Efficient Video Streaming Rate Control Based on a Deadline-Sensitive Selection of SVC Layers

Andrés Sanhueza
University of Chile
Email: asanhueza@ing.uchile.cl

Hugo Méric
NIC Chile Research Labs
Email: hmeric@niclabs.cl

Claudio Estevez
University of Chile
Email: cestevez@ing.uchile.cl

***Abstract*—Video streaming over the Internet is challenging due to the varying intrinsic and extrinsic network conditions such as loss rates, throughput and delay. In this work, a cross-layer solution using scalable video coding in collaboration with TCP is proposed. The objective is to ensure that all frames arrive before their deadline by implementing a novel deadline-sensitive discarding policy that adjusts the video stream rate to the available bandwidth. Our proposal thus improves stream fluidity and it also reduces transmission of unnecessary traffic. Simulation results show that no rebuffering occurs, while maintaining a high quality video, in a 70-ms RTT link with a loss rate of 1%.**

I. Introduction

Multimedia streaming is becoming one of the dominant applications of the Internet. A recent study estimates that video traffic will comprise at least 80 percent of all consumer Internet traffic by 2019 [1]. Thus designing a robust and network-friendly video delivery solution is a current exigency.

Multimedia content transmission faces several challenges inherent from networks such as losses, varying throughput and delay. This needs to be surmounted to provide a satisfactory quality of experience (QoE) to the users. Several protocols may be employed at the network level to carry the video stream, for instance the user datagram protocol (UDP) or the transmission control protocol (TCP). Even with the growth of streaming applications which often use UDP, TCP flows still represent the majority of the Internet traffic [2]. One reason is that enterprises are implementing packet filtering, particularly of UDP packets, to avoid unrestrained congestion-unaware data flows and the ingress of unsolicited UDP packets. As a corollary, UDP-based applications are transitioning to TCP, a congestion-aware protocol. Concepts such as media-friendly and TCP-friendly rate control are ameliorating the new video streaming protocol designs but are still rarely used.

This work focuses on adaptive video streaming solutions where a stored video file is encoded and transmitted in a unicast fashion using TCP. The objective is twofold: (1) to vary the video quality with changing network conditions and (2) to avoid re-buffering as it severely impairs the QoE [3]. To that end, we propose a rate control algorithm that adjusts the transmission rate by discarding parts of a scalable video bitstream. This is the main contribution of this work. To the best of our knowledge, a rate control algorithm that matches scalable video coding (SVC) video bitrate to the classical TCP throughput was never studied before.

The paper is organised as follows: Section II introduces related work on video transmission over networks. We present our main contribution, a deadline-sensitive system for scalable video streaming over TCP, in Section III. We evaluate the performance of our proposal through simulations in Section IV. Finally, we summarize the results and discuss future research directions in Section V.

Terminology. A segment is the transport layer unit, broadly speaking a packet. The available segment transmission rate (ASTR) is the instantaneous link capacity due to (mainly) the congestion in the network. Loosely speaking, this can also be referred as the bandwidth.

II. Related Work

Several approaches exist to adapt the video throughput to the available bandwidth. Many proposals rely on cross-layer solutions between the transport protocol and the application layer. In [7], the authors studied the interactions between layered video and congestion control for video streaming. However the proposed solution relies on MPEG-4, scalable video coding (SVC) did not exist at that time, and on binomial congestion control. Starting from different optimisation problems to transmit video on the Internet, several control policies are derived in [8]. Once again, the evaluation relies on MPEG-4. In [5], an external cross-layer input (the bandwidth for instance) controls the video encoder by changing the coding parameters. The solution is applied to a single layer H.264 encoded video over datagram congestion control protocol (DCCP). A streaming system using SVC and providing a congestion control algorithm is introduced in [9]. Finally, a well-known technique is the dynamic adaptive streaming over HTTP (DASH) [10]. This solution considers the HTTP protocol (above TCP or UDP) to send the video. The media presentation on the HTTP server generally requires to encode the video several times. Moreover DASH adapts the quality for each video segment (time scale in s), while our proposal reacts at every congestion windows (time scale in ms).

Compared to previous work, our proposal aims to exhibit a low complexity and to be the least invasive possible. At the server side, the video is encoded one time only using the SVC concept. Moreover, our solution directly works above TCP with no need to modify its behaviour. To the best of our knowledge, it is the first proposal that combines SVC with the classical TCP algorithm.

