor Corp. has performed an internal extensive subjective test to analyze DRC. Figure 26 is a snapshot of a specific frame from the "Magor_moj" sequence.

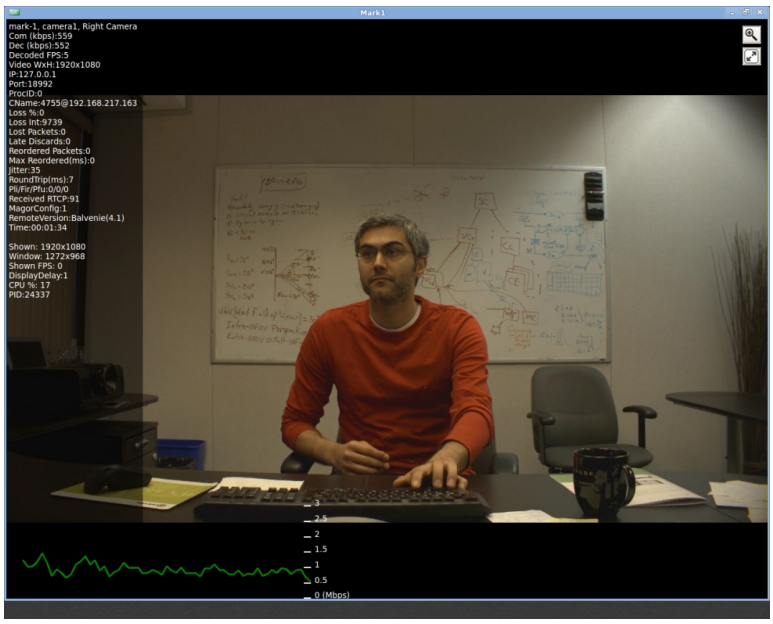


Figure 26: Subjective test result for the specific frame at "moj_sequence".

The subjective tests have involved viewing the same decoded image from different rate controllers on a 47-inch 1080p monitor and subjectively comparing them by zooming in on particular sections of a frame and switching between frames to compare the quality of the frames. This job is done

by Magor video experts as well as lab students who are working on the multimedia related areas. We have asked them to compare each frame quality and tell which frame has the better quality. We have asked 17 students to participate in the experiment and their ages are between 22 and 30. The orders of RCs have been randomly selected (in some cases they were shown DRC first and in some cases DRC last). Their opinions have been collected and rounded to the near integer value and depicted in Figure 28. The text on the whiteboard is shown in Figure 27. Figure 28 shows the observers' opinions for different frames. The subjects have ranked the quality of frames by giving scores between 5 and 0, in which 5 and 0 represent the best and the worst qualities, respectively. As seen in Figure 27 and Figure 28, DRC offers better quality most of the time. The person in the picture starts moving at frame 350, so DRC offers higher quality. The same response occurs for frame 500, in which there is typical activity. High-motion activity starts at frame 560 and ends at frame 615. In this case, CRF has higher video quality because DRC reduces the quality to maintain the bitrate regarding to the long-term considerations; however, DRC still offers a higher quality than ABR. At the last frame, there is no activity, and DRC is still superior.

ABR DRC CRF

Figure 27: Zoomed version of the specific frame.

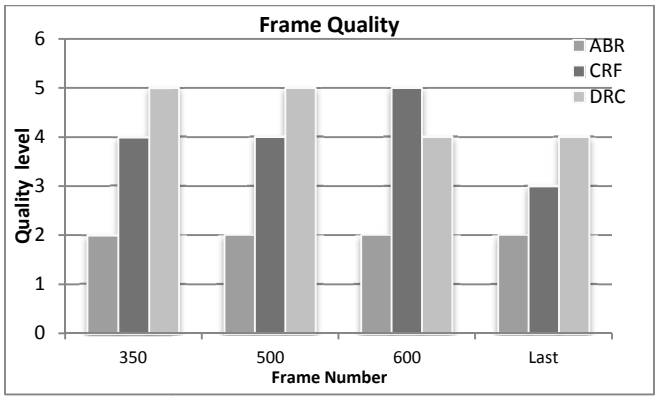


Figure 28: The overall subjective test results.

5.2 CONGESTION PREDICTION

5.2.1 EXPERIMENT SETUP

In this section, we present our comprehensive experimental simulations (NS2) and real-trace implementation and evaluation. We show that the speed and accuracy of our method is superior to most commonly used methods, such as the rate adjustment algorithm of RTT-based schemes [62] [93]. To examine the performance of the proposed method, we used the x264 open source reference software developed in [29] as the H.264/AVC video encoder and the NS2 software developed in [94] as the network simulator platform. In addition to the simulations, the proposed method has been implemented in the commercial HDVC solution provided by our industry partner Magor Corp., and the real-world test scenarios have been performed over the Internet to demonstrate the robustness of the proposed scheme.

We should mention that we have not done any simulation related to TCP-friendly [60] feature for our method. This is because our main objective is to predict network congestion based on the incoming packet information as soon as possible and have relaxed the TCP-friendliness requirement in favor of faster reaction to congestion.

5.2.2 NS2 SIMULATIONS

The simulation setup is depicted in Figure 29. S1 sends the video stream over RTP, and the video packets are streamed through the routers R1 and R2. The whole throughput of the system is determined by the link capacity L1. S4 acts as a receiver, whereas S2 and S3 are used to generate additional traffic to represent the typical traffic activity. S1 sends video packets, and S4 collects the video packets and stores them for the analysis. All the network parameters are stored in log files.

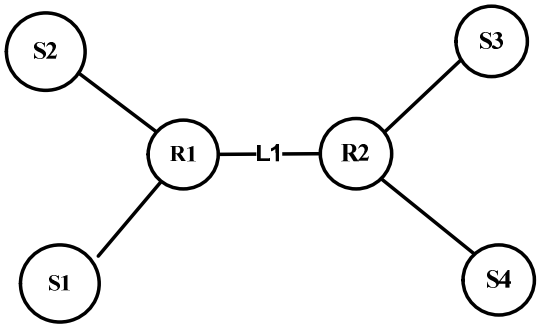


Figure 29: The simulation setup.

In our simulation scenario, various videos are coded and sent through the NS2 software. For this experiment, we consider $n_1 = 20$ and $n_2 = 50$. We use a dynamic Δ_2 and update the bounds every 50 packets arrived based on the update rules discussed in the previous chapter. In order to examine the performance of our method under different network conditions, we have considered six cases for HD videos and two cases for SD videos as shown in Table 3.

Table 3: The network condition cases.

Case Number	Network Bandwidth (Mbps)			
	HD Videos		SD Videos	
	Initial	Secondary	Initial	Secondary
1	2.0	3.0	2.0	1.0
2	2.0	2.5	2.0	1.5
3	2.0	2.25		
4	2.0	1.75		
5	2.0	1.5		
6	2.0	1.0		

For all the cases in Table 3, the video encoder bitrate is fixed and is equal to the initial bandwidth of each case. The network then changes its bandwidth at frame 60 (i.e., at second 2 of the video) to the secondary bandwidth indicated in Table 3. Note that at second 2, the bandwidth reduction process is initiated and the network therefore takes a while to reach the secondary target bandwidth. The video encoder bitrate is not adjusted after the bandwidth manipulation.

Table 4 and Table 5 show the response times of the tested methods for various HD and SD video sequences and network conditions, respectively. It should be mentioned that the recommended time spacing between RTCP packets is 5 seconds [93] [95]; however, because one of our goals is to illustrate the speed of our approach, we have decreased the RTCP transmission intervals to 1 second (in Table 4) and 0.1 seconds (in Table 5) to give our competitor an edge. It should be noted that when we refer to TCP-friendly methods in this chapter, we mean the conventional schemes that are based on the TCP-friendly approaches described in section 2.2.1. Recall that these approaches typically use RTCP to exchange the network statistics between the sender and the receiver. The term "no detection" represents the cases in which no bandwidth change is detected during the simulation period, which is 5.5 seconds. As the results illustrated in Table 4 and Table 5 show that our proposed method outperforms RTCP-based methods in 95.8% of the simulation cases. When both approaches have successful detection, our approach detects bandwidth changes 511.5 milliseconds sooner on average with a standard deviation of 322.95 milliseconds. In addition, the competing approach behaves poorly when the bandwidth increases. In case 6 (bandwidth decreasing from 2 Mbps to 1 Mbps for HD videos), our approach is 320.6 milliseconds

faster on average, and the standard deviation of the differences is 125 milliseconds. Moreover, in the first four cases, the competing approach has no detection during the simulation period.

Table 4: The response time comparison for proposed and the TCP-friendly schemes with one second RTCP interval – HD videos.

