ADAPTATION STRATEGIES FOR STREAMING SVC VIDEO

Burak Görkemli and A. Murat Tekalp

College of Engineering, Koç University, 34450 Istanbul, Turkey
bgorkemli@ku.edu.tr

ABSTRACT

This paper aims to determine the best rate adaptation strategy to maximize the received video quality when streaming SVC video over the Internet. Different bandwidth estimation techniques are implemented for different transport protocols, such as using the TFRC rate when available or calculating the packet transmission rate otherwise. It is observed that controlling the rate of packets dispatched to the transport queue to match the video extraction rate resulted in oscillatory behavior in DCCP CCID3, decreasing the received video quality. Experimental results show that video should be sent at the maximum available network rate rather than at the extraction rate, provided that receiver buffer does not overflow. When the network is over-provisioned, the packet dispatch rate may also be limited with the maximum extractable video rate, to decrease the retransmission traffic without affecting the received video quality.

Index Terms— Video streaming, DCCP, TCP, SVC, adaptation

1. INTRODUCTION

The Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) are the most popular transport protocols used to stream video over IP [1]-[5]. Video Transport Protocol (VTP) is proposed in [6] for adaptive video streaming. The Datagram Congestion Control Protocol (DCCP) [7] is a relatively new transport protocol, implementing bi-directional unicast connections of congestion-controlled, unreliable datagrams. It is designed for applications like streaming media, which does not prefer to use TCP due to arbitrary long delays introduced by reliable in-order delivery and congestion control, and which does not like to implement the complex congestion control absent in UDP. DCCP accommodates a choice of modular congestion control mechanisms, which are the TCP Friendly Rate Control (TFRC) identified by Congestion Control Identifier 3 (CCID3) and the TCP-like Congestion Control identified by CCID2. Several works compare the performance of streaming video over DCCP with TCP, UDP and the Stream Control Transmission Protocol (SCTP), showing promising results [8], [9].

Whichever protocol, DCCP, UDP or TCP, is used for transport, a mechanism should exist to adapt the video rate to the network conditions, in order to achieve better video quality. This adaptation can be performed by using the receiver buffer occupancy information to prevent any buffer underflow/overflow [4] or by combining the receiver buffer state with the bandwidth estimate [10]. [11] employs a virtual network buffer between the sender and the receiver together with end-to-end delay constraints to adapt the video transmitted. Packets may be sent depending on their rate distortion values, as in [12] and [13]. In case DCCP CCID3 is used, the TFRC rate calculated by DCCP can be utilized by the sender to estimate the available network rate.

It is the video coding characteristics that determines how to adapt the video rate to the available bandwidth. The video can be encoded in multiple streams to perform stream switching [11], or it can be coded in multiple layers for layer switching. In this respect, the Scalable Video Coding (SVC) is a promising video format which introduces spatial, temporal and quality scalability [14], [15].

In this work, adaptation strategies for streaming SVC video over DCCP and TCP are investigated and the results are given in terms of received video quality and used network resources. The contributions of the paper are:

- To the best of our knowledge, the work is the first in streaming SVC video using actual DCCP over IP with the adaptation of the video rate to the estimated bandwidth.
- The oscillatory nature of DCCP is discovered when the transmission rate is matched to the video extraction rate, which lowers the quality of the received video.
- An adaptive Automatic Repeat reQuest (ARQ) scheme is proposed, which requests all of the lost packets or only the base layer packets, depending on the decoder buffer state.
- A comparison of transport protocols, DCCP (CCID3, CCID2) and TCP, is presented.

The paper is organized as follows: Section 2 introduces the adaptive streaming framework and Section 3 presents the tests performed. Concluding remarks are given in Section 4.

2. AN ADAPTIVE STREAMING FRAMEWORK

We have implemented an adaptive video streaming framework that works on IP networks and uses DCCP (both CCIDs) or TCP as the transport protocol. The framework,
The application can also average these TFRC rates over an averaging window, to filter out noise. When DCCP CCID2 or TCP is used, the packet sending buffer occupancy and slows its transmission, if necessary.

Several problems of adaptive video streaming are addressed in the paper, which are: i.) determination of the video extraction rate; ii.) control of the transmission rate in the application layer; iii.) retransmission of lost packets.

DCCP CCID3 continuously calculates the TFRC rate based on packet loss and delay statistics during streaming, which can be used by the sender application directly or through an algorithm, to determine the next extraction rate. The application can also average these TFRC rates over an averaging window, to filter out noise. When DCCP CCID2 or TCP used, the sender application has to estimate the available network on its own, without consulting to the transport layer. The implemented framework uses the TFRC rate directly to calculate the extraction rate when CCID3 is utilized, with the option to perform averaging. When DCCP CCID2 or TCP is used, the packet sending rate is calculated to estimate the available bandwidth.

As video packets are extracted, the sender dispatches them to the transport queue for being sent. This dispatch operation can be done without any delay, if possible, or with some delay so that the video extraction rate is matched. This application layer rate control may be necessary if there is decoder buffer overflow risk. Moreover, it may be thought that streaming video at the extracted rate rather than the higher available network rate would result in lower loss, leaving more bandwidth to competing flows. In order to investigate the application layer rate control issue, we have implemented three selectable modes of rate control in the framework, which are: i.) sending the extracted video at the available bandwidth rate (send @ max. rate); ii.) sending video at the extraction rate (send @ ext. rate); iii.) sending video at the available bandwidth rate but not more than the maximum extractable video rate (send @ lim. max. rate).