978-1-5090-4749-9/17/$31.00 ©2017 IEEE

III. Scalable video streaming over TCP

Scalable Video Coding Overview. The SVC extension of the H.264/AVC standard enables to generate a multi-layer video bitstream with a moderate increase in encoding/decoding complexity relative to single-layer coding [4]. The standard offers three types of scalability: temporal, spatial and quality. This means that a substream obtained by dropping packets from the original bitstream results in a lower frame rate, lower resolution or lower quality video signal. Our proposal takes advantage of this feature by discarding packets to adjust the video stream rate to the ASTR. In our work, we only consider quality scalability; nevertheless the extension to temporal and spatial scalabilities is also feasible.

In video coding, a group of pictures (GOP) is a group of successive frames within a coded video stream. The GOP contains at least one intra-coded frame (I-frame), which is compressed independently of the other frames, predictive (P-frame) and/or bi-predictive (B-frame) frames. Fig. 1 depicts the GOP structure used in the simulations. We consider a GOP size of 8 frames with two quality layers per frame. The B-frame encoding and decoding dependencies are also illustrated.

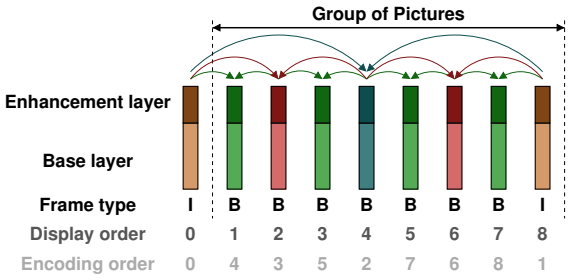


Fig. 1: Scalable video coding with two quality layers. The arrows represent the dependencies between the frames.

SVC over TCP. A cross-layer mechanism that links SVC with TCP can efficiently adapt the video bitstream to the ASTR in real-time by selecting the transmitted segments. The instantaneous throughput is determined by the congestion window size ($cwnd$) and round trip time (RTT). TCP is aware of both of these parameters: it controls the $cwnd$ and constantly estimates the RTT. Meanwhile, SVC has knowledge of the frame types, frame layers and video segmentation. By combining this information, some segments may be omitted to improve the fluidity of the video, i.e., avoiding rebuffering events at the receiver. Our discarding policy takes into account the type (I or B), the quality layer (base or enhanced) and the deadline of the frame contained in each segment.

Discarding policy. A central problem in video streaming arises when the transmission rate is lower than the bitstream rate. In that case, two solutions are typically implemented: rate adaptation at the transmitter side or rebuffering at the receiver side. In this work, we focus on the former solution as we propose to regulate the stream rate by discarding packets in the video stream. Two questions naturally arise: when should we discard segments? and which segment should be discarded?

Several metrics may signal the need to perform rate adaptation. For instance, the authors in [5] rely on the instantaneous throughput. Our solution uses the time difference Δ_F between the estimated arrival time of a frame F and its deadline. When the estimated arrival time approaches the deadline, we start discarding segments based on their content.

The discarding prioritization is straightforward as SVC generates dependencies during the encoding. Consider a SVC-encoded video with two quality layers, base and enhanced, as illustrated in Fig. 1. Video decoding is not possible without the base information making it high priority; therefore the enhanced layer is low priority. Moreover if the base layer of a I-frame is not received, the I-frame will be lost and this generates errors in two GOPs. If the base layer of a B-frame is lost, the B-frame is lost and artefacts are present in the corresponding GOP. In that case, the number of frame affected depends on the frame position in the GOP. For instance in Fig. 1, if frame 2 (encoding order) is lost, artefacts are present in frame 3 to frame 8; if frame 4 is lost, no other frame is affected. This also suggests that B-frames data should generally be discarded before I-frame data.

Based on the previous discussion about data priorization, Table I resumes the proposed discarding policy. The different thresholds (in the left column) have been chosen experimentally. On one hand, they enable to quickly react to low bandwidth; on the other hand, they avoid discarding segments unnecessarily, i.e., segments that still may be decoded on time.