Case #	Sequence Name	Response Time (Sec)		Difference
		Proposed Method	[38][41]	
1	Tractor	0.0990	No detection	∞
	Sunflower	0.1224	No detection	∞
	Station 2	0.1201	No detection	∞
	Rush hour	0.1382	No detection	∞
2	Tractor	0.1216	No detection	∞
	Sunflower	0.1616	No detection	∞
	Station 2	0.1438	No detection	∞
	Rush hour	0.1647	No detection	∞
3	Tractor	0.088198	No detection	∞
	Sunflower	0.182185	No detection	∞
	Station 2	0.150716	No detection	∞
	Rush hour	0.155048	No detection	∞
4	Tractor	0.116134	No detection	∞
	Sunflower	0.185065	No detection	∞
	Station 2	0.150716	No detection	∞
	Rush hour	No detection	No detection	0
5	Tractor	0.1995	1.1970	0.9975
	Sunflower	0.1649	1.1970	1.0321
	Station 2	0.2262	0.5966	0.3703
	Rush hour	0.1867	0.5966	0.4099
6	Tractor	0.1836	0.5966	0.4130
	Sunflower	0.1541	0.5966	0.4425
	Station 2	0.1994	0.3973	0.1979

	Rush hour	0.1683	0.3973	0.2290
Average time ahead when both methods have detection				**0.5115**

Table 5: The response time comparison for proposed and the TCP-friendly schemes with 0.1 second RTCP interval – SD videos.

Case #	**Sequence Name**	**Response Time (Sec)**		**Difference**
		Proposed Method	**[41][38]**	
1	City	0.1708	0.4035	0.2327
	Crew	0.2734	0.2042	-0.0692
	Harbor	0.3283	0.4035	0.0752
	Ice	0.1982	0.4035	0.2053
2	City	0.1755	0.5027	0.3272
	Crew	0.2745	0.2042	-0.0703
	Harbor	No detection	0.6022	∞
	Ice	0.2043	0.4035	0.1992
Average time ahead when both methods have detection				**0.128586**

To conclude, in order to show the efficiency of our method, we have considered the case of an RTCP transmission interval of 0.2 seconds for HD-sized videos. The simulation results for this case are presented in Table 6. As the table shows, our proposed method outperforms the RTCP-based scheme in most simulation cases.

Table 6: The response time comparison for proposed and the TCP-friendly schemes with 0.2 second RTCP interval – HD videos.

Case #	Sequence Name	Response Time (Sec)		Difference
		Proposed Method	[93][62]	
1	Tractor	0.0943	No detection	∞
	Sunflower	0.1302	No detection	∞
	Station 2	0.0936	No detection	∞
	Rush hour	0.1332	No detection	∞
2	Tractor	0.1056	No detection	∞
	Sunflower	0.1872	No detection	∞
	Station 2	0.0936	No detection	∞
	Rush hour	0.1653	No detection	∞
3	Tractor	0.122249	No detection	∞
	Sunflower	0.164523	No detection	∞
	Station 2	0.137348	No detection	∞
	Rush hour	0.108936	No detection	∞
4	Tractor	0.401153	0.283389	-0.1178
	Sunflower	No detection	0.243406	∞
	Station 2	0.279008	0.203455	-0.0755
	Rush hour	No detection	0.103394	∞
5	Tractor	0.2110	3.0552	2.8442
	Sunflower	0.1496	3.0552	2.9056
	Station 2	0.4107	1.9919	1.5811
	Rush hour	0.2005	1.9919	1.7914
6	Tractor	0.1975	0.9321	0.7345
	Sunflower	0.1520	0.9321	0.7801
	Station 2	0.2537	0.9321	0.6784
	Rush hour	0.2082	0.9321	0.7238
Average time ahead when both methods have detection				**1.18459**

In addition, Figure 30 shows the graphs of $Diff$ with the generated upper and lower bounds for the rush hour video sequence, and the network condition is set based on cases 3–6 of Table 3. With the exception of only one case, our method detects changes efficiently. As Figure 30(b) shows, our method

does not detect the slight decrease in the bandwidth, which means that the amount of the decrease has a very slight effect on the delays and an insignificant effect on the inter-arrival delay. Although the curve of $Diff$ turns flat and then decreases, the behavior of our detection method is expected, because the curve does not cross the boundaries, which can be seen in Figure 30. The key part of the idea is based on the calculation of the inter-arrival time of the consequent packets and updating the Bayesian model accordingly. The main indicator of bandwidth change in Figure 30 is the blue curve (Diff). The blue curve shows the bandwidth changes almost immediately. The values of the bounds are not an integrated part of our detection system. These bounds can be chosen such that Figure 30(b) shows the detection occurs; however, we chose a lower sensitivity, which is why the figure shows no detection. It is important to note that for all cases depicted in Figure 30, the sending rate of the video is capped at 2 Mbps. Therefore, a decrease of 0.25 Mbps in bandwidth does not drastically affect the inter-arrival time of the subsequent packets or the video quality; however, an increase of the same size in the bandwidth would trigger a rush of packets and cause shorter inter-arrival times. Hence, the blue curve in Figure 30(a) starts increasing dramatically and follows this change. Our analysis shows that the Bayesian model developed in our study is more sensitive to the bandwidth increase than to sudden same-size decrease.

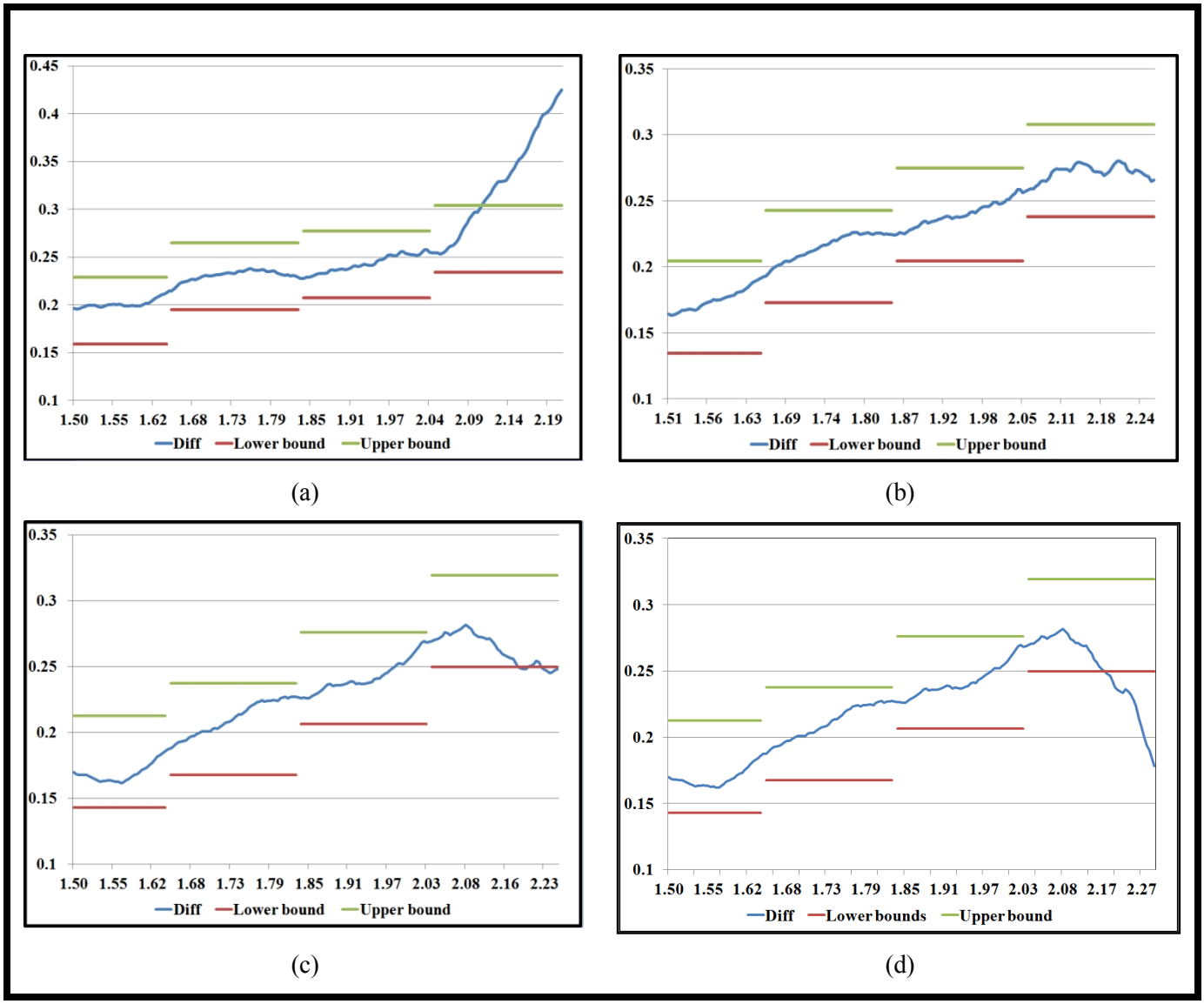


Figure 30: Graphs for our proposed approach for the rush hour video in various decreasing cases (a) case 3 of Table 3 (2 Mbps to 2.25 Mbps), (b) case 4 Table 3 (2 Mbps to 1.75 Mbps), (c) case 5 Table 3 (2 Mbps to 1.5 Mbps), and (d) case 6 Table 3 (2 Mbps to 1 Mbps).