Although video rate is adapted to the network, packet losses occur when DCCP is used. As the receiver detects missing packets with enough time for replay, it may request these from the sender using an ARQ scheme. Three ARQ methods exist in the framework, which are: i.) request all missing packets (arq all); ii.) request base layer missing packets only (arq base); iii.) request all or base layer missing packets only, depending on the decoder buffer occupancy (arq adaptive). While the first two methods are straightforward, the third method requests all missing packets if there is a risk for the decoder buffer to overflow – meaning that the available network rate is more than the extracted video rate. Otherwise, it requests only base layer missing packets for retransmission.

3. RESULTS

This section presents the results of the tests performed to address the listed problems. The Soccer and City sequences are used throughout the tests, each looped 10 times to have 2880 frames and then encoded with the SVC reference software the Joint Scalable Video Model (JSVM) version 9.19.3, to have a base layer and a Medium-Grained Scalability (MGS) layer, which is fragmented into two sub-layers using the MGS vector (6,10). The GoP size and the intra-period are taken as 16 frames. Byte-limited slice mode is used so that none of the Network Abstraction Layer (NAL) units exceed 1400 bytes, guaranteeing they won’t be fragmented in the IP layer.

Extraction is done GoP-by-GoP, using the hierarchic extraction scheme proposed in [16]. In extraction, base layer packets are always retained, and MGS packets are selected in case rate budget allows. Should a base layer
packet lost during transmission, frame repetition is performed for error concealment. The minimum and the maximum extractable rates (and the corresponding PSNR values) of the Soccer video are 316 kbps (32.2 dB) and 945 kbps (37.0 dB), respectively. Similarly, the minimum and the maximum values for the City video are 320 kbps (33.5 dB) and 855 kbps (38.0 dB). Moreover, the decoder buffer is limited to contain video data worth of 10 seconds at maximum, and the pre-roll delay is guaranteed to be less than 5 seconds.

The tests are run in a controlled LAN environment. In order to simulate the Internet traffic, ns2 network simulator version 2.33 is used in emulation mode, together with the PackMime-HTTP [17] module to generate web traffic. For cross traffic, the rate parameter is set to 10 HTTP requests per second. ns2 is run on a node between the sender and the receiver, connecting the two so that the traffic from sender to receiver is passed through ns2. The bottleneck bandwidth is varied from 500 kbps to 1500 kbps in 100 kbps increments, and at each rate, 15 streaming tests are done. The results are averaged over these tests.

Tests performed with different averaging window lengths show that the TFRC rate received from DCCP CCID3 can be used to calculate the extraction rate without any averaging in most cases, as given by Figure 1. However, averaging improves the received video quality significantly when DCCP CCID2 and TCP are used, because the network estimation is done based on the bursty transmission rate. A comparison of the protocols is shown in Figure 2, where it is apparent that TCP performs the worst, while both CCIDs show comparable results.

Rate control tests reveal that sending video at the maximum or limited maximum rate produces videos with similar quality, whereas sending at the extraction rate generates lower quality videos, especially when there is cross traffic, as shown in Figure 3 and Figure 5. Additionally, limiting the video transmission rate (send @ lim. max. rate) results in less packet loss and therefore less retransmission traffic when the available bandwidth is more than the maximum extractable video rate.

It is interesting to observe that sending video at the extraction rate performs worse than the other methods only at specific network rates, when there is no cross traffic. The reason of this performance decrease can be discovered in Figure 4, where the TFRC rate calculated by DCCP varies significantly when transmission rate is kept at the video extraction rate. Sending video at the extraction rate results in TFRC rate oscillating. This is due to the nature of TFRC rate calculation: Packet loss is used to estimate the available bandwidth, along with RTT, and when a source transmits data at a rate lower than the calculated TFRC rate, less number of packets gets lost and DCCP overshoots the next bandwidth estimate. Then the sender node in our streaming framework uses this overshoot estimate to extract new packets, many of which in turn will be lost. Hence this time DCCP will lower the TFRC rate, resulting in oscillations. The oscillatory behavior of TFRC is experienced when the available network rate is close to the maximum extractable video rate. Before that, when the network rate is low, the video is extracted at the bottleneck rate; hence there is no difference between sending at the extraction rate or at the available rate. When the available network rate is high, the oscillatory behavior of TFRC is compensated, because the
network is over-provisioned. Under cross traffic, sending at the extraction rate performs poorly also at low rates, because cross traffic consumes the traffic share unused by the streaming application.

Performances of the existing ARQ schemes without cross traffic are shown in Figure 6, which reveals that resending all missing packets does not noticeably improve the quality of the received video but increase the retransmission traffic, when the bandwidth is scarce. In case the available network rate is high, however, requesting only base layer missing packets results in lower video quality, because of the lost enhancement layer packets. The proposed adaptive ARQ scheme aims to scale the retransmission traffic, requesting only base layer packets when the network rate is low and requesting all missing packets when the network allows.

4. CONCLUSIONS

In this paper, several adaptation strategies are investigated to improve the received video quality in streaming SVC video. Different bandwidth estimation techniques are implemented for different transport protocols, such as utilizing the TFRC rate when DCCP CCID3 is used or calculating the transmission rate otherwise. Tests performed using various averaging window lengths reveal that averaging improves the video quality when DCCP CCID2 or TCP is utilized, but hardly at all when CCID3 is used.

Controlling the transmission rate in application layer exposed the oscillating nature of DCCP CCID3, decreasing the quality of the received video. Video should be sent at the maximum available network rate rather than sending at the extraction rate. When the network capacity is high, the transmission rate may also be limited with the maximum extractable video rate, to decrease the retransmission traffic without affecting the received video quality.

When dealing with losses, retransmitting only base layer packets are sufficient, except when the network is over-provisioned. A better approach is retransmitting base or all lost packets, depending on the available network rate.

5. REFERENCES