TABLE I: Discarding policy based on the packet content and the time difference Δ_F between the estimated arrival time of each frame and its deadline

	Base layer		Enh. layer	
	I frame	B frame	I frame	B frame
$0 \leqslant \Delta_F \leqslant 5$ RTT	✓	✗	✗	✗
5 RTT $\leqslant \Delta_F \leqslant 10$ RTT	✓	✓	✗	✗
10 RTT $\leqslant \Delta_F \leqslant 15$ RTT	✓	✓	✓	✗
$\Delta_F \geqslant 15$ RTT	✓	✓	✓	✓

Example. We illustrate our discarding policy through an example in Fig. 2. The transmitter searches to fill a congestion window of 7 segments. In the transmitter buffer, the first seven segments correspond to only one frame called F_1. We start by computing Δ_{F_1} based on the knowledge of the segment content and the RTT (more details are given in the next paragraph about the computation of Δ_F for any frame F). Assuming that 5 RTT $\leqslant \Delta_{F_1} \leqslant 10$ RTT, the transmitter fills the $cwnd$ with packets verifying this discarding policy (i.e., segments containing base quality data) and discards the other. If new frames are introduced in the $cwnd$ during the filling operation, we do not compute their estimated arrival time. The estimated arrival time is only computed for the frames initially present (fully or partially) in the $cwnd$ and we apply the most restrictive discarding rule using Table I.

Finally the algorithm transmits three base quality frames rather than one entire high quality frame. This behaviour is representative of all simulation scenarios where the ASTR is

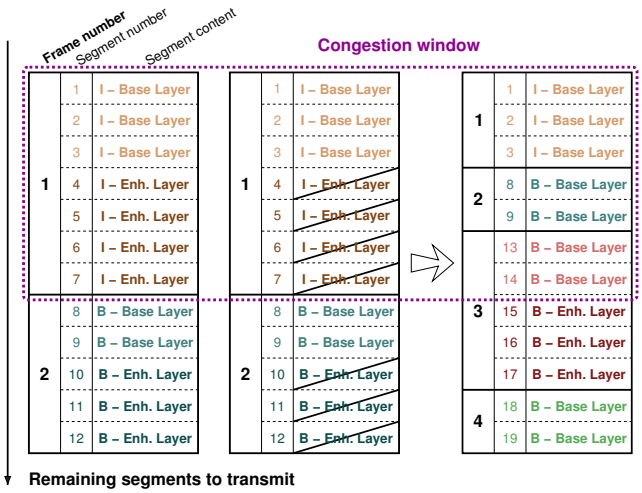


Fig. 2: Discarding frame policy to fill a congestion window of 7 packets and assuming that 5 RTT $\leqslant \Delta_{F_1} \leqslant$ 10 RTT

lower than the video stream bitrate.

Computation of Δ_F. The computation of Δ_F requires two components: the deadline and the estimated arrival time of each frame. The frame deadline depends on the starting time of the streaming, the playout buffer length, the frame rate and the frame number. Its computation is straightforward.

The computation of the estimated arrival time works as follow: assume that at least one segment containing data relative to frame F is initially present in the $cwnd$. The first step is to identify all the segments corresponding to F content. Then we determine how many congestion windows are required to transmit these segments. Two cases are possible: all the packets fit or not in the current $cwnd$. In the first case, one congestion window is necessary and the estimated arrival time is given by $t_c + RTT/2$, where t_c is the current time. In the second case, the transmission of F requires several congestion windows. However the evolution of the $cwnd$ size is unknown due to the losses. To tackle this issue, we assume that no loss occurs during the transmission enabling to compute the (minimum) number of congestion window required to transmit F, noted w_F. Then the estimated arrival time of frame F is equal to $t_c + (w_F - 1) \times RTT + RTT/2$.

We conclude this section with two important remarks: (1) when losses occur, the packets are not acknowledged and we keep them in the buffer to transmit and (2) the estimated arrival time of each frame is refreshed at each $cwnd$ if necessary.

IV. Simulation Results

Simulation setup. The proposed deadline-sensitive video streaming rate control is assessed using Joint Scalable Video Model (JSVM), the reference software for SVC coding, and MATLAB. In the simulations, the 900-frames video *Highway* with CIF resolution[1] is encoded at 30 frames per second resulting in a stream of 30 seconds. The GOP size is set to 8 and we consider two quality layers (base and enhancement). The quantization parameters relative to the base and enhanced quality layers are 40 and 20, respectively.

After the encoding, JSVM generates a *trace file* that summarizes the information about the encoded video stream. This trace file is used by MATLAB, which implements the TCP congestion control and the proposed rate control algorithm. The congestion control is based on the traditional TCP behaviour: it reflects the network state (for instance a large loss rate is representative of a congested network). MATLAB outputs the segments that arrive before their deadline at the receiver.

The last stage is the video decoding, achieved by JSVM. If a frame base layer is not available when the frame deadline arrives, the frame is lost. In that case, the frame is replaced by a blank frame as JSVM does not implement error concealment algorithms. Fig. 3 illustrates the impact of a frame loss. The blank frames maintain the same video length between the transmitter and the receiver; it also enables the JSVM decoder to run properly. Finally, it is assumed that the travel time of each congestion window is equal to RTT/2 and a frame is received if the receiver possesses all the segments that compose it.