5.3 COMMERCIAL TESTBED EXPERIMENTS

In addition to simulations, we have tested similar scenarios using a commercial HDVC system testbed provided by our partner Magor Corp. Figure 31 presents the experiment setup. In the figure, the participants send video to each other. The figure represents two parties in one HDVC session. A snapshot of this HDVC session is shown in Figure 32. Figure 33 is representing one way of the solution, where another way is identical to it. The proxy server is responsible for emulating various network conditions and policies. We have used *netem* in the proxy to emulate real network traffic patterns [96]. The netem scripts used can be found in the Appendix section of this thesis. To have a fixed reference to compare the results, a prerecorded (raw input) video of a real session has been used in the experiment. The experiments have been performed with multiple scenarios. In the first scenario, the video bitrate is initially set to 1.5 Mbps, and the proxy is configured to provide this bitrate at first and drop it to 1 Mbps after 7 seconds. In the second scenario, the video bitrate is initially set to 1.5 Mbps, and the proxy is configured to increase it to 2 Mbps after 7 seconds. The RTCP packets are sent every 200 ms and also our maximum packet size is 1500 bytes and sent around 166 packets per second. Figure 33 presents the results of the first scenario. Two variables are shown in the figure. The x-axis represents the time of the bandwidth change, with the negative and positive values indicating the time before and the time after the bandwidth change, respectively; i.e., the bandwidth change occurs at time zero. The y-axis represents the *Diff* variable according to equation (16). The *Diff* variable determines the behavior of the network in our proposed method, and crossing the lower bound and the upper bound indicates that the method has detected the bandwidth modification, which is shown in List 1. Figure 34 shows that our solution detects the fluctuation in available bandwidth about 160 milliseconds after the bandwidth modification.

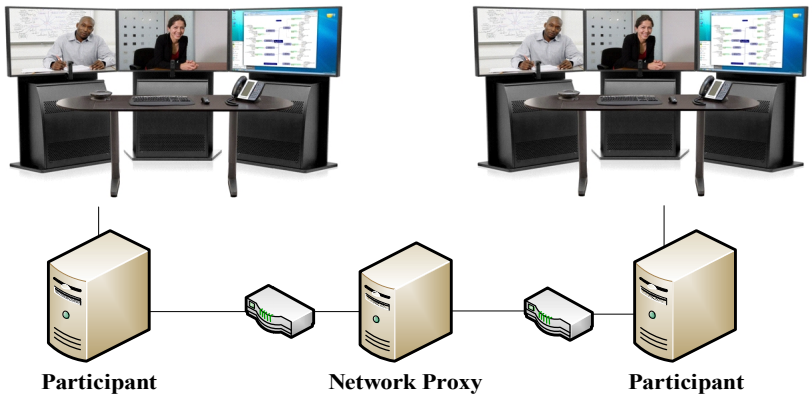


Figure 31: The experiment setup.

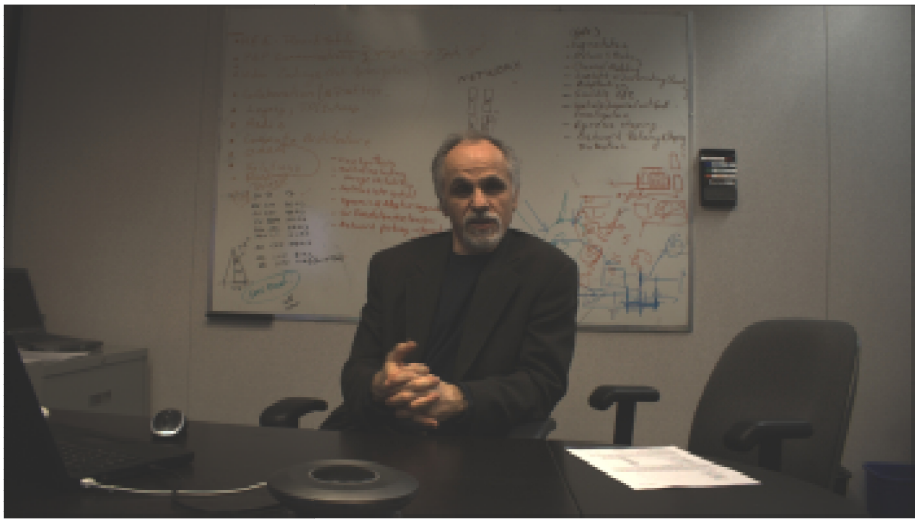


Figure 32: A snapshot of one HDVC session.

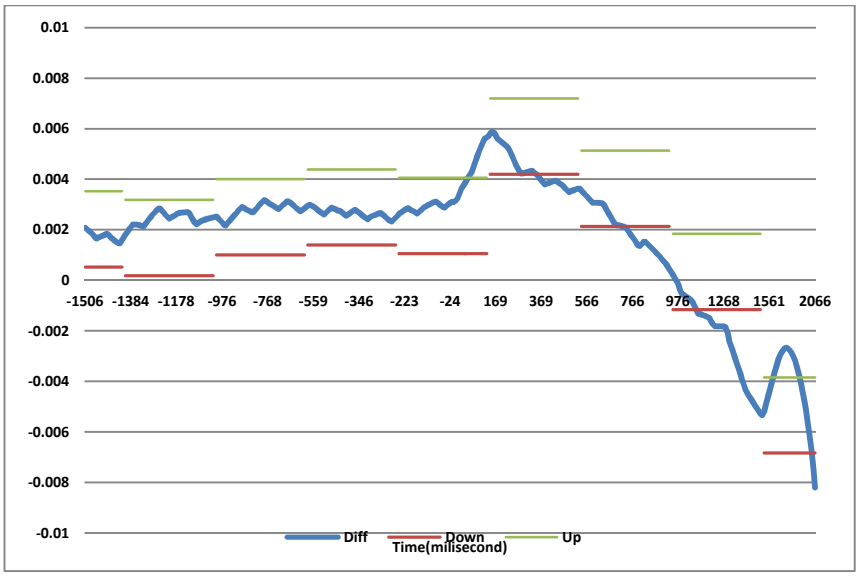


Figure 33: The performance of the proposed method when bandwidth changes from 1500 Kbps to 1000 Kbps with a 50 Kbits router buffer.

The available bandwidth calculated by the TCP-friendly method is depicted in Figure 34. We have used the same scenario in this test in which the bandwidth change also occurs at time zero. This method is unable to provide a clear and useful status for the bandwidth changes.

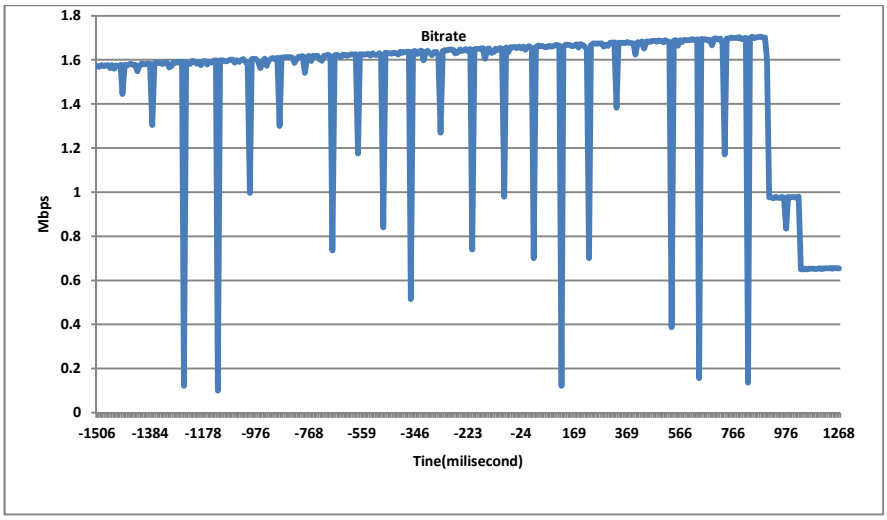


Figure 34: The amount of the available bandwidth calculated by the TCP-friendly method when the bandwidth changes from 1500 Kbps to 1000 Kbps with a 50 Kbits router buffer.

Figure 35 shows the results of the second scenario, in which our method detects the bandwidth increase again within 900 milliseconds. The TCP-friendly result is presented in Figure 36, which shows that the TCP-friendly method detects the bandwidth increase after 1200 milliseconds; however, the clear result appears after 9 seconds. This is expected because it is a conservative method.

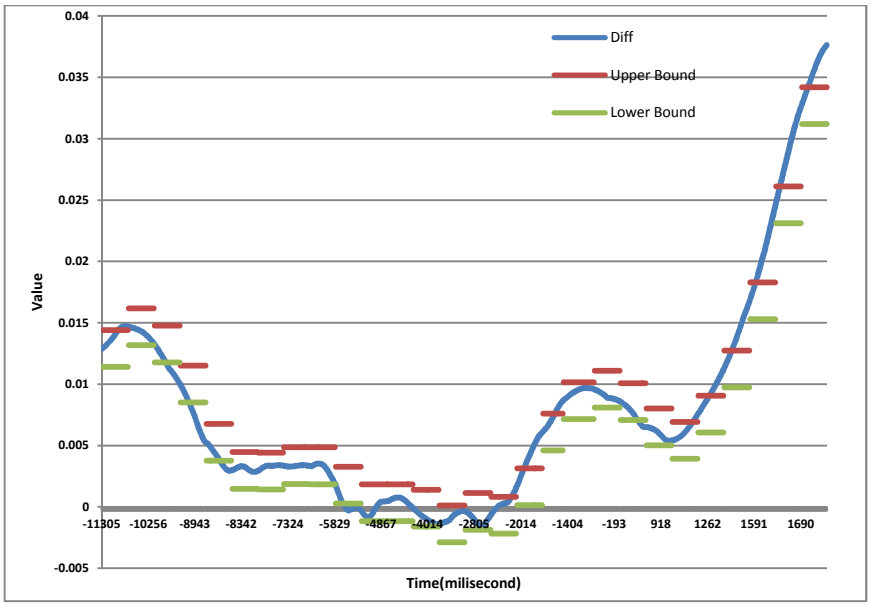


Figure 35: The performance of the proposed method when the bandwidth changes from 1500 Kbps to 2000 Kbps.

