Video Quality Estimation for Internet Streaming

Amy Reibman
AT&T Labs-Research
Florham Park, NJ 07932
amy@research.att.com

Subhabrata Sen
AT&T Labs-Research
Florham Park, NJ 07932
sen@research.att.com

Jacobus Van der Merwe
AT&T Labs-Research
Florham Park, NJ 07932
kobus@research.att.com

Categories and Subject Descriptors: C.2.2 [Computer Communication Networks]: Network Protocols; C.4.2 [Performance of systems]: Measurement Techniques

General Terms: Measurement, Performance

Keywords: Video Quality, Streaming, Performance, Network Measurement

1. INTRODUCTION

As broadband access connectivity becomes more prevalent, more users are able to stream video over the Internet. However, the best-effort service model and shared infrastructure of IP networks means network impairments (such as delays, jitter, congestion, and loss) have the potential to hamper viewing experiences. Network service providers would like an online performance monitoring tool in order to create a real-time understanding of video streaming applications. This can aid in monitoring compliance of service-level agreements (SLAs) between Internet Service Providers (ISPs), hosting centers, and content providers; alert operators to potential performance problems; and help in root-cause analysis and debugging.

In this paper, we consider the problem of predicting application-level video quality by monitoring video packets within the network. One particular focus is to be able to identify when the encoded bitstream has been modified (by either server or network) enough to impair the displayed quality. We denote all such cases as "bitstream impairments". Another aspect that we consider is the "original encoded" quality of the video when there are no network induced impairments. We focus on the widely used proprietary Windows media streaming format for this study.

Our work is unique in studying the video quality of Internet streaming content by extracting quality related information from different levels in the protocol stack, and by analyzing the quality of streaming content observed at a broadband access network. A number of earlier experimental studies have addressed the *network level* quality of service (delay, loss, jitter etc.) experienced by multimedia streams transmitted across networks [1, 2]. A number of earlier works explore computing video quality using information gathered from video pixels [3, 4]. But these works either rely on access to the original video and/or require expensive decoding of the received stream, requirements which make them unsuited to network environments where the original video is not available and computation scalability is a must.

The contributions of this paper are twofold. First, we develop techniques to extract information related to video quality from video packets observed at an in-network monitoring point. Our approach involves processing different levels of information in the packet stream, encompassing the network, transport and application payloads. Second, we apply our techniques and framework to perform a measurement study of the performance experienced by streaming video sessions observed at a broadband access network.

WWW 2005, May 10–14, 2005, Chiba, Japan.
ACM 1-59593-051-5/05/0005.

2. STREAMING VIDEO QUALITY ESTIMATION

In general, perceived video quality depends on many factors, some intrinsic to the displayed video itself, and others depending on the viewer and his or her environment. Here, we focus on the first set of factors; namely, we would like to objectively characterize the quality of the displayed video. Video quality is determined (a) by how well the video was encoded and (b) by what impairments the video was subjected to, while streamed across the networks. The encoding parameters impose an upper bound on the quality of a particular stream. Any impairments incurred by the video reduces its quality from this initial upper bound. For example, missing frames or buffer violations (or other temporal deviations from the intended display times) would reduce the video quality. Further, switching to a lower-rate stream would immediately reduce the achievable quality because a new upper bound would be imposed by the lower encoding rate.

For our study, we monitor video packets inside the network. For each packet, we have the entire content and the time at which it was observed. To obtain the most information regarding network conditions and video quality, a video quality tool should extract information from every level of the protocol stack and further process as deep into the video packets as possible. For the work presented in this paper the innermost level from which we extract information is the Advanced Systems Format (ASF) [5] protocol layer. This layer consists of an interleaved sequence of compressed digital video frames and associated audio segments. The next level of encapsulation involves adding Microsoft Media Streaming (MMS) [6] protocol-specific information to the ASF data stream to create MMS packets. One or more ASF packets are typically encapsulated in the payload of a single MMS data packet. The outermost two layers we consider correspond to the Internet transport (TCP or UDP) and network (IP) protocols respectively.

2.1 Extracting ASF packet observations

Each ASF transmission, or file, starts with a header describing file information (including the file GUID, overall file duration, and overall file bit-rate), the number of available audio and video streams within the file, and summary information for each audio (sample rate) and video stream (spatial resolution, compression algorithm) [5].

We can also obtain the following information regarding the media-objects within each ASF data packet: the number of bytes in media-object j from stream number i; a flag indicating if this media object is a key frame; the presentation time for this media object; and the send, monitoring, and arrival times of the last ASF packet which contains information about this media object. The arrival time at which the client itself receives all information about this media-object may not be exactly observable. We make the simplifying assumption here that there is no extra jitter on the path between the monitoring point and the client. By mining TCP-level information we reduce the errors in this estimation.

2.2 Identifying Video Quality Impairments

We next show how values extracted from the ASF information can

be used to identify two types of server rate adaptations and to identify sub-optimal client-buffer behavior for each stream.

2.2.1 Detecting server rate adaptation

Two types of rate adaptation are typically seen. In the first, the ASF file contains multiple video streams. When the server receives a request to reduce the transmitted rate, it switches from one video stream to a lower-rate video stream at the next key-frame. In this case, the new rate is nearly constant with time. We call this type of rate change *stream switching*. In the second type of rate adaptation, the server reduces the rate by suppressing the transmission of non-key frames, i.e., a *key-frame-only* rate change. At the client, the audio still plays, but the video display becomes a series of still frames, often several seconds apart. Now, because key-frames can appear with arbitrary frequency in the video bitstream, the transmitted rate is bursty. In cases of severe network congestion, the server can also stop sending any video frames at all. In this case, the last received video frame is displayed while audio is played.