(a) Blank frame replacing a lost frame

(b) Artefacts due to a frame loss

(c) Artefacts due to a frame loss

Fig. 3: Examples of video frames after a frame loss (i.e., a frame that did not meet its deadline). For each figure, the original picture is on the left while the reconstructed frame is on the right. In Fig. 3a, a frame whose base layer did not meet its deadline is replaced by a blank frame. Due to error propagation, this may generate artefacts in some frames (depending on the position of the lost frame in its GOP).

[1]Reference YUV video sequences: http://trace.eas.asu.edu/yuv/index.html

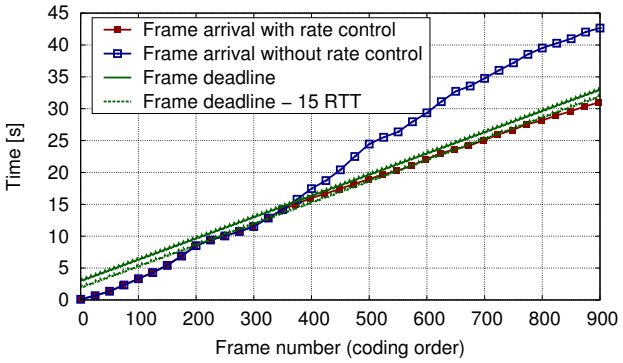


Fig. 4: Frame arrival time with and without rate control (RTT $= 70$ ms, loss rate $= 0.01$ and buffer of 3 s)

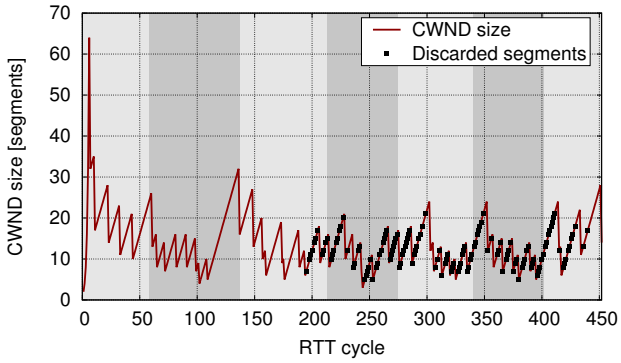
Results. The first simulation results consider a RTT of 70 ms, a loss rate of 0.01 and a playout buffer of 3 seconds at the receiver side. Fig. 4 presents the frame arrival time with and without rate control. First, we remark that the frame deadline curve is not strictly increasing. This is because we consider the coding order and not the display order (see Fig. 1). Then the results show that the proposed rate control algorithm enables to meet the frame deadline for all frames. When no rate control is used, the frame arrival time exceeds the frame deadline after transmitting about 370 frames. Moreover at the end of the transmission, a time gap of about 12 seconds exists between the two schemes. We also remark that the proposed scheme keeps a margin with the deadline, this ensures a quality smoothness at the receiver.

Fig. 5 shows the congestion window size and the frame quality in terms of luminance peak signal to noise ratio (Y-PSNR) for a simulation with the same parameters as before: a RTT of 70 ms, a loss rate of 0.01 and a playout buffer of 3 seconds. We choose the Y-PSNR as the human eye is more sensitive to the brightness information. Fig. 5b also depicts the mean opinion score as defined in [6]. In both figures, we present the discarded segments in order to see the behaviour of our rate control algorithm. In Fig. 5a, the discarded segments do not half the congestion window because it is a consequence of our rate control algorithm, this is not a loss in the network.

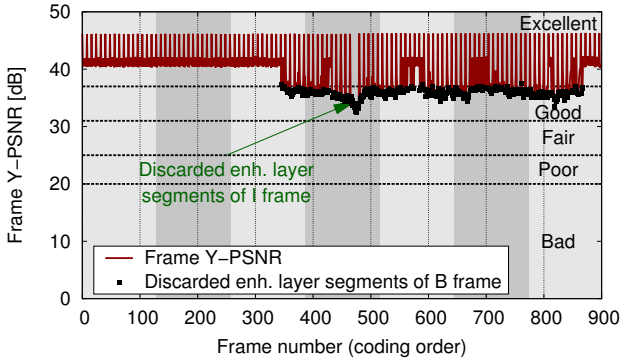
At the beginning of the streaming, there is no discarded segment and the video quality is excellent with a Y-PSNR above 40 dB. This is due to the presence of the playout buffer.