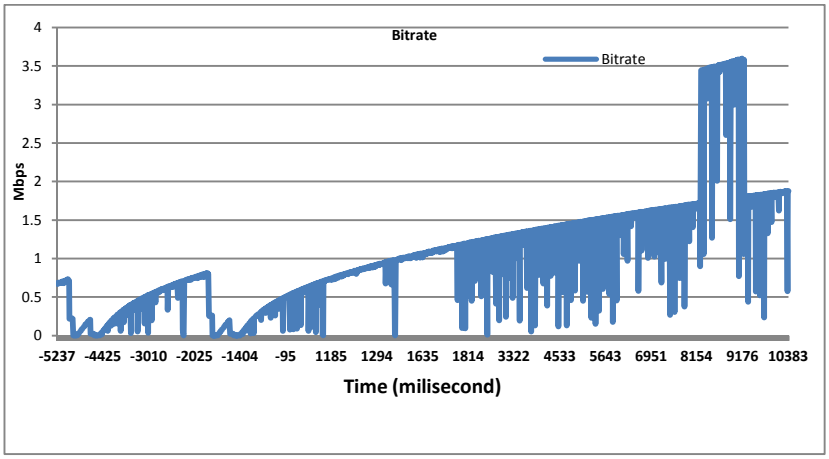


Figure 36: The amount of the available bandwidth calculated by the TCP-friendly method when the bandwidth changes from 1500 Kbps to 2000 Kbps.

In addition to these two scenarios, we have experimented with the following scenarios: 1500 Kbps

to 1250 Kbps, 1500 Kbps to 1750 Kbps, and 1500 Kbps to 2500 Kbps. All results are very similar to the results shown in the previous figures and are not reported here for brevity.

5.3.1 ROUTER'S QUEUE

The router's queue size affects the TCP-friendly method and our method. The large queue size imposes a delay in the TCP-friendly method as the packet loss decreases and the packet delay increases. Figure 37 and Figure 39 represent the TCP-friendly method with a 200 Kbits and a 100 Kbits buffer size in the router, and Figures 38 and 40 represent our method with a 200 Kbits and a 100 Kbits buffer size in the router. Comparing Figure 37 and Figure 38, it can be seen that TFRC detects the bandwidth reduction after 3600ms while our method detects the network reduction at 292ms after the bandwidth reduction. Again, the same story happens for Figure 39 and Figure 40. However the main difference is that TFRC is faster and detects network modification at 2500ms while we do not see any change in our method. The main outcome of these experiments is that our method is independent from the router's buffer size. Yet the TFRC is affected by the router's buffer size. The larger the buffer size, the greater the delay imposed by the TCP-friendly method; however, our proposed method is able to detect bandwidth modifications with a large buffer size as well.

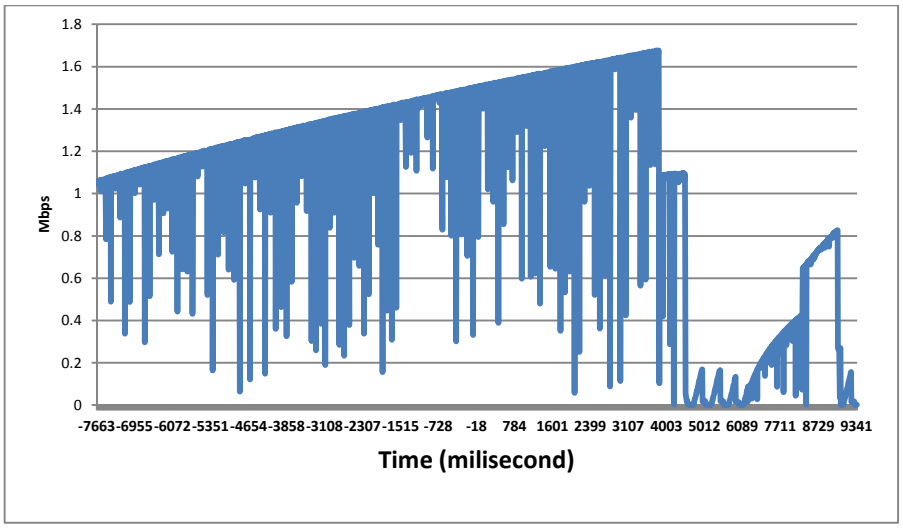


Figure 37: The amount of available bandwidth calculated by the TCP-friendly method when the bandwidth changes from 1500 Kbps to 1000 Kbps with a 200 Kbits buffer.

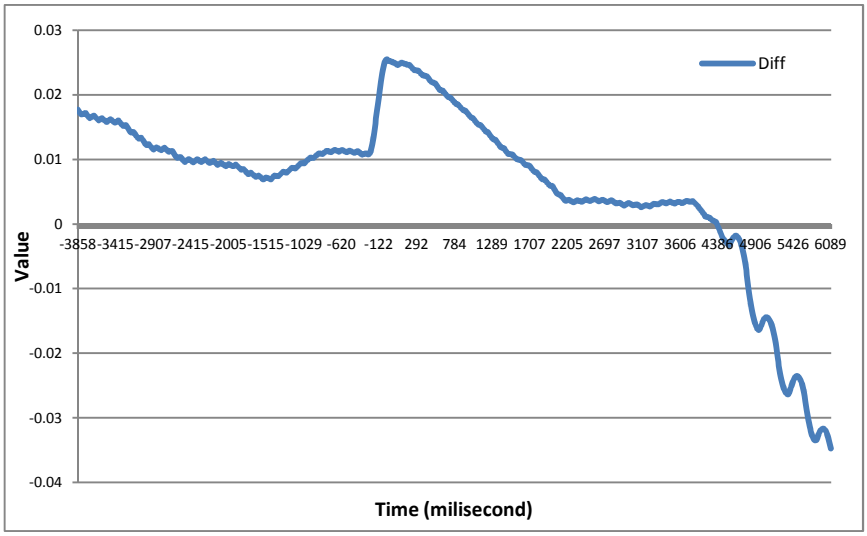


Figure 38: The performance of the proposed method when the bandwidth changes from 1500 Kbps to 1000 Kbps with a 200 Kbits buffer.

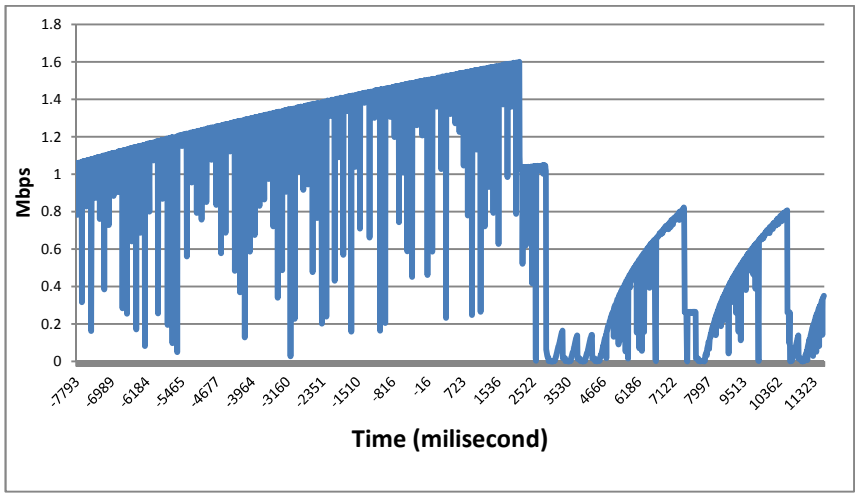


Figure 39: The amount of the available bandwidth calculated by the TCP-friendly method when the bandwidth changes from 1500 Kbps to 1000 Kbps with a 100 Kbits buffer.

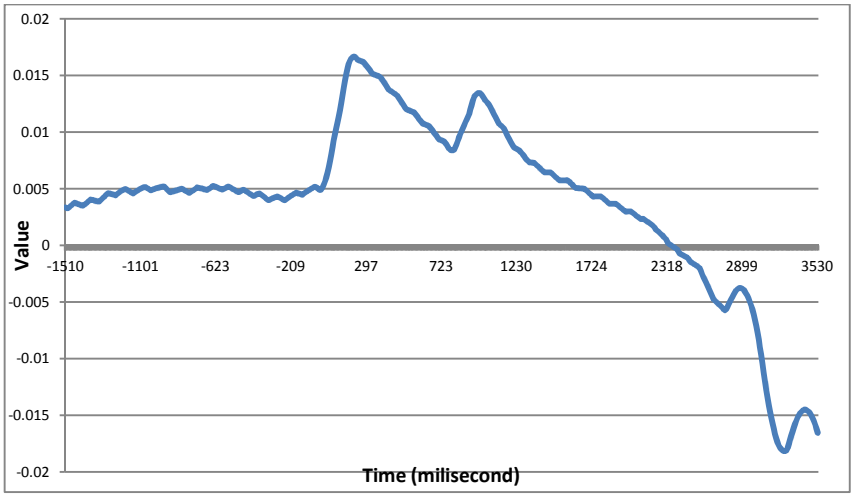


Figure 40: The performance of the proposed method when bandwidth changes from 1500 Kbps to 1000 Kbps with a 100 Kbits buffer.