It is a simple matter to observe the stream IDs of media-objects to determine when rate adaptation via stream-switching has occurred. To detect the case where non-key frames have been discarded, we look for gaps in the sequence of j, the media-object ID for video stream i. Gaps caused by a lost packet are typically followed by a non-key frame, while gaps caused by intentional server rate adaptation are followed by a key-frame. The visual impact of both are similar: a freeze-frame. Gaps in the audio streams can be found the same way, indicating audio glitches at the client.

2.2.2 Understanding client-buffer behavior

The best-effort nature of IP networks typically introduces a fair amount of jitter during the delivery of audio and video streams. To accommodate for this jitter, streaming applications accumulate a playout buffer before playback starts in an attempt to "ride-out" any jitter introduced by the network and/or delivery protocols. When this buffer runs dry, i.e., when there is no content to play out, the video stream effectively comes to a halt which causes a very negative viewing experience.

To analyze the client-buffer behavior for a particular stream, it is necessary to know the sequence of both the arrival times and presentation times for each frame. Once this information is obtained by parsing the ASF packets, it is straightforward to emulate any specific client-buffer strategy to evaluate its performance. However, inside the network, we can never know the specific strategy used by the client to manage its buffer. For this reason we apply two methods in examining buffer related issues. First, we take a *model-independent* approach that derive properties that are independent of the actual client buffer strategies and provide an upper bound on the amount of buffering needed by a player to avoid buffer related impairments. Second, we assume one specific client playout strategy, to derive *model-dependent* properties. We now explain these approaches.

For the model-independent approach, we focus on detecting when the arrival times – in relationship to the intended presentation times – force the client to show frames at times other than those intended by the encoder, since this will degrade the temporal rendition of the video. The client will be forced to alter the display durations if any frame arrives after its original intended display time. For our model-independent evaluation, we characterize Δ_{max}, the minimum initial startup delay a client had to tolerate to ensures that all frames arrive ahead of their intended presentation times.

For the model-dependent approach we assume a media player model where initially the client waits for a pre-specified amount (say X seconds) of data to arrive before it begins playback. At this point, we assume the following client behavior: (i) The client starts decoding and playback. (ii) At any time, as long as there is any data in the buffer, the client will try to decode it. (iii) Only if the buffer is completely empty, will there be a buffer starvation. (iv) After a starvation occurs, the client resumes playback when the next full frame arrives in the buffer. We assume that the client does not wait for the buffer to fill up to a preset point before resuming playback. (v) If however, the user gives an interactive request, then the client will again wait to receive the initial amount of data before restarting playback.

2.3 Encoded Video Quality

The quality of the video displayed to the viewer is upper bounded by the quality intended by the encoder, and any impairments introduced by the server, network, or client can only reduce the quality. We characterize spatial quality of individual frames using four parameters: the spatial resolution (height and width of each image), the compression algorithm, the encoded temporal frame rate, and the number of compressed bits per coded pixel (*cbpcp*). We can also characterize temporal quality using three parameters: the encoded frame rate (i.e., how many frames are intended by the encoder), the received frame rate (i.e., how many frames can be decoded by the decoder), and any temporal deviations in the intended presentation times.

3. RESULT SUMMARY

We next summarize our main observations from the study, which examined one week of Windows Media video traffic observed at a cable headend in March 2004.

Quality without impairments: Over 42% of the videos used Windows Media 9, and over 37% of the videos used Windows Media 8. While there were over 74 distinct spatial resolutions used, over 67% were 320 by 240. Newer codecs typically used higher frame rates and greater spatial resolutions, creating higher quality videos.

Impairments: Most streaming segments in our data (86%) experienced no impairments. Videos with longer playtimes were more likely to experience impairments. Live videos were more likely to experience impairments than on-demand videos. 59% of impaired streams had a key-frame-only rate change; 16% of impaired streams had a stream-switching rate change; and 62% of impaired streams had buffer impairments.

Buffer impairments: For 90% of the videos, a 5-second startup delay would have prevented any temporal impairments. However, 1% of videos required over a 45-second startup delay. 95% of the videos receive X seconds of data in less than 1.5 times X seconds, for X=5,10,15, and 30 .

Rate-change impairments: Of those videos that experienced some form of rate change, 95% of these experienced fewer than 5 rate changes. 60% of videos experiencing buffer starvation also experienced some form of rate reduction. However, the type of rate change (stream-switching or key-frame-only) does not have a significant impact on either the overall playback duration or the mean duration between rate changes.

Our future work will extend our video quality estimation to map the information observed in the network into user perceived video quality metrics.

4. REFERENCES

[1] J. M. Boyce and R. D. Gaglianello, "Loss effects on MPEG video sent over the public Internet," in *Proc. ACM Multimedia*, September 1998.

[2] D. Loguinov and H. Radha, "Measurement study of low-bitrate Internet video streaming," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[3] M. A. Masry et al., "A metric for continuous quality evaluation of compressed video with severe distortions," *Signal Processing: Image Comm.*, vol. 19, pp. 133–146, February 2004.

[4] A. Webster et al., "An objective video quality assessment system based on human perception," in *Proc SPIE, vol 1913*, pp. 15–26, 1993.

[5] Microsoft Corporation, "Advanced systems format (ASF) specification, rev. 1.20.02." `http://download.microsoft.com`, June 2004.

[6] Streaming Download Project, "MMS streaming protocol." `http://sdp.ppona.com/`.