Around the congestion window #200, the rate control algorithm starts to discard segments and the video quality decreases. In this simulation, the discarded segments contain mainly the enhanced layer of B-frames. Indeed, only one I-frame was affected by the removal of its enhanced layer (see Fig. 5b). Moreover no segment containing base layer information (for I or B-frames) was discarded. This means that the rate control algorithm always kept a 10 RTT margin with the frame deadline. Finally, the resulting video quality oscillates between good and excellent.

In Table II, we present the impact of the RTT on: (1) the number of frames affected by base or enhanced layer removal,



(a) Congestion window size



(b) Frame Y-PSNR

Fig. 5: Simulation results with a loss rate of 0.01, a RTT of 70 ms and a playout buffer of 3 s. The background color enables to match the RTT cycles with the transmitted frames.

(2) the difference between the last frame arrival time with or without rate control and (3) the average Y-PSNR. The loss rate is set to 0.1 and the playout buffer to 3 s. When the RTT is 70 ms, we retrieve the results presented in Fig. 4 and Fig. 5. The results point out that the proposed algorithm obtains good performance up to 150 ms as the video quality is still good. The video quality decreases heavily for a RTT of 175 ms, this is mainly due to the losses of many B-frames base layer. In that case, two solutions are possible: modify the proposed algorithm (for instance the different discarding thresholds) or encode the video with a larger quantization parameter.

TABLE II: Simulations results for various RTT values (loss rate of 0.01 and playout buffer of 3 s)

RTT	Base layer discarded		Enh. layer discarded		Time	Average
	I frame	B frame	I frame	B frame	Gap	Y-PSNR
50 ms	0	0	1	143	2.4 s	40 dB
70 ms	0	0	1	339	11.6 s	38 dB
100 ms	0	0	25	462	20 s	37 dB
150 ms	0	0	78	705	61.1 s	35 dB
175 ms	0	87	95	771	70.7 s	15 dB

V. Conclusion

A deadline-sensitive rate-control algorithm for scalable video streaming is presented and evaluated. As the frame

arrival time approaches its deadline, the proposed solution discards segments in a SVC stream based on the segment content. Our technique works over TCP. The concept is novel and proven to efficiently control the bitstream to produce a continuous video stream (no re-buffering), while maintaining a high quality video under bandlimited scenarios. The cross-layer interaction between SVC and TCP improves the QoE and reduces unnecessary traffic in the underlying network.

Future work will extend the current results in several directions. First, we will investigate the impact of the system parameters (discarding thresholds, loss rate, playout buffer, TCP version, etc.) on the video quality. We also plan to use rate-distortion studies in order to improve our discarding policy. Finally, the potential benefits of using the late-arrival segments [11] will be researched.

REFERENCES

[1] "Cisco visual networking index: Forecast and methodology, 2014-2019," CISCO, Tech. Rep., 2015.

[2] D. Lee, B. E. Carpenter, and N. Brownlee, "Observations of UDP to TCP ratio and port numbers," in *Internet Monitoring and Protection (ICIMP), International Conference on*, 2010, pp. 99–104.

[3] O. Oyman and S. Singh, "Quality of experience for HTTP adaptive streaming services," *Communications Magazine, IEEE*, vol. 50, no. 4, pp. 20–27, 2012.

[4] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, 2003.

[5] G. Sarwar, R. Boreli, and E. Lochin, "Xstream-x264: Real-time H.264 streaming with cross-layer integration," in *Multimedia and Expo (ICME), IEEE International Conference on*, 2011, pp. 1–4.

[6] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid – a framework for video transmission and quality evaluation," in *13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, 2003, pp. 255–272.

[7] N. Feamster, D. Bansal, and H. Balakrishnan, "On the interactions between layered quality adaptation and congestion control for streaming video," in *International Packet Video Workshop*, 2001.

[8] P. de Cuetos, "Network and content adaptive streaming of layered-encoded video over the Internet," Ph.D. dissertation, Institut Eurécom, 2003.

[9] D. T. Nguyen and J. Ostermann, "Congestion control for scalable video streaming using the scalability extension of H.264/AVC," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 2, pp. 246–253, 2007.

[10] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP - Standards and Design Principles," in *ACM Conference on Multimedia Systems*, 2011, pp. 133–144.

[11] J. Xiao, T. Tillo, C. Lin, Y. Zhang, and Y. Zhao, "A real-time error resilient video streaming scheme exploiting the late- and early-arrival packets," *Broadcasting, IEEE Transactions on*, vol. 59, no. 3, pp. 432–444, 2013.