5.4 REAL INTERNET TRIAL

Finally, we have evaluated our method using an actual HDVC session performed over the Internet. This is done in two stages. First, we have established one HDVC session between our DISCOVER Lab at the University of Ottawa and Magor's location in Kanata, a suburb of Ottawa. The computer at the university streams a recorded (raw input) HDVC session to the other computer. The network parameters such as loss event and inter-arrival delay are calculated and passed through to the TFRC and our method simultaneously. Each result generated by TFRC and our method is then stored in the file to make the comparison.

Figure 41 and Figure 42 compare the performance of our method over the Internet with that of the TCP-friendly method over the Internet. Figure 41 shows that our method detects the bandwidth reduction at 9973 milliseconds of the experiment, whereas the TCP-friendly method does so at 10703 milliseconds, 730 milliseconds later than our method. Therefore, as expected, our method measures the bandwidth reduction faster than the other methods, leading to the possibility of adjusting the video bitrate sooner with a higher quality experience for the end user.

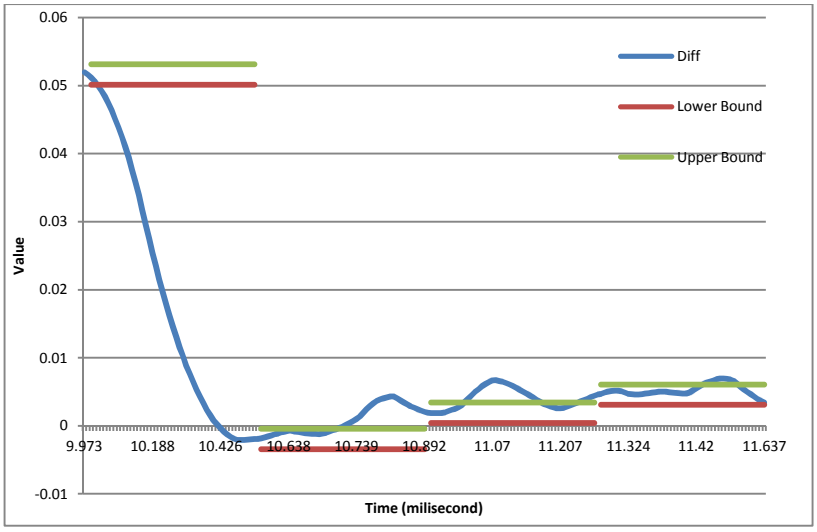


Figure 41: The real trace of the proposed method over the Internet (stage 1).

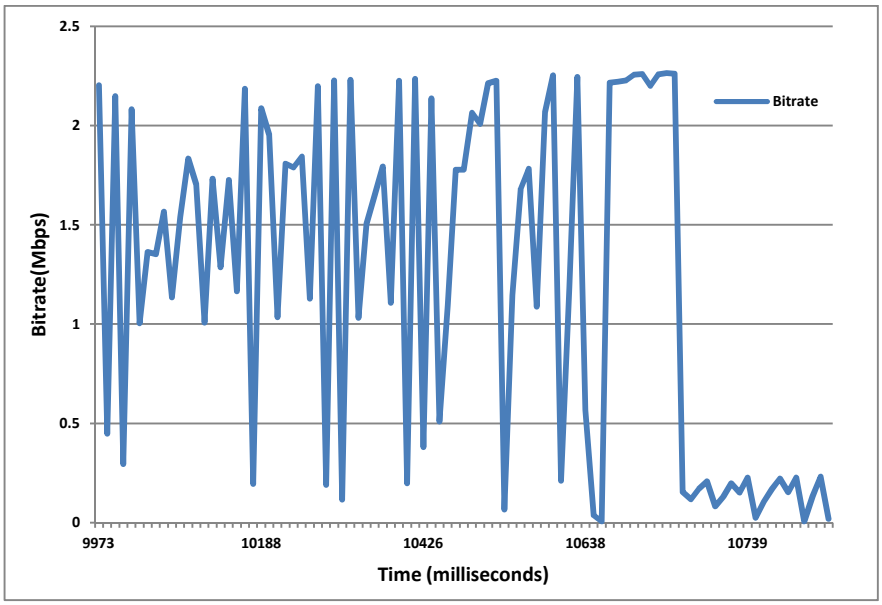


Figure 42: The real trace of the TCP-friendly method over the Internet (stage 1).

In the second stage, we have established another HDVC session between two nodes. These two nodes are connected to the other two nodes via an ADSL connection. The network topology is depicted in Figure 43. We have applied the same procedure that was explained in the first stage. Figure 44 and Figure 45 compare the performance of our method over the Internet with that of the TCP-friendly method. Figure 44 shows that our method detects the bandwidth reduction at 1931 milliseconds of the experiment, whereas the TCP-friendly method does so at 1978 milliseconds, 47 milliseconds later than our method. Even in this situation, in which the network was too unstable, our method measures the bandwidth reduction faster than the other methods; however, the performance of our system in the first stage is much better than it is in the second stage because the first stage is more realistic.

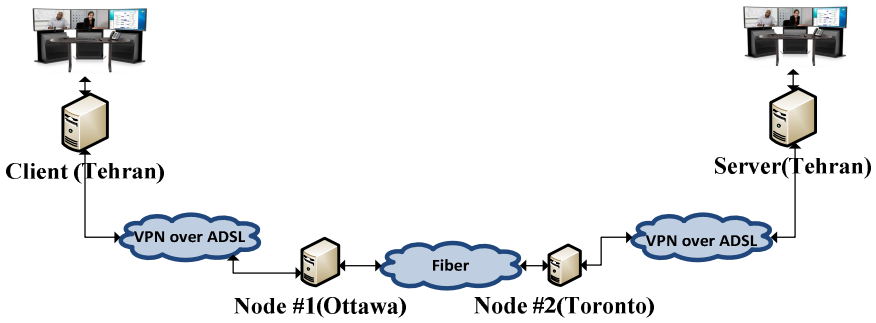


Figure 43: The network topology.

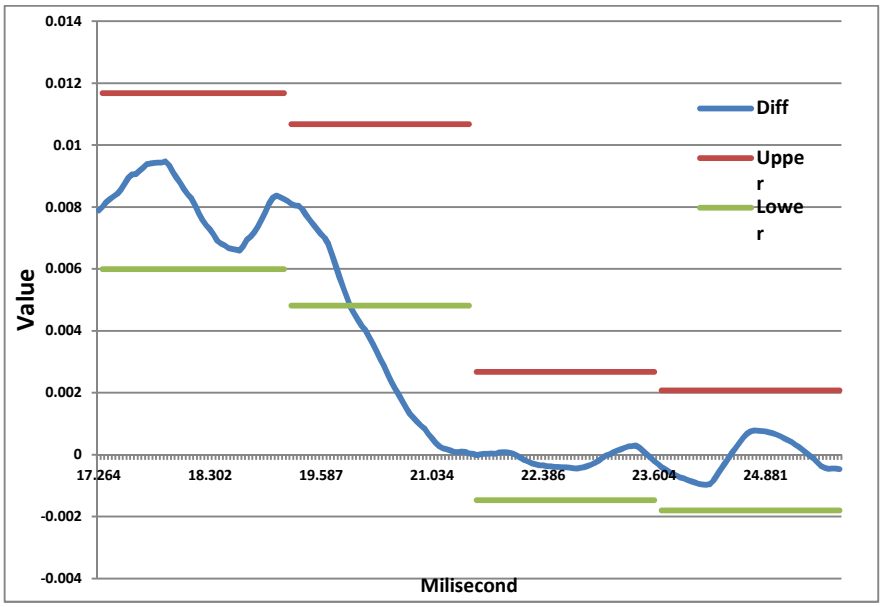


Figure 44: The real trace of the proposed method over the Internet (stage 2).

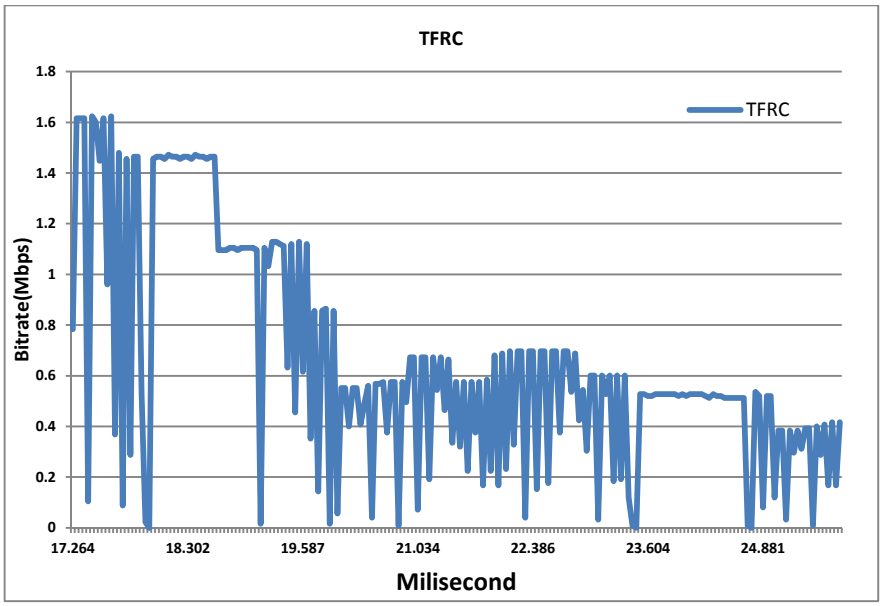


Figure 45: The real trace of the TCP-friendly method over the Internet (stage 2).

CHAPTER 6

CONCLUSION AND FUTURE RESEARCH

In this thesis, we have proposed two main ideas related to a video conferencing solution: rate control and congestion prediction. Rate control is the key part of the encoder and affects its overall performance. We have proposed the DCRF and DRC methods. DRC is more important because it is designed specifically for HDVC. The simulations have shown that it outperforms the existing rate control methods. To achieve a complete solution, we have introduced a congestion prediction method. The main feature of our method is its ability to predict congestion and to enable the encoder to adapt the video quality. The simulations have demonstrated that our method indeed outperforms its competitors. To obtain more accurate results, we also have implemented a real-world solution (the Magor system) and have generated results from a real trace. Each simulation has also been described and analyzed. The results have confirmed that the DRC and the congestion prediction improve overall video quality and user satisfaction.

Though this study has been focused on the videoconferencing, rate control is a very important concept in the video communications and can be optimized in different aspects. Rate control is an ongoing problem. For each new type of communications, such as wired, wireless, 3G, LTE, and optic, rate control is the required component of the system, and optimal rate control should take advantage of the communication infrastructure efficiently. The list below outlines some avenues for the future research.

- Distributed DRC: This is an extension of the DRC that works for HDVC with more than two streams at the same time while using the shared bandwidth.
- Congestion monitor: This is the complement of the congestion prediction method and is used to calculate the accurate amount of the bandwidth that each stream is using in one HDVC session. The key element is distributed DRC.
- DASH: A simplified DRC can be used in this system. The main feature of DASH is that the QP selection is discrete and coarse, which makes a DRC task much simpler.

- Multiview videoconferencing: This is a new area in videoconferencing that offers multiviews instead of a single view, and a hierarchical DRC would be useful for this situation.
- WebRTC: This is web-based videoconferencing. This feature enables web browsers to have real-time communication capabilities by simply using JavaScript API. This project has been initiated by Google and is open source. The WebRTC uses GCC that can determine the available bandwidth, yet the main concern on the WebRTC is that how current RCs are able to adapt their bitrate accordingly. The close collaboration between RC and GCC is the key element of the WebRTC success and the DRC can play a key role. The DRC adapts its bitrate within 2 to 6 frames. Moreover, our network prediction method can determine the network behavior which complements the WebRTC feature.

REFERENCES

[1] L. Amico, "Internet video consumption trends for 2009," *ComScore*, 2010. [Online]. Available: http://lamanagementco.com/featured-articles/internet-video-consumption-trends.

[2] P. Gevros, J. Crowcroft, P. Kirstein, and S. Bhatti, "Congestion Control Mechanisms and the Best Effort Service Model," *Network, IEEE*, vol. 15, no. 3, pp. 16–26, 2001.

[3] Polycom, "HIGH-DEFINITION : The Evolution of video conferencing," 2005. [Online]. Available: http://www.jkcit.co.uk/pdf/polycom-hd-videoconferencing-whitepaper.pdf.

[4] Cisco, "TelePresence," 2014. [Online]. Available: http://www.cisco.com/web/CA/solutions/tele_need.html.

[5] Lifesize, "LifeSize Solutions," 2014. [Online]. Available: http://www.lifesize.com/en/solutions.

[6] Polycom, "Polycom RealPresence Immersive: Video Conferencing & Telepresence Solutions," 2014. [Online]. Available: http://www.polycom.com/products-services/hd-telepresence-video-conferencing/realpresence-immersive.html.

[7] Magor, "Aerus," 2014. [Online]. Available: www.magorcorp.com.

[8] Vidyo Personal Telepresence, "VidyoRoom ™ Administrator and User Guide," 2013. [Online]. Available: http://www.vidyo.com/wp-content/uploads/2013/11/VidyoRoom_Admin_and_User_Guide_2.2.2-A.pdf.

[9] G. Huston, "Best Efforts Networking," *potaroo*, 2001. [Online]. Available: http://www.potaroo.net/ispcol/2001-09/2001-09-best.pdf.

[10] J. Postel, "RFC 793:Transmission Control Protocol," *DARPA Internet Program Protocol Specification, RFC 791, USC/Information Sciences Institute*, 1981. [Online]. Available: ftp://ftp.rfc-editor.org/in-notes/rfc793.txt.

[11] J. Postel, “User Datagram Protocol (RFC 768),” *DARPA Network Working Group Rep. RFC-768, U.S.C. Inform. Sci. Inst*, 1980. [Online]. Available: http://www.rfc-editor.org/in-notes/rfc768.txt.

[12] A. Javadtalab, M. Omidyeganeh, S. Shirmohammadi, and M. Hosseini, “On the Suitability of Current x264 Rate Controller Algorithms for High Definition Video Conferencing,” in *International Symposium on Artificial Intelligence and Signal Processing (AISP)*, 2011, pp. 107–112.

[13] E. Casilari, a. Reyes, a. Díaz-Estrella, and F. Sandoval, “Heavy-tailed distribution of scene duration in VBR video,” *Electron. Lett.*, vol. 35, no. 2, p. 134, 1999.

[14] A. Kamel, A. Al-fuqaha, D. Kountanis, and I. Khalil, “Towards a client-side QoS monitoring and assessment using Generalized Pareto Distribution in a cloud-based environment,” in *IEEE Workshops on Wireless Communications and Networking Conference Workshops (WCNCW)*, 2013, pp. 123–128.

[15] A. Javadtalab, X. Zhu, and S. Shirmohammadi, “Demo Paper : A Fast-Adjusting Rate Control Algorithm Using NADA for HD Video Conferencing,” in *IEEE International Symposium on Multimedia*, 2014, pp. 4–5.

[16] A. Javadtalab, S. Shirmohammadi, and M. Hosseini, “DEMO PAPER : A fast-adjusting high-quality rate control algorithm for HD video streaming,” in *IEEE International Conference on Multimedia and Expo*, 2013, pp. 3–4.

[17] M. Baldi, P. Torino, and Y. Ofek, “End-to-end delay of videoconferencing over packet switched networks,” in *IEEE Computer and Communications Societies*, 1998, vol. 3, pp. 1084–1092.

[18] T. Wiegand, G. J. Sullivan, S. Member, G. Bjøntegaard, and A. Luthra, “Overview of the H . 264 / AVC Video Coding Standard,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, 2003.

[19] S. Ma, W. Gao, and Y. Lu, “Rate-distortion analysis for H.264/AVC video coding and its application to rate control,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 12, pp. 1533–1544, 2005.

[20] H. Wang and S. Kwong, “Rate-Distortion Optimization of Rate Control for H.264 With Adaptive Initial Quantization Parameter Determination,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 1, pp. 140–144, 2008.

[21] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, “A Novel Rate Control Technique for Multiview Video Plus Depth Based 3D Video Coding,” *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 562–571, 2011.

[22] J. Sun, Y. Duan, J. Li, J. Liu, and Z. Guo, “Rate-Distortion Analysis of Dead-Zone Plus Uniform Threshold Scalar Quantization and Its Application- Part II: Two-Pass VBR Coding for H.264/AVC,” *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 215–228, 2013.

[23] F. Shao, G. Jiang, A. Member, W. Lin, S. Member, and M. Yu, “Joint Bit Allocation and Rate Control for Coding Multi-View Video Plus Depth Based 3D Video,” *IEEE Trans. Multimed.*, vol. 15, no. 8, pp. 1843–1854, 2013.

[24] M. M. Ghandi and M. Ghanbari, “A Lagrangian optimized rate control algorithm for the H.264/AVC encoder,” in *International Conference on Image Processing*, 2004, vol. 1, pp. 123–126 Vol. 1.

[25] S. K. Im, M. M. Ghandi, and C. T. Lam, “Non-integer bit estimation for rate-distortion optimized video coding,” in *IEEE International Conference onConsumer Electronics (ICCE)*, 2012, pp. 88–89.

[26] Z. Xie, Z. Bao, C. XU, and G. Zhang, “Optimal bit allocation and efficient rate control for H64/AVCsed on general rate-istortion model and enhanced coding complexity measure,” *IET Trans. Image Process.*, vol. 4, no. 3, pp. 172–183, 2010.

[27] G. Lin, S. Zheng, and J. Hu, “A two-stage p-domain rate control scheme for H.264 encoder,” in *IEEE International Conference on Multimedia and Expo, 2008*, 2008, pp. 713–716.

[28] D. Vatolin, “List of MPEG-4 AVC/H.264 video codecs,” *doom9.org*, 2007. [Online]. Available: http://forum.doom9.org/showthread.php?t=95939.

[29] X264, “x264,” *videolan*, 2014. [Online]. Available: http://developers.videolan.org/x264.htm.

[30] Mpeg-4, “MPEG-4 AVC/H.264 video codec comparison,” *CS MSU Graphics & Media Lab Video*, 2013. [Online]. Available: http://www.compression.ru/video/index.htm.

[31] M. Van der Schaar and P. A Chou, *Multimedia Over IP And Wireless Networks Compression Networking And Systems*, First. Academic Press, 2007, p. 712.

[32] S. Wu, Y. Huang, and T. Ikenaga, “A Macroblock-Level Rate Control Algorithm for H.264/AVC Video Coding with Context-Adaptive MAD Prediction Model,” in *International Conference on Computer Modeling and Simulation*, 2009, pp. 124–128.

[33] D. An, X. Tong, and Y. He, “A Novel R-D Optimized MB-level Rate Control Scheme for Real-Time Video Coding,” *J. Signal Process. Syst.*, vol. 74, no. 2, pp. 175–187, 2014.

[34] D. Zhao, Y. Zhou, D. Wang, and J. Mao, “Effective macroblock layer rate control algorithm for H.264/AVC,” *Elsevier J. Comput. Electr. Eng.*, vol. 37, no. 4, pp. 550–558, Jul. 2011.

[35] M. Li, Y. Chang, F. Yang, S. Wan, S. Lin, and L. Xiong, “Frame layer rate control for H.264/AVC with hierarchical B-frames,” *Elsevier J. Signal Process. Image Commun.*, vol. 24, no. 3, pp. 177–199, Mar. 2009.

[36] X. Wang, S. Kwong, and Y. Zhang, “Applying Game Theory to Rate Control Optimization for Hierarchical B-Pictures,” *IEEE Trans. Broadcast.*, vol. 59, no. 4, pp. 591–601, 2013.

[37] V. P. Binh and S.-H. Yang, “A better bit-allocation algorithm for H.264/SVC,” in *the Fourth Symposium on Information and Communication Technology - (SoICT ’13)*, 2013, pp. 18–26.

[38] H. Yuan, J. Liu, J. Ren, and Y. Li, “Coding modes-based frame skip avoidance scheme for low bit rate video coding,” *J. Real-Time Image Process.*, vol. 9, no. 4, pp. 609–619, 2014.

[39] M. Jiang, X. Yi, and N. Ling, “Improved frame-layer rate control for H.264 using MAD ratio,” in *IEEE International Symposium on Circuits and Systems*, 2004, vol. 3, pp. III–813–16 Vol.3.

[40] Y. Wu, S. Yoon, D. Park, and J. Yang, “An enhanced initialization method for rate control of H.264 based on image quality balance,” in *International Conference on ICT Convergence*, 2011, pp. 727–732.

[41] A. Javadtalab, M. Omidyeganeh, S. Shirmohammadi, and M. Hosseini, “A Rate Control Algorithm for X264 High Definition Video Conferencing,” in *International Conference on Multimedia and Expo*, 2011, pp. 1–6.

[42] S. Park, “GOP Level Rate-Control for Real-Time Video Transmission,” *Int. J. Multimed. Ubiquitous Eng.*, vol. 8, no. 4, pp. 207–216, 2013.

[43] L. Zheng, L. Tian, and Y. Wu, “A rate control scheme for distributed high performance video encoding in cloud,” in *International Conference on Cloud and Service Computing (CSC ’11)*, 2011, pp. 131–133.

[44] M. Jiang and N. Ling, “Low-Delay Rate Control for Real-time H.264/AVC video coding,” *IEEE Trans. Multimed.*, vol. 8, no. 3, pp. 467–477, 2006.

[45] C. Huang, C. Lin, and C. Chuang, “A Multilayered Audiovisual Streaming System Using the Network Bandwidth Adaptation,” *IEEE Trans. Multimed.*, vol. 11, no. 5, pp. 797–809, 2009.

[46] L. Shen, Z. Liu, and Z. Zhang, “A novel H.264 rate control algorithm with consideration of visual attention,” *Multimed. Tools Appl.*, vol. 63, no. 3, pp. 709–727, Oct. 2011.

[47] H.-M. Hu, W. Lin, B. Li, and M.-T. Sun, “A region-based rate-control scheme using inter-layer information for H.264/SVC,” *J. Vis. Commun. Image Represent.*, vol. 22, no. 7, pp. 615–626, Oct. 2011.

[48] X. Lan, N. Zheng, W. Ma, M. Hui, and J. Xue, “Arbitrary ROI-Based Wavelet Video Coding,” in *Data Compression Conference*, 2010, no. 2009, pp. 537–537.

[49] B. Xiong, X. Fan, C. Zhu, S. Member, X. Jing, and Q. Peng, “Face Region Based Conversational Video Coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 917–931, 2011.

[50] S.-F. Huang, M.-J. Chen, and M.-S. Li, “Region-of-interest segmentation based on Bayesian theorem for H.264 video transcoding,” in *IEEE Visual Communications and Image Processing (VCIP)*, 2011, pp. 1–4.

[51] Y.-B. Lin and X.-M. Zhang, “Recent developments in perceptual video coding,” in *International Conference on Wavelet Analysis and Pattern Recognition*, 2013, pp. 259–264.

[52] X. Wang, L. Su, H. Qi, Q. Huang, and G. Li, “Face Distortion Recovery Based on Online Learning Database for Conversational Video,” *IEEE Trans. Multimed.*, vol. 16, no. 8, pp. 2130–2140, 2014.

[53] Y. Zhang, W. Huangfu, K. Li, and C. Xu, “A refined rate allocation scheme with adaptive playback adjustment for robust hd video stream transmission,” in *Proceedings of the 15th International Conference on Multimedia*, 2007, pp. 537–540.

[54] K. D. Huang, K. R. Duffy, and D. Malone, “H-RCA: 802.11 Collision-Aware Rate Control,” *IEEE/ACM Trans. Netw.*, vol. 21, no. 4, pp. 1021–1034, Aug. 2013.

[55] M. Baldi and Y. Ofek, “End-to-end delay analysis of videoconferencing over packet-switched networks,” *IEEE/ACM Trans. Netw.*, vol. 8, no. 4, pp. 479–492, 2000.

[56] L. Merritt and R. Vanam, “Improved Rate Control and Motion Estimation for H.264 Encoder,” in *IEEE International Conference on Image Processing*, 2007, vol. 5, pp. V – 309–V – 312.

[57] M. Semsarzadeh, M. J. Langroodi, and M. R. Hashemi, “An adaptive rate control for faster bitrate shaping in x264 based video conferencing,” in *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2010, pp. 1–4.

[58] M. Lee, “Video Traffic Prediction Based on Source Information and Preventive Channel Rate Decision for RCBR,” *IEEE Trans. Broadcast.*, vol. 52, no. 2, pp. 173–183, Jun. 2006.

[59] S. Floyd and K. Fall, “Promoting the Use of End-to-End Congestion Control in the Internet,” *IEEE/ACM Trans. Netw.*, vol. 7, no. 4, pp. 458–472, 1999.

[60] N. Singhal and R. M. Sharma, “Survey on TCP Friendly Congestion Control for Unicast and Multicast Traffic,” *Int. J. Comput. Appl.*, vol. 26, no. 1, pp. 23–30, 2011.

[61] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-based congestion control for unicast applications,” in *Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2000, pp. 43–56.

[62] M. Handley, J. Padhye, and J. Widmer, “TCP Friendly Rate Control (TFRC): Protocol Specification,” *Network Working Group*, 2008. .

[63] J. Widmer and M. Handley, “Extending equation-based congestion control to multicast applications,” in *conference on Applications, technologies, architectures, and protocols for computer communications*, 2001, pp. 275–285.

[64] F. Hernandez-Campos, J. S. Marron, G. Samorodnitsky, and F. D. Smith, “Variable heavy tailed durations in Internet traffic. Part I. Understanding heavy tails,” in *EEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, 2002, pp. 43–50.

[65] W.-B. Gong, Y. Liu, V. Misra, and D. Towsley, “Self-similarity and long range dependence on the internet: a second look at the evidence, origins and implications,” *Elsevier J. Comput. Networks*, vol. 48, no. 3, pp. 377–399, Jun. 2005.

[66] A. J. Field, U. Harder, and P. O. Harrison, “Measurement and modelling of self-similar traffic in computer networks,” *IEE Proceedings-Communications*, vol. 151, no. 4, pp. 355–363, 2004.

[67] T. Field, U. Harder, and P. Harrison, “Network traffic behaviour in switched Ethernet systems,” *Elsevier J. Perform. Eval.*, vol. 58, no. 2–3, pp. 243–260, Nov. 2004.

[68] G. Giorgi and C. Narduzzi, “A Study of Measurement-Based Traffic Models for Network Diagnostics,” *IEEE Trans. Instrum. Meas.*, vol. 57, no. 8, pp. 1642–1650, Aug. 2008.

[69] M. Bertocco, R. Tittoto, E. Rizzi, and L. Benetazzo, “Statistical Analysis of Measurements for Telecommunication-Network Troubleshooting,” in *IEEE Conference on Instrumentation and Measurement Technology*, 2002, no. May, pp. 21–23.

[70] D. Zhang, S. Member, and D. Ionescu, “Measurement and Control of Packet Loss Probability for MPLS VPN Services,” *IEEE Trans. Instrum. Meas.*, vol. 55, no. 5, pp. 1587–1598, 2006.

[71] D. Zhang and D. Ionescu, “A New Method for Measuring Packet Loss Probability Using a Kalman Filter,” *IEEE Trans. Instrum. Meas.*, vol. 58, no. 2, pp. 488–498, 2009.

[72] A. Vakili and J.-C. Gregoire, “Real-Time Packet Loss Probability Estimates from IP Traffic Parameters,” *Int. J. Adv. Networks Serv.*, vol. 5, no. 1, pp. 34–42, 2012.

[73] D. Zhang and D. Ionescu, “Online Packet Loss Measurement and Estimation for VPN-Based Services,” *IEEE Trans. Instrum. Meas.*, vol. 59, no. 8, pp. 2154–2166, 2010.

[74] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk, “Multifractal cross-traffic estimation,” in *ITC Specialist Seminar on IP Traffic Measurement, Modeling, and Management*, 2000, pp. 1–15.

[75] B. Melander, M. Bjorkman, and P. Gunningberg, “A new end-to-end probing and analysis method for estimating bandwidth bottlenecks,” in *IEEE. Global Telecommunications Conference*, 2000, vol. 1, no. 1, pp. 415–420.

[76] M. Jain and C. Dovrolis, “End-to-End Available Bandwidth : Measurement Methodology , Dynamics , and Relation With TCP Throughput,” *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 537–549, 2003.

[77] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, “pathChirp : Efficient Available Bandwidth Estimation for Network Paths,” in *Workshop on Passive and Active Monitoring*, 2003, pp. 1–11.

[78] M. Jain and C. Dovrolis, “Pathload : a measurement tool for end-to-end available bandwidth,” in *Passive and Active Measurements (PAM) Workshop*, 2002, pp. 14–25.

[79] J. Strauss, D. Katabi, and F. Kaashoek, “A measurement study of available bandwidth estimation tools,” in *The conference on Internet measurement conference*, 2003, p. 39.

[80] V. Paxson, “End-to-end Internet packet dynamics,” *IEEE/ACM Trans. Netw.*, vol. 7, no. 3, pp. 277–292, Jun. 1999.

[81] X. Zhu and B. Girod, “Distributed Media-Aware Rate Allocation for Wireless Video Streaming,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 11, pp. 1462–1474, Nov. 2010.

[82] X. Zhu, T. Schierl, T. Wiegand, and B. Girod, “Distributed Media-Aware Rate Allocation for Video Multicast Over Wireless Networks,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 9, pp. 1181–1192, Sep. 2011.

[83] L. De Cicco, G. Carlucci, and S. Mascolo, “Understanding the Dynamic Behaviour of the Google Congestion Control for RTCWeb,” in *IEEE Packet Video Workshop*, 2013, pp. 1–8.

[84] A. Khanchi, M. Semsarzadeh, A. Javadtalab, and S. Shirmohammadi, “Continuous one-way available bandwidth change detection in high definition video conferencing,” in *ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2013, pp. 25–30.

[85] L. De Vito, S. Rapuano, and L. Tomaciello, “One-Way Delay Measurement: State of the Art,” *IEEE Trans. Instrum. Meas.*, vol. 57, no. 12, pp. 2742–2750, Dec. 2008.

[86] I. Phillips, D. Parish, M. Sandford, O. Bashir, and A. Pagonis, “Architecture for the management and presentation of communication network performance data,” *IEEE Trans. Instrum. Meas.*, vol. 55, no. 3, pp. 931–938, 2006.

[87] M. . Crovella and A. Bestavros, “Self-similarity in World Wide Web traffic: evidence and possible causes,” *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, 1997.

[88] K. Krishnamoorthy, *Handbook of statistical distributions with applications*. Chapman & Hall, 2006, p. 376.

[89] V. Paxson and S. Floyd, “Wide area traffic: the failure of Poisson modeling,” *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun. 1995.

[90] A. K. Bansal, *Bayesian Parametric Inference*. Alpha Science Intl Ltd, 2007, p. 400.

[91] M. A. Ferreira and P. Santa-Clara, “Forecasting stock market returns: The sum of the parts is more than the whole,” *J. financ. econ.*, vol. 100, no. 3, pp. 514–537, Jun. 2011.

[92] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessment : From Error Visibility to Structural Similarity,” *IEEE Trans. image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[93] H. Schulzrinne, S. Casner, H. Frederick, and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications, RFC3550 standard, 2003.,” *Network Working Group*, 2003. .

[94] “NS-2,” *Network Simulator version 2 (NS-2)*, 2011. [Online]. Available: http://www.isi.edu/nsnam/ns/.

[95] H. Gharavi, K. Ban, and J. Cambiotis, “RTCP-based frame-synchronized feedback control for IP-video communications over multipath fading channels,” in *IEEE International Conference on Communications*, 2004, pp. 1512–1516.

[96] Linux Foundation, “Netem,” 2009. [Online]. Available: http://www.linuxfoundation.org/collaborate/workgroups/networking/netem.

APPENDIX

8.1 NETEM SCRIPT:

```
#  Written By Abbas Javadtalab
# No buffering, filtering, or other magic:  if data arrives higher
# than the police rate rule,  it gets dropped outright.

# apply a netem qdisc  for ingress only:  note FFFF: appears to be the only acceptable handle
# nothing else worked...    FFFF: is acceptable for each interface, handles
# are unique per ethernet device
#set -v
avocadoInterface=eth0
bananaInterface=eth2
avocadoIP=0.0.0.0/0
ratelimit=$1
delay_line=5ms
delay_threshold=0
buffersize=75k
burst=400k
#input
tc qdisc add dev $avocadoInterface handle ffff: ingress
tc qdisc add dev $bananaInterface handle 1: root htb
tc class add dev $bananaInterface parent 1: classid 1:1 htb rate 100Mbps
tc class add dev $bananaInterface parent 1:1 classid 1:11 htb rate 100Mbps
tc  qdisc  add  dev  $bananaInterface  parent  1:11  handle  10:  netem  delay    $delay_line
$delay_threshold
tc filter add dev $bananaInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11

#output
tc qdisc add dev $bananaInterface handle ffff: ingress
tc qdisc add dev $avocadoInterface handle 1: root htb
tc class add dev $avocadoInterface parent 1: classid 1:1 htb rate 100Mbps
tc class add dev $avocadoInterface parent 1:1 classid 1:11 htb rate 100Mbps
#tc  qdisc  add  dev  $avocadoInterface  parent  1:11  handle  10:  netem  delay    $delay_line
$delay_threshold
```

```
tc filter add dev $avocadoInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11

tc  filter  add  dev  $avocadoInterface  parent  ffff:  protocol  ip  prio  50  u32  match  ip  src
$avocadoIP police rate $ratelimit  burst $burst drop flowid :1

tc filter add dev $bananaInterface parent ffff: protocol ip prio 50 u32 match ip src $avocadoIP
police rate $ratelimit  burst $burst drop flowid :1

# dump some stats about what we've done
tc -s -d qdisc show dev $avocadoInterface;
#set ="foo"
read input
while [ "$input" != c ]
do
echo "Type c to clear impairment and exit or enter new rate"
ratelimit=$input
echo "$ratelimit"
#input
tc  filter  del  dev  $avocadoInterface  parent  ffff:  protocol  ip  prio  50  u32  match  ip  src
$avocadoIP police rate $ratelimit  burst $burst drop flowid :1
tc  filter  add  dev  $avocadoInterface  parent  ffff:  protocol  ip  prio  50  u32  match  ip  src
$avocadoIP police rate $ratelimit  burst $burst drop flowid :1
tc filter del dev $bananaInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11
tc filter add dev $bananaInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11

#output
tc filter del dev $bananaInterface parent ffff: protocol ip prio 50 u32 match ip src $avocadoIP
police rate $ratelimit  burst $burst drop flowid :1
tc filter add dev $bananaInterface parent ffff: protocol ip prio 50 u32 match ip src $avocadoIP
police rate $ratelimit  burst $burst drop flowid :1
tc filter del dev $avocadoInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11
tc filter add dev $avocadoInterface  protocol ip prio 2 u32 match ip src $avocadoIP police rate
$ratelimit burst $burst drop flowid 1:11


#dump the stats again as we exit
tc -s -d qdisc show dev $avocadoInterface;
```

```
tc -s -d qdisc show dev $bananaInterface;
read input
done
ratelimit=$1
# and clean up and exit
#input
tc filter del dev $bananaInterface protocol ip prio 2 u32 match ip src 0.0.0.0/0 police rate
$ratelimit burst $burst drop flowid 1:11
tc filter del dev $avocadoInterface protocol ip prio 2 u32 match ip src 0.0.0.0/0 police rate
$ratelimit burst $burst drop flowid 1:11
tc qdisc del dev $bananaInterface parent 1:11 handle 10: netem delay $delay_line
$delay_threshold
tc qdisc del dev $avocadoInterface handle ffff: ingress
tc qdisc del dev $bananaInterface handle ffff: ingress
tc qdisc del dev $bananaInterface handle 1: root htb
tc qdisc del dev $avocadoInterface handle 1: root htb
#output